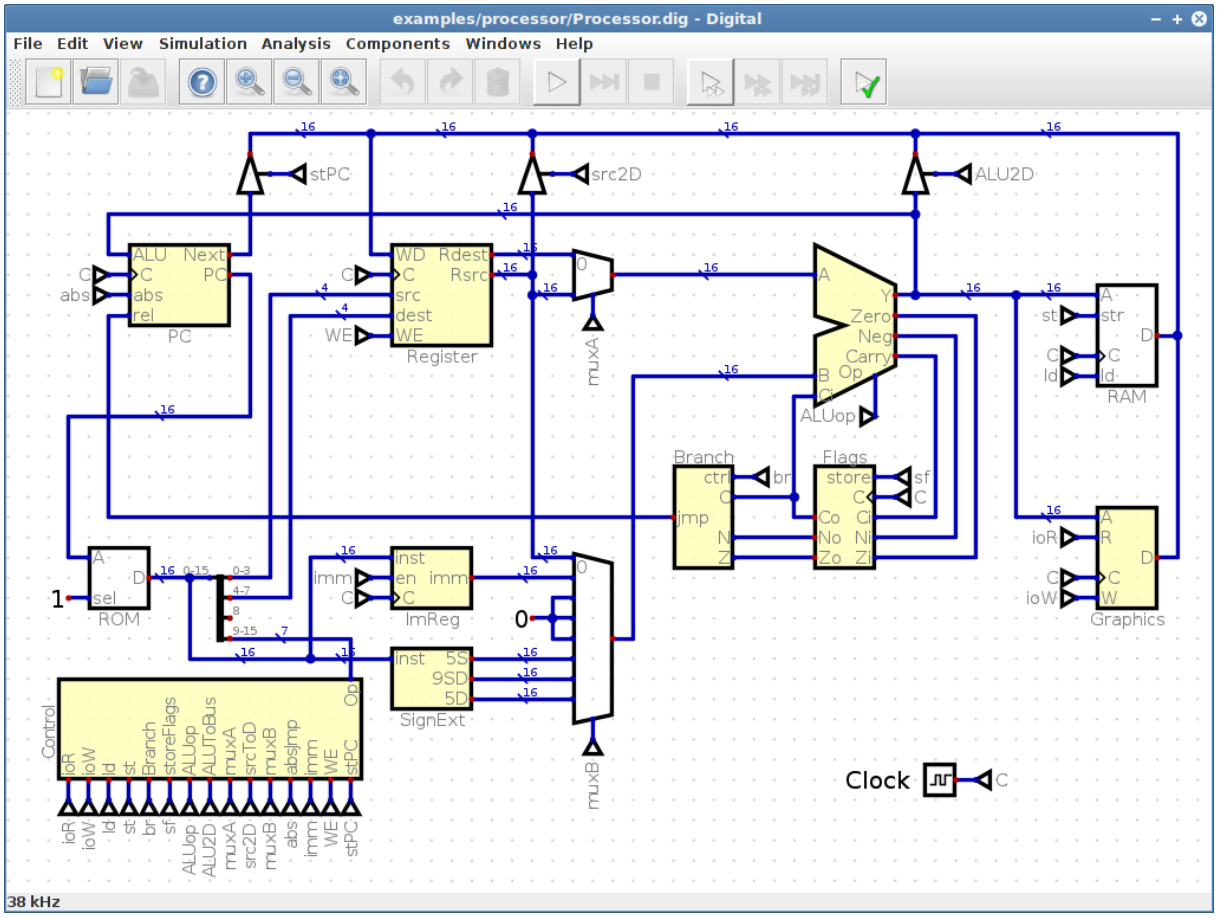
Digital



Version: v0.26.1
Datum: 2021-02-26 07:18

Inhaltsverzeichnis

A Allgemein

1. Digital

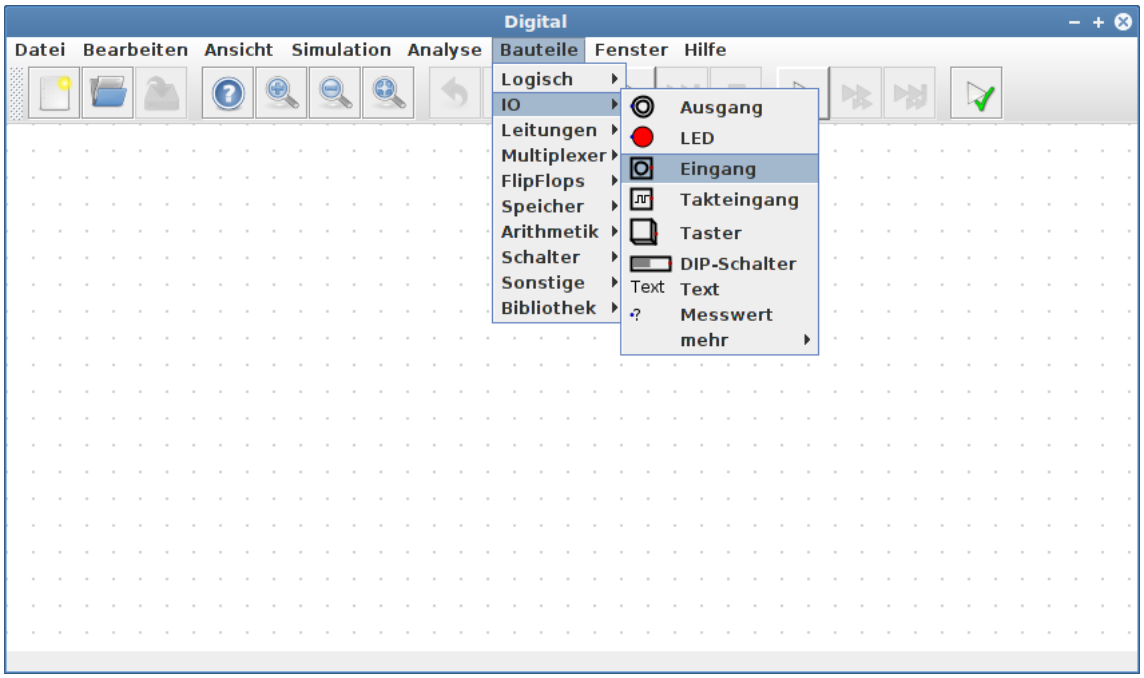
1.1. Einführung

Digital ist ein Simulator für digitale Schaltkreise. Ein Schaltung besteht dabei aus einer Anzahl logischer Gatter, welche miteinander durch Leitungen verbunden sind. Das Verhalten der Gesamtschaltung kann simuliert werden. Der Anwender kann mit der laufenden Simulation über bestimmte Elemente interagieren, indem er z.B. Knöpfe betätigt oder Eingangswerte setzt.

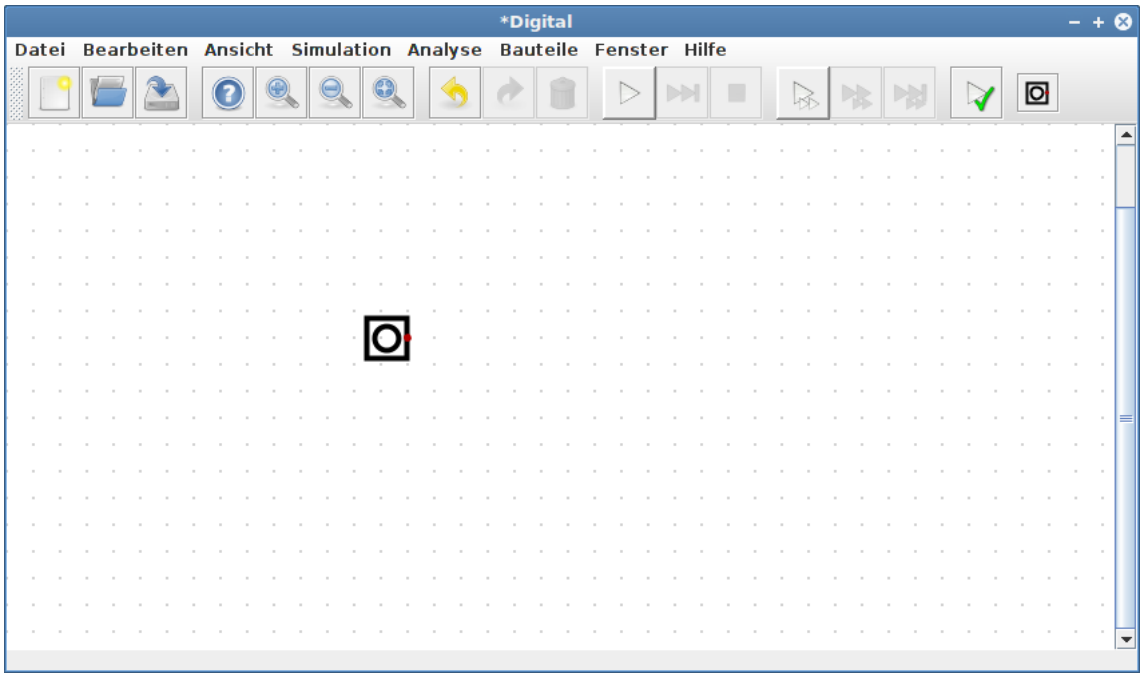
Auf diese Weise können die meisten Grundschaltungen der Digitaltechnik aufgebaut werden. Im Ordner *examples* finden sich viele Beispiele bis hin zum funktionsfähigen 16-Bit Single Cycle Harvard Prozessor.

Der Simulator kennt dabei zwei Modi: Im Bearbeitungsmodus kann die Schaltung bearbeitet werden. In diesem Modus ist die eigentliche Simulation nicht aktiv. Der Simulationsmodus wird durch den *Start*-Knopf aktiviert. Beim Wechsel in diesen Modus wird die Schaltung zunächst auf Konsistenz überprüft. Enthält die Schaltung keine Fehler, wird die Simulation aktiviert. Jetzt kann mit der laufenden Simulation interagiert werden. Möchte man die Schaltung weiter bearbeiten, muss man die Simulation zunächst wieder stoppen.

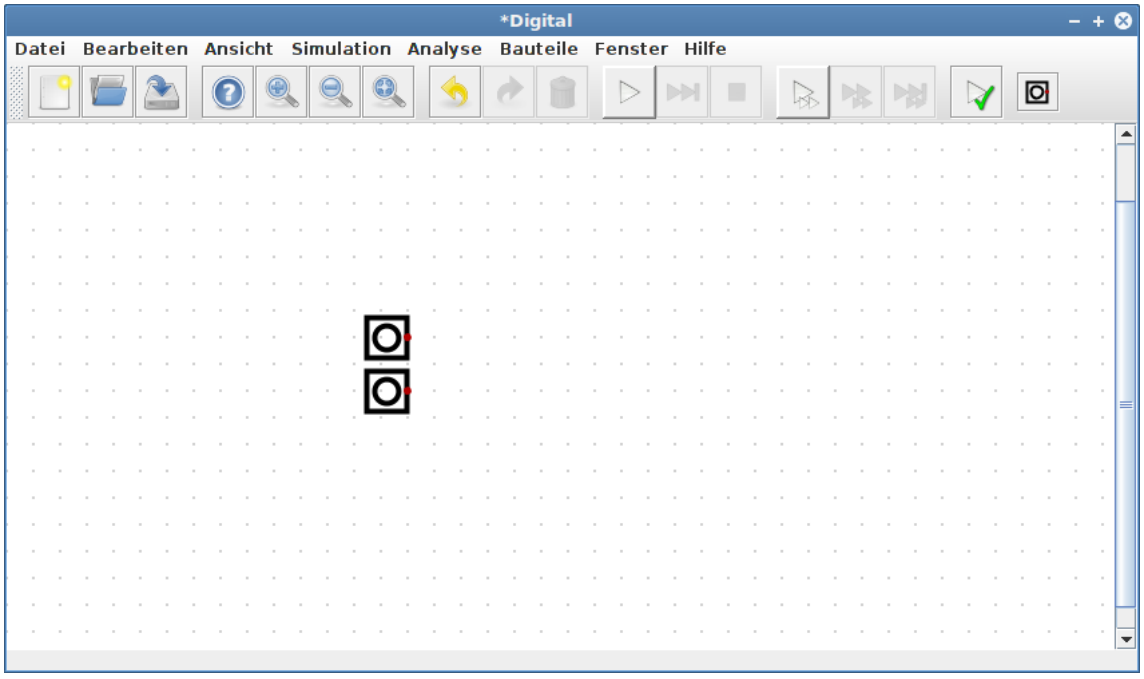
1.2. Erste Schritte



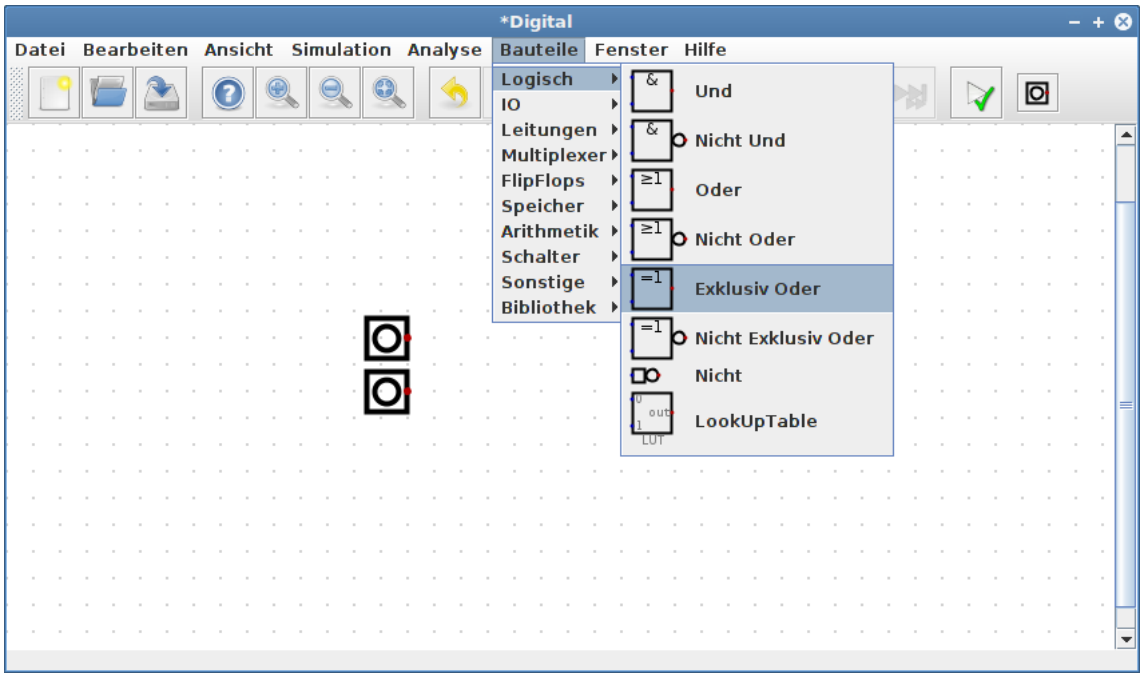
Als erstes Beispiel soll eine Schaltung mit einem Exklusiv-Oder Gatter aufgebaut werden. Im Hauptfenster erlaubt das Menü *Bauteile*, die verschiedenen Komponenten auszuwählen. Danach werden diese auf dem Zeichenfeld positioniert. Das Positionieren kann mit der ESC-Taste jederzeit abgebrochen werden. Als erstes soll der Schaltung ein Eingang hinzugefügt werden. Dieser lässt sich später interaktiv mit der Maus steuern.



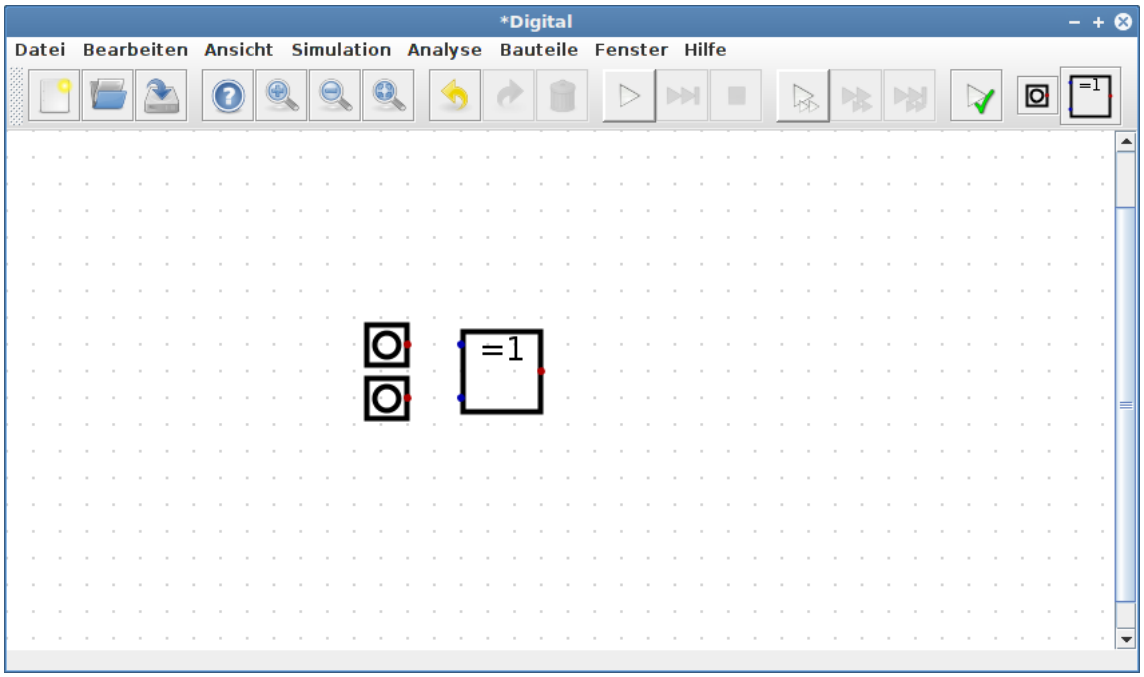
Nach der Auswahl kann der erste Eingang auf die Zeichenfläche gesetzt werden. Der rote Punkt markiert den Leitungsanschluss. Hier wird später eine Verbindungsleitung angeschlossen. Rot zeigt an, dass dieser Anschluss aktiv ist. Er definiert also einen Signalwert.



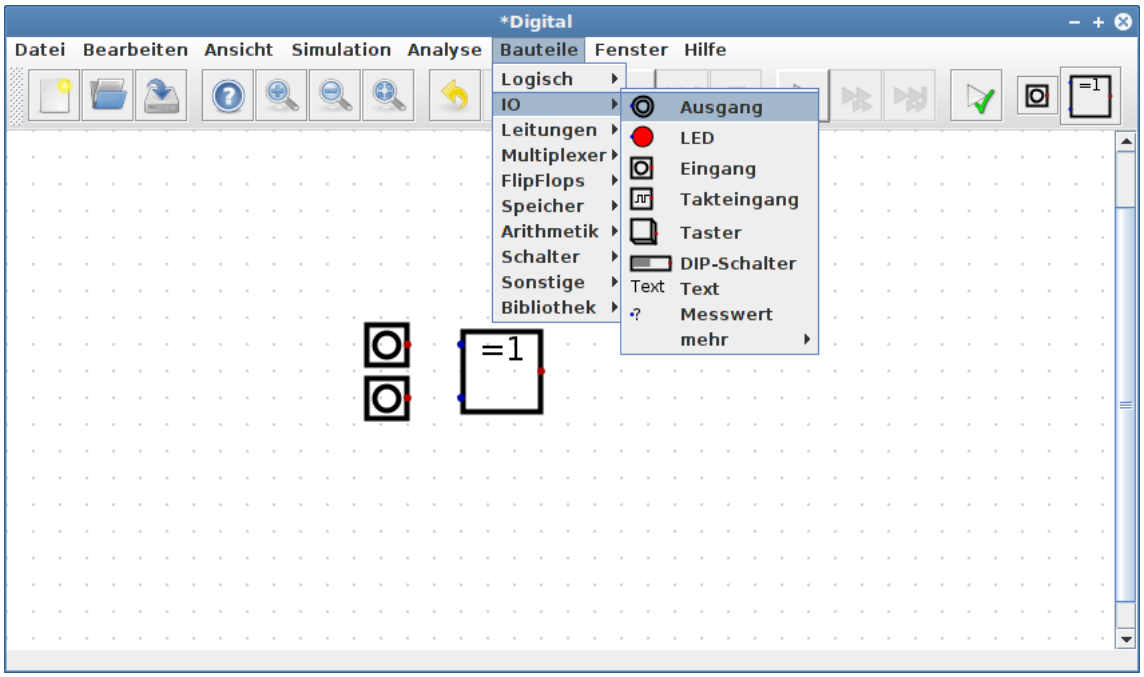
Auf die gleiche Art und Weise wird ein zweiter Eingang hinzugefügt. Am besten setzt man ihn direkt unter den ersten Eingang.



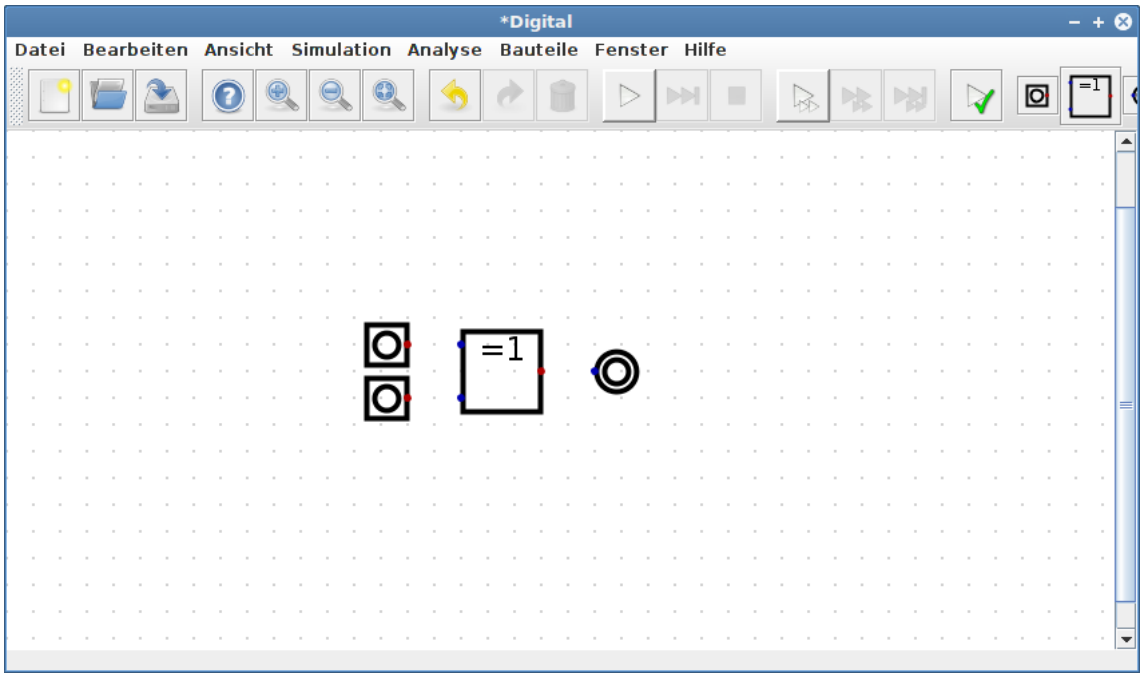
Danach wird das Exklusiv-Oder Gatter ausgewählt. Dieses Gatter stellt die eigentliche logische Funktion dar.



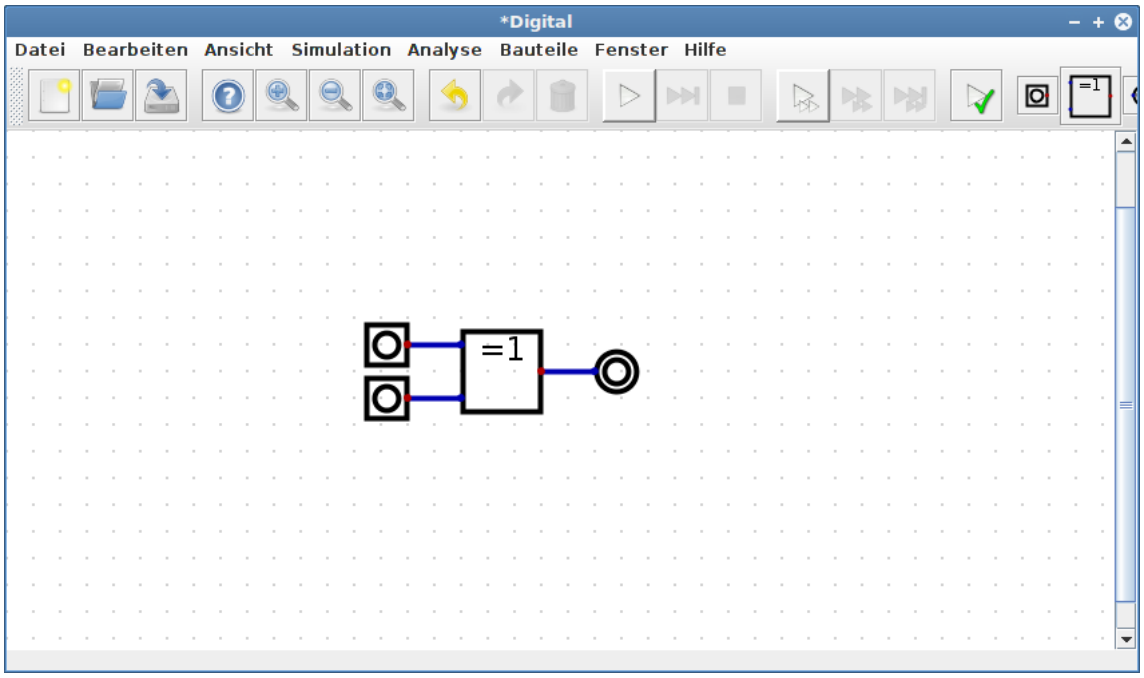
Dieses kann nun ebenfalls in die Schaltung eingefügt werden. Am besten setzt man es so, dass die spätere Verdrahtung möglichst einfach wird. Die blauen Punkte bezeichnen die Eingänge des Gatters.



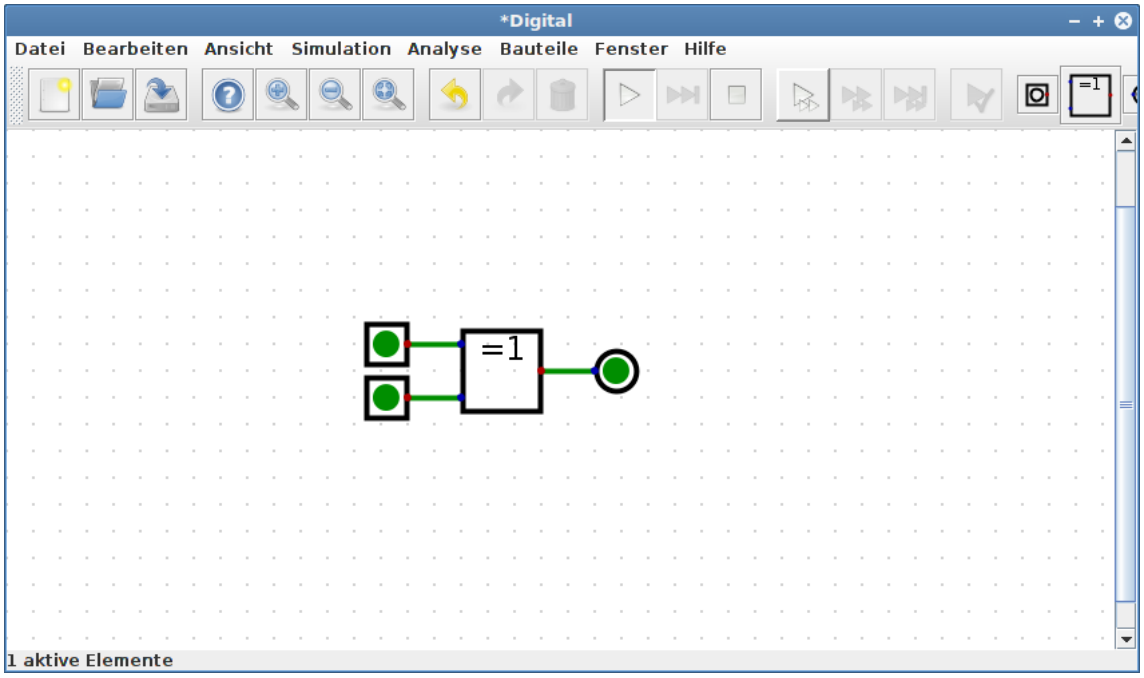
Jetzt wählt man einen Ausgang aus. Dieser kann verwendet werden, um einen Zustand anzuzeigen oder um später Signale an eine einbettende Schaltung weiterzugeben.



Auch der Ausgang wird so positioniert, dass die Verdrahtung möglichst einfach wird. Er hat einen blauen Punkt, verfügt also über einen Eingang.

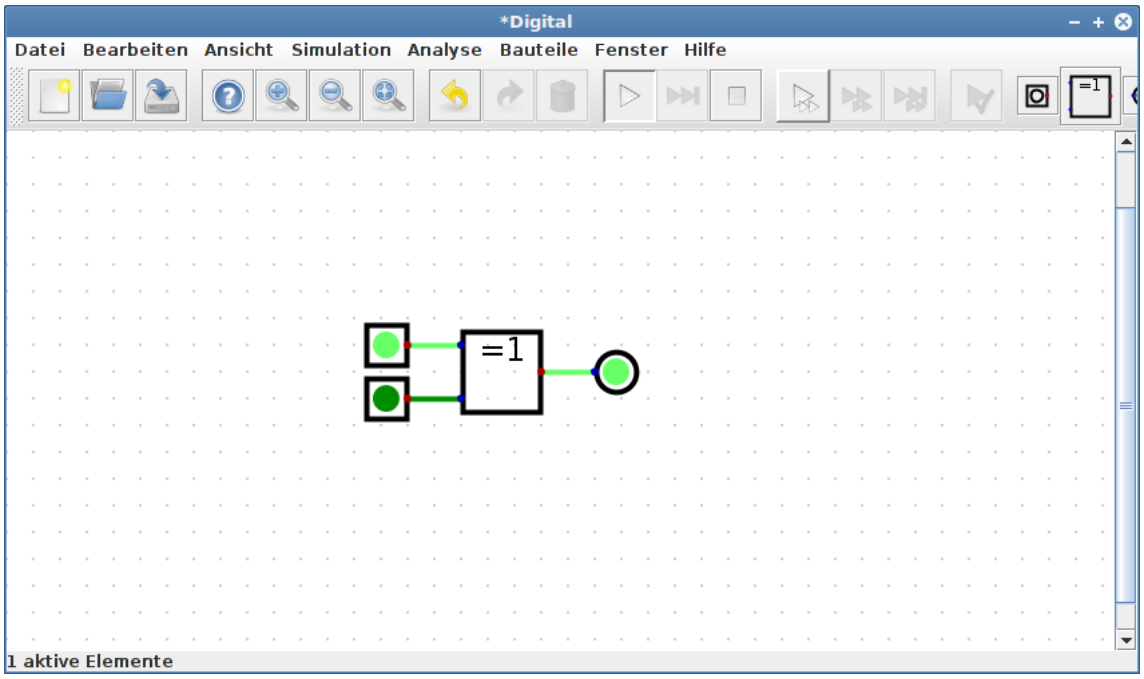


Damit sind alle erforderlichen Komponenten vorhanden. Jetzt ist die Verdrahtung zu ergänzen. Mit der Maus kann direkt eine Verbindungsleitung gezogen werden. Die blauen und roten Punkte sind miteinander zu verbinden. Es muss dabei immer genau ein roter Punkt mit einer beliebigen Anzahl blauer Punkte verbunden werden. Nur der Einsatz von Threestate-Ausgängen erlaubt es, von dieser Regel abzuweichen und mehrere rote Punkte miteinander zu verbinden. Verbindungsleitungen werden immer senkrecht oder waagerecht gezogen. Mit der Taste 'D' kann in den Diagonalmodus gewechselt werden. Sind alle noch fehlenden Leitungen ergänzt, ist eine erste Schaltung komplett.

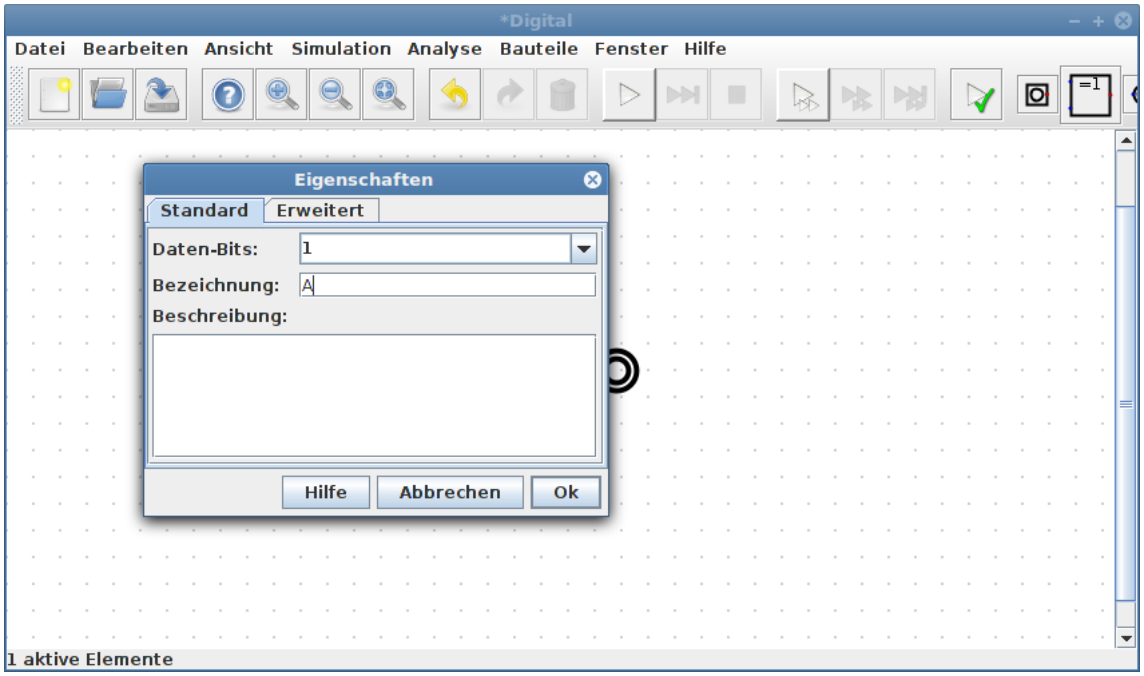


Um mit der Schaltung interagieren zu können, ist die Simulation zu starten. Dazu kann der Start-Knopf in der Toolbar verwendet werden. Nach dem Starten der Simulation ändert sich die

Farbe der Leitungen und die Ein- und Ausgänge sind jetzt ausgefüllt. Ein helles Grün markiert eine logische '1' und ein dunkles Grün eine logische '0'. In obiger Abbildung findet sich auf allen Leitungen eine logische '0'.

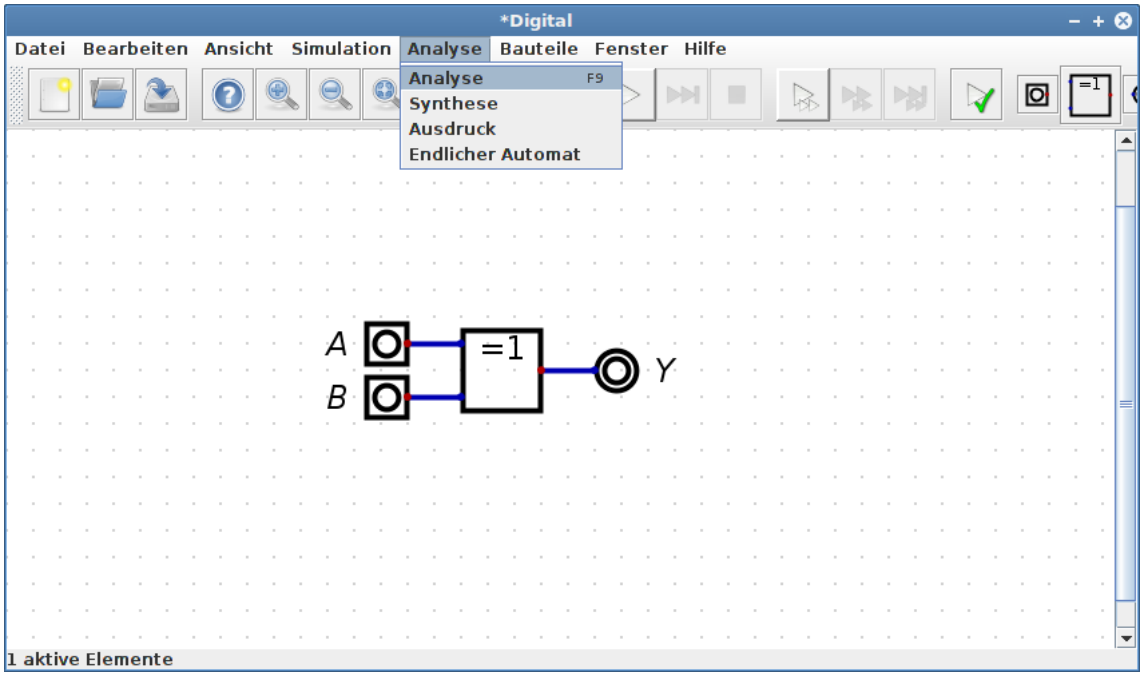


Durch Klicken mit der Maus können die Eingänge umgeschaltet werden. Da die Simulation jetzt aktiv ist, verändert sich der Ausgang entsprechend der aktuellen Eingangszustände. Die Schaltung verhält sich wie erwartet wie ein XOR-Gatter.

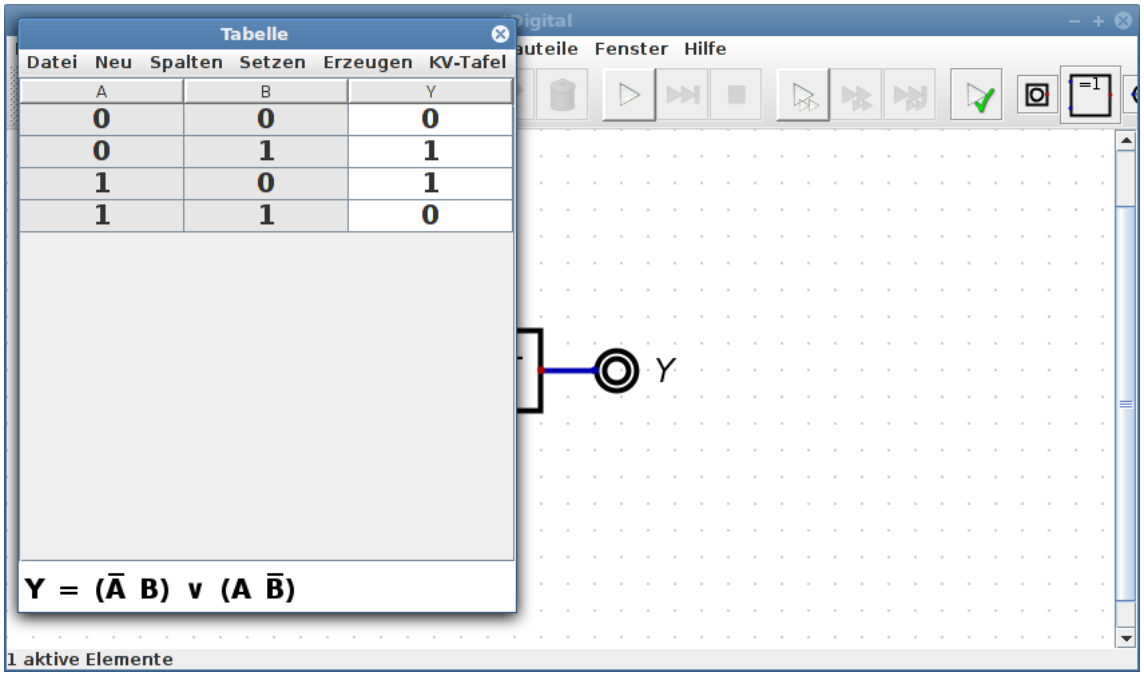


Um die Schaltung weiter bearbeiten zu können, muss die Simulation zunächst gestoppt werden. Dies geschieht am einfachsten mit dem Stopp-Knopf in der ToolBar. Durch Klicken mit der rechten Maustaste auf eine Komponente (Control-Click unter MacOS) öffnet sich ein Dia-

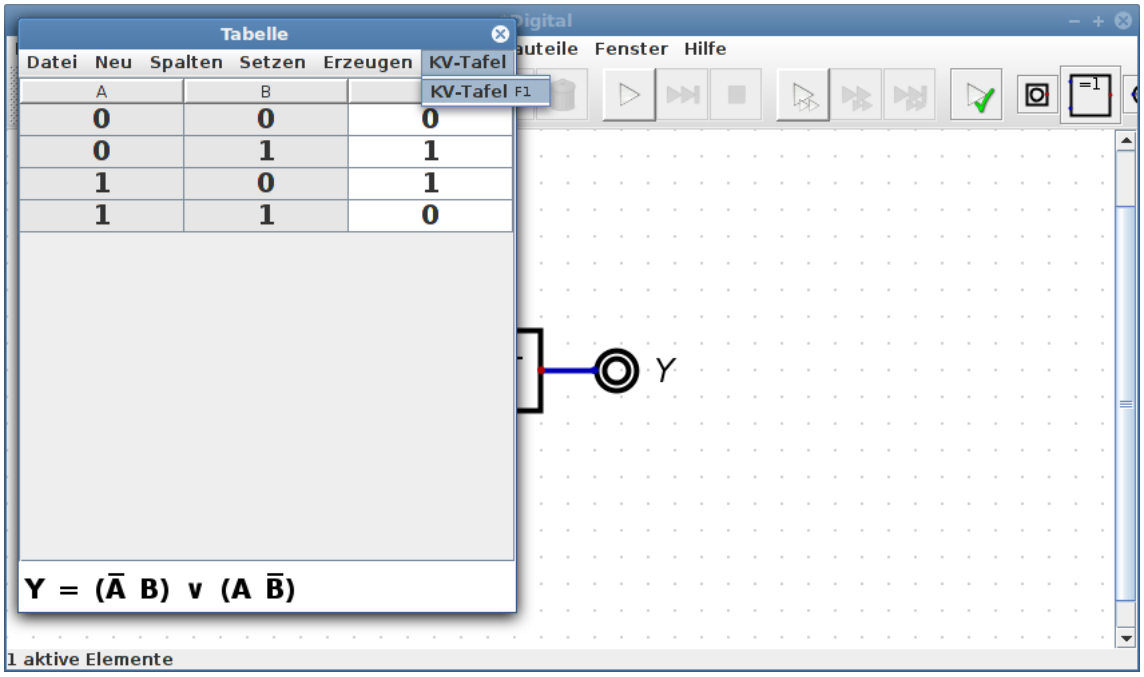
log, welcher die Eigenschaften dieser Komponente anzeigt. Für den ersten Eingang kann über diesen Dialog die Bezeichnung 'A' festgelegt werden.



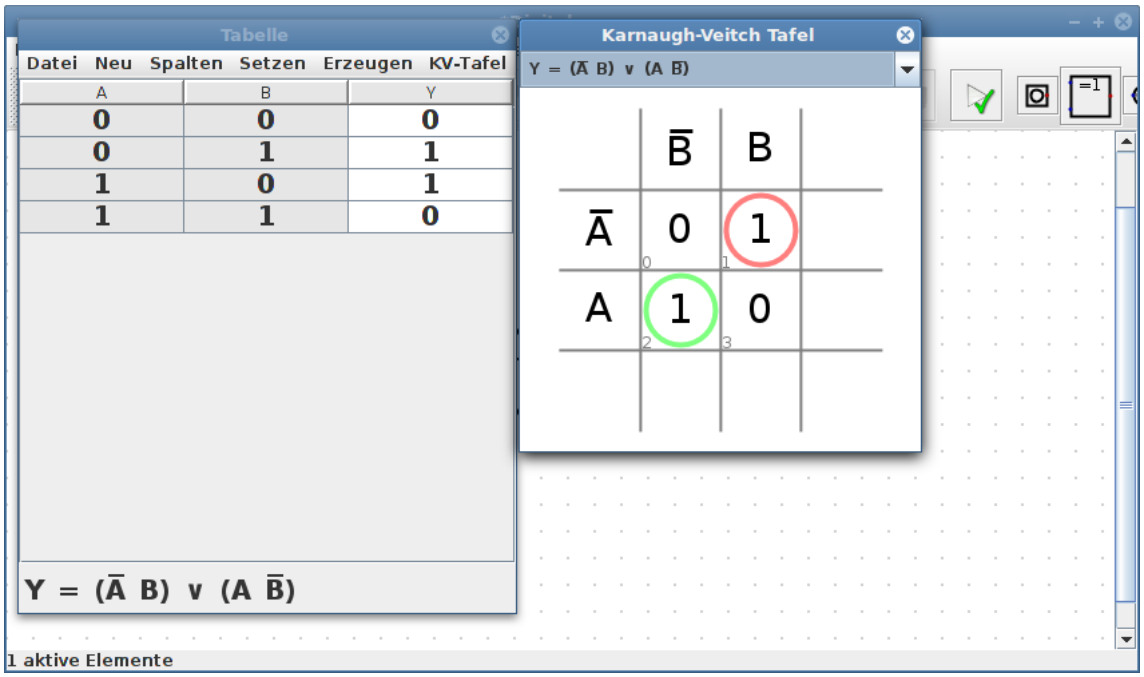
Auf diese Weise können die Bezeichnungen für die übrigen Ein- und Ausgänge definiert werden. Im Hauptmenü im Menü *Analyse* findet sich der Menüpunkt *Analyse*. Dieser ermittelt die Wahrheitstabelle der aktuellen Schaltung. Diese kann nur ermittelt werden, wenn alle Ein- und Ausgänge mit einer Bezeichnung versehen sind.



Es öffnet sich ein Fenster, welches die Wahrheitstabelle der aktuellen Schaltung zeigt. Unter der Tabelle findet sich der algebraische Ausdruck, welcher zu der Schaltung gehört. Gibt es mehrere mögliche algebraische Ausdrücke, öffnet sich ein separates Fenster, welches alle Ausdrücke anzeigt.



Der Tabellendialog hat in seinem Hauptmenü den Menüeintrag *KV-Tafel*. Über diesen lässt sich die Wahrheitstabelle in Form einer KV-Tafel anzeigen.



Oben in diesem Dialog gibt es eine Auswahlbox, über welche der gewünschte Ausdruck in der KV-Tafel ausgewählt werden kann. Auf diese Weise lässt sich z.B. verdeutlichen, wie sich mehrere äquivalente algebraische Ausdrücke ergeben können. In diesem Beispiel gibt es jedoch nur einen minimalen Ausdruck. Durch Klicken in der KV-Tafel lässt sich die Wahrheitstabelle bearbeiten.

1.3. Verbindungsleitungen

Alle Elemente müssen über Leitungen verbunden werden. Es ist nicht möglich, zwei Elemente direkt miteinander zu verbinden, indem man sie direkt nebeneinander platziert.

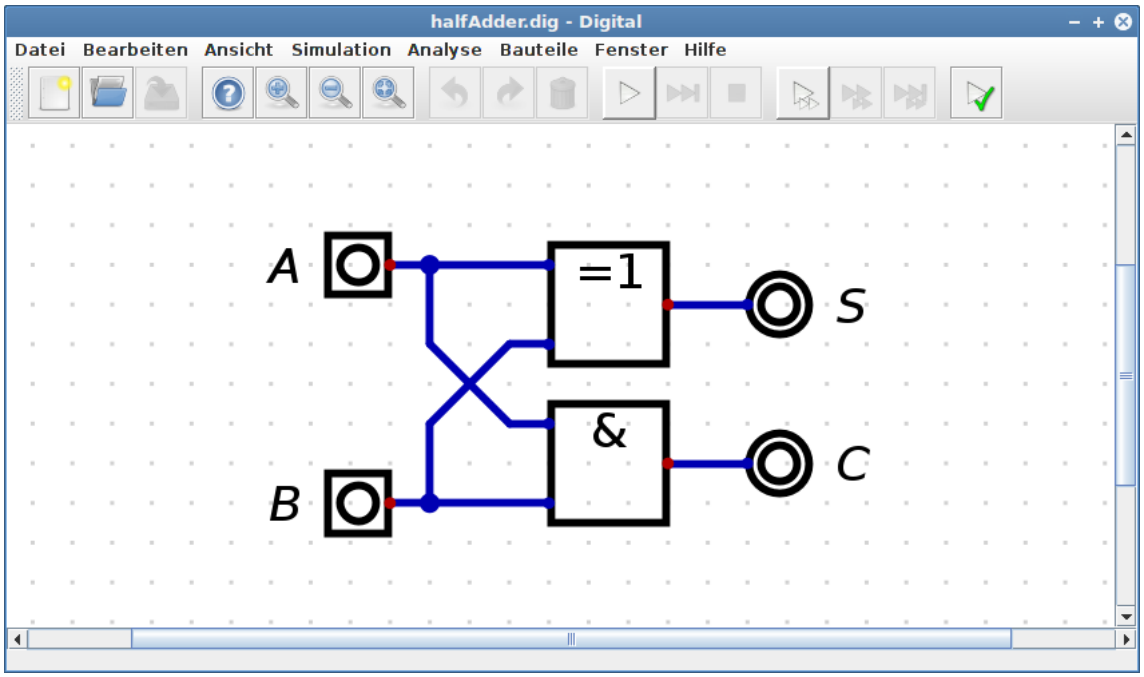
Zudem gibt es nur Verbindungen zwischen einem Leitungsende und einem Bauteil. Wird ein Bauteil mitten auf eine Leitung gesetzt, wird keine Verbindung zwischen dem Bauteil und der Leitung hergestellt. Daher muss an jedem Bauteileanschluss, welcher mit einem anderen verbunden werden soll, eine Leitung tatsächlich enden bzw. beginnen. Selbst wenn das Tunnel-Element verwendet wird, muss es eine Leitung zwischen dem Pin und dem Tunnel-Element geben.

Soll ein Element incl. der angeschlossenen Verbindungsleitungen verschoben werden, ist das Element mit der Rechteckauswahl auszuwählen. Danach lässt es sich mit der Maus verschieben. Wenn ein Element per Mausklick selektiert wird, wird nur das Bauteil allein - also ohne die angeschlossenen Leitungen mitzunehmen - verschoben.

Mit STRG-Click kann ein einzelner Leitungsabschnitt selektiert werden, um diesen zu verschieben oder zu löschen. Wird beim Zeichnen einer Leitung die Taste D gedrückt, kann eine diagonale Leitung gezogen werden. Die Taste S erlaubt das aufsplitten eines Leitungssegments in zwei Segmente.

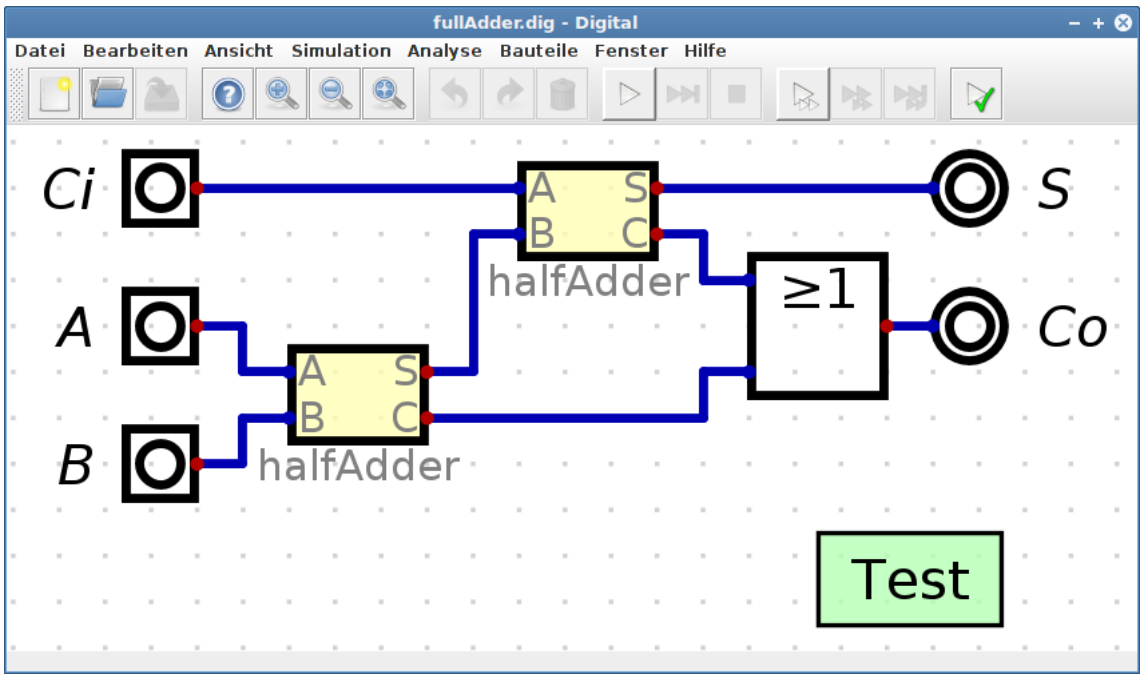
1.4. Hierarchisches Design

Wenn eine komplexe Schaltung aufgebaut wird, kann diese schnell sehr unübersichtlich werden. Um hier die Übersicht zu behalten, können die verschiedenen Teile einer Schaltung in unterschiedlichen Dateien gespeichert werden. Dieser Mechanismuss erlaubt es auch, eine einmal erstellte Teilschaltung mehrfach in einer weiteren Schaltung zu verwenden. Dieses Vorgehen bietet zudem den Vorteil, dass die Dateien unabhängig voneinander in einem Versionskontrollsystem abgelegt und Änderungen verfolgt werden können.

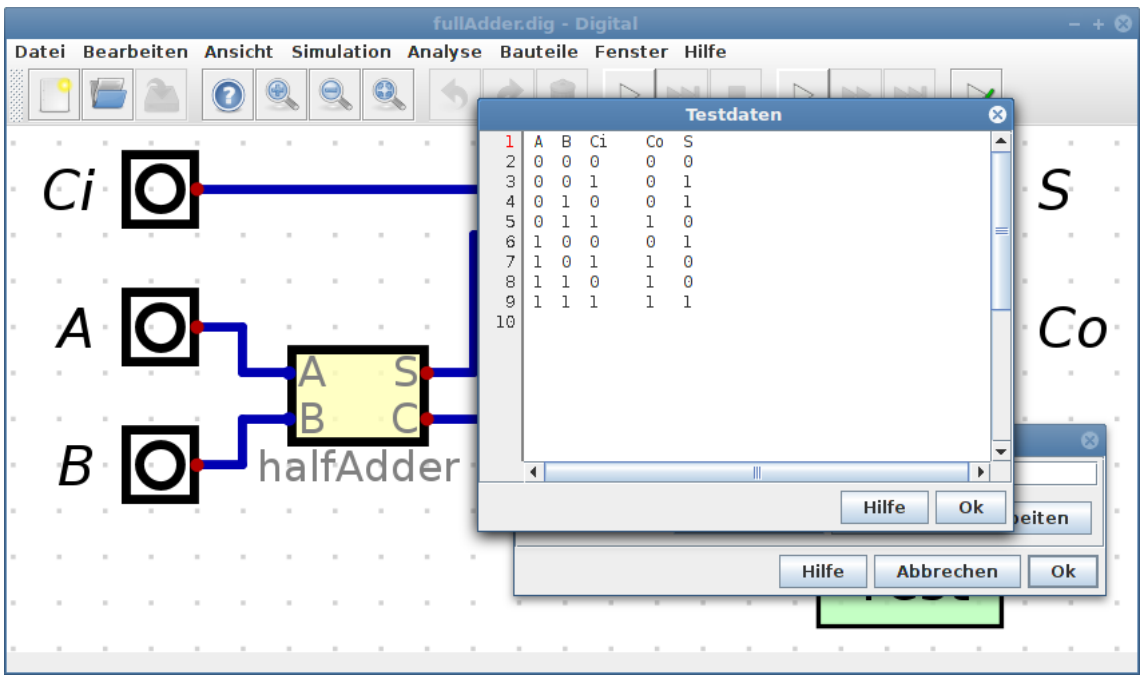


Als Beispiel sei ein 4 Bit Addierer betrachtet: Zunächst wir ein einfacher Halbaddierer aufgebaut. Dieser besteht aus einem XOR-Gatter und einem UND-Gatter. Die Summe der beiden

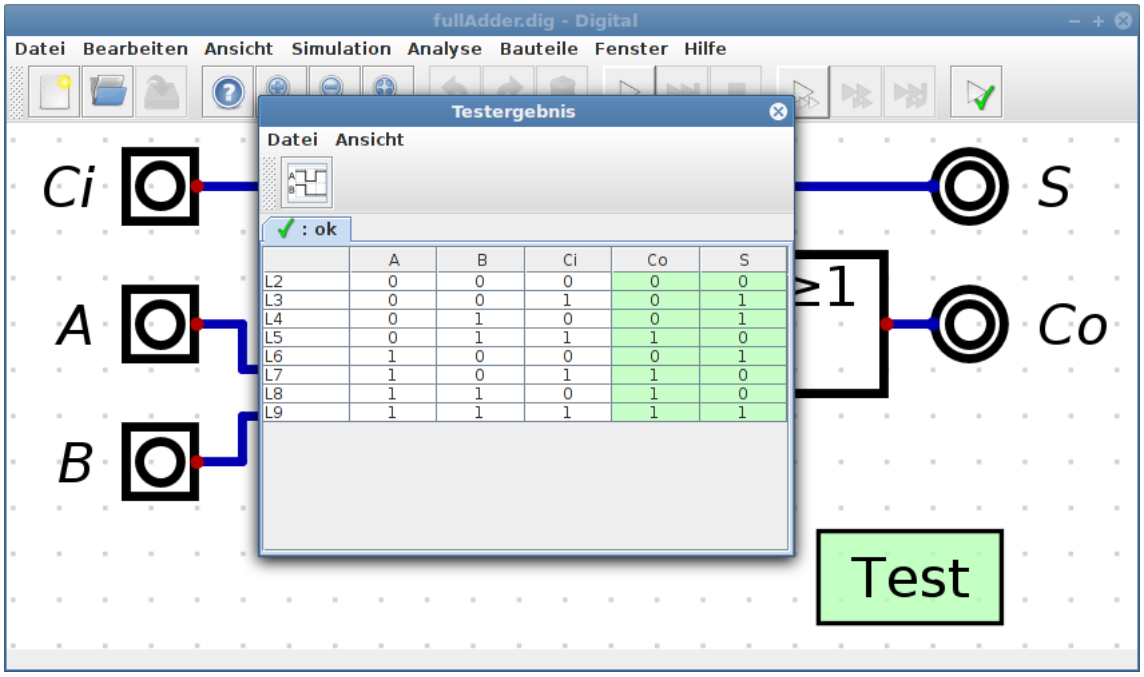
Bits 'A' und 'B' wird an den Ausgängen 'S' und 'C' ausgegeben. Diese Schaltung wird in der Datei *halfAdder.dig* gespeichert.



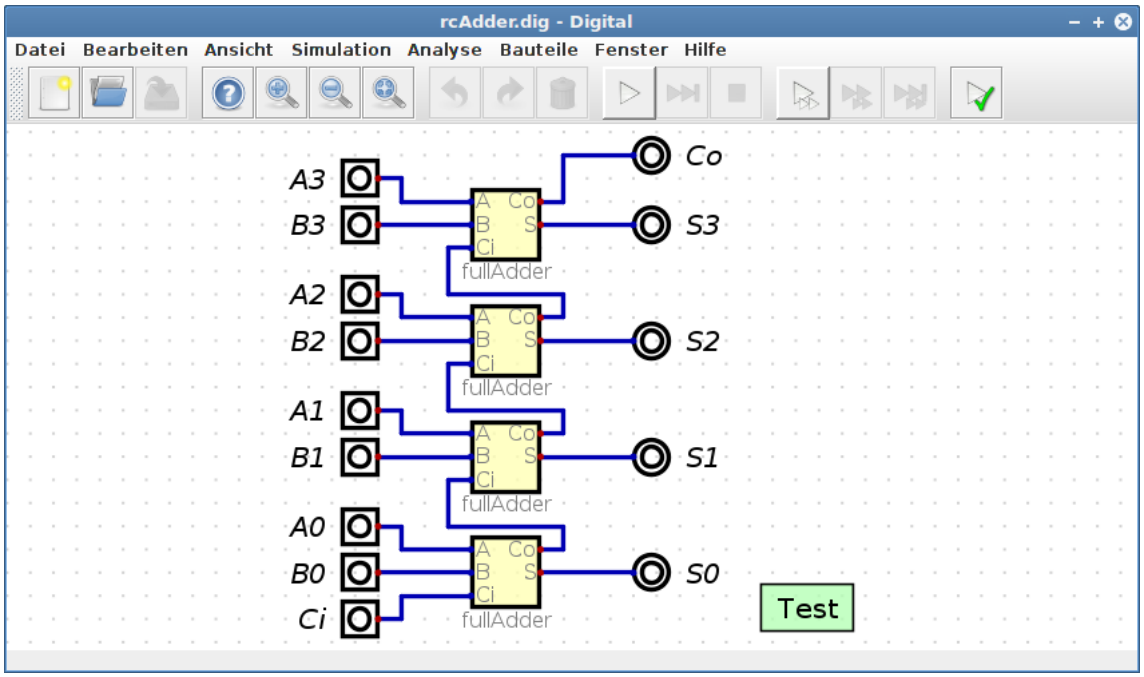
Aus zwei Halbaddierern kann nun ein Volladdierer aufgebaut werden. Dazu erzeugt man eine neue leere Datei und speichert die leere Datei unter dem Name *fullAdder.dig* im selben Ordner wie zuvor den Halbaddierer. Danach kann der Halbaddier über das Menu *Bauteile→Benutzerdefiniert* der neuen Schaltung hinzugefügt werden. Die Reihenfolge der Pins am Gehäuse des Halbaddierers kann im Halbaddierer im Menu *Bearbeiten→Sortieren der Eingänge* bzw. *Bearbeiten→Sortieren der Ausgänge* verändert werden. Der Volladdierer addiert die drei Bits 'A', 'B' und 'Ci' und gibt die Summe an 'S' und 'Co' aus.



Um die korrekte Funktion des Volladdierers zu überprüfen, sollte ein Testfall angelegt werden. Im Testfall wird die Warheitstabelle hinterlegt, welche die Schaltung erfüllen soll. Auf diese Weise kann automatisch überprüft werden, ob das tatsächlich der Fall ist.

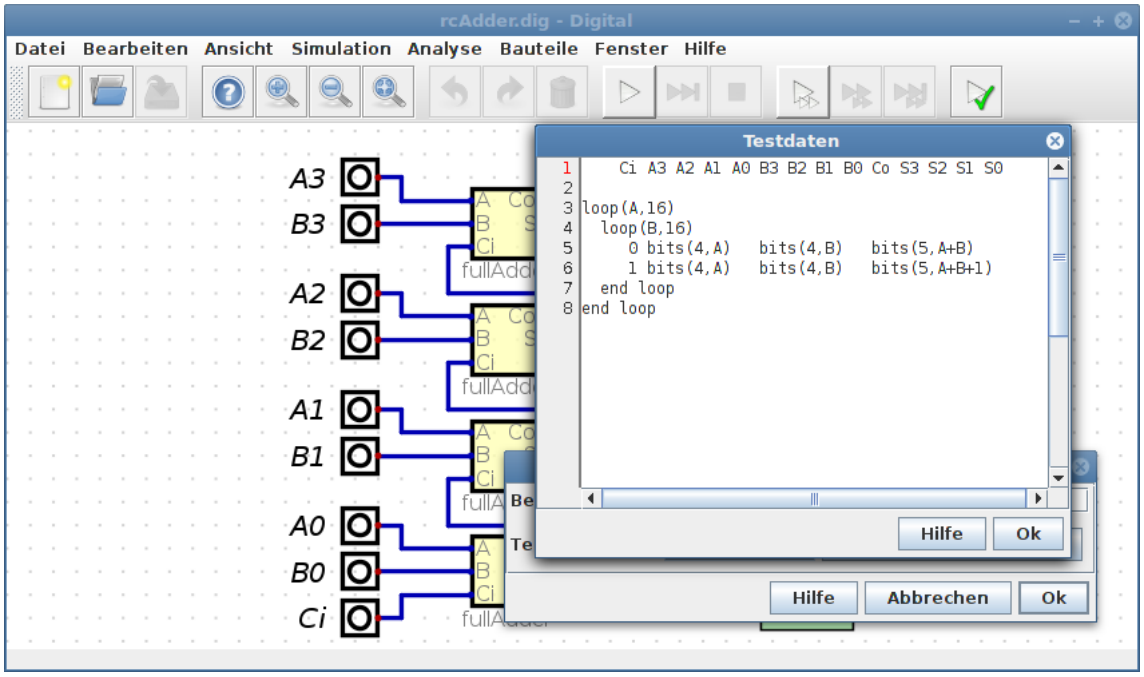


Über den Testfall-Editor oder den Testknopf in der Toolbar können die Tests ausgeführt werden. Die in der Tabelle grün hinterlegten Felder zeigen an, dass die Ausgabe der Schaltung mit der Vorgabe in der Wertetabelle des Testfalls übereinstimmt.



Nun lassen sich die Volladdierer zu einem so gen. Ripple-Carry Addierer zusammensetzen. Dabei wird der Carry-Ausgang einer Addition als Carry-Eingang an die Addition des nächst höherwertigen Bits weitergegeben, genau so, wie es bei der schriftlichen Addition üblich ist.

Dieser 4-Bit Addierer soll auf korrekte Funktion überprüft werden. Dazu wurde auch hier ein Testfall eingefügt.



Dieser Testfall führt einen 100% Test durch, was nur bei relativ einfachen Schaltungen möglich ist: Alle möglichen 512 Eingangskombinationen werden an die Schaltung angelegt, und es wird überprüft, ob die Ausgabe der Schaltung korrekt ist. In der ersten Zeile sind die Eingangs- und Ausgangssignale aufgelistet. Darunter würden in je einer Zeile die anzulegenden Eingangswerte und die zu überprüfenden Ausgangswerte angegeben, wie in einer Wahrheitstabelle. In diesem Beispiel sind jedoch 512 Zeilen erforderlich. Diese einzugeben, wäre eine mühsame und fehleranfällige Aufgabe. Einfacher und zuverlässiger ist es, die erforderlichen Zeilen automatisch erzeugen zu lassen. Dazu werden die Variablen *A* und *B* jeweils von 0 bis 15 durchlaufen. Die jeweiligen Werte von *A* und *B* werden dann den Eingängen 'A[n]' und 'B[n]' zugewiesen. Dann wird überprüft, ob die Schaltung den Wert *A+B* ausgibt. Das wird dann noch einmal mit gesetztem Carry-Bit geprüft, wobei sich in diesem Fall *A+B+1* ergeben muss. Die Details der Test-Syntax liefert der Hilfe-Dialog.

Wird eine Schaltung in eine andere eingebunden, wird in der einbindenden Schaltung nur der Dateiname gespeichert, nicht die Schaltung selbst. Die entsprechenden Dateien der eingebundenen Schaltungen müssen daher zur Laufzeit der Simulation im Dateisystem gefunden werden. Um die verschiedenen Arbeitsweisen der Nutzer bestmöglich zu unterstützen und dennoch auf eine komplexe Verwaltung von Importpfaden usw. zu verzichten, ist eine etwas ungewöhnliche Import-Strategie implementiert.

In einer Schaltung sind nur die Dateinamen der eingebetteten Schaltungen gespeichert, kein kompletter Pfad. Soll die Datei geöffnet werden, werden alle Unterordner nach einer Datei des entsprechenden Namens durchsucht. Wird eine passende Datei gefunden, wird diese importiert. Dabei kommt es nur auf den Dateinamen der einzulesenden Datei an, nicht auf deren Pfad. Entsprechend wird eine Fehlermeldung erzeugt, wenn sich in verschiedenen Unterordnern mehrere Dateien gleichen Namens befinden, da dann Mehrdeutigkeiten entstehen.

Eine geeignete Projektstruktur sieht daher wie folgt aus: In einem eigenen Ordner befindet sich die Wurzelschaltung. Alle importierten Schaltungen müssen sich im selben Ordner oder in Unterordnern befinden. Alle Schaltungen müssen unterschiedliche Namen haben, es darf

also nicht vorkommen, dass sich in verschiedenen Ordnern Schaltungen gleichen Namens befinden.

2. Simulation

2.1. Gatterlaufzeiten

Bei der Simulation wird davon ausgegangen, dass jedes Gatter eine bestimmte Laufzeit hat, und dass diese Laufzeit für alle Gatter identisch ist. Die Laufzeit eines jeden Elementes aus der Bibliothek hat genau diese Laufzeit, unabhängig von seiner Komplexität. Das Und-Gatter hat also dieselbe Signalverzögerung wie der Multiplizierer. Ausgenommen davon sind Dioden, Leitungssplitter für die Erzeugung von Datenbussen und Schalter. Diese Elemente haben keine Gatterlaufzeit bzw. diese beträgt Null.

Soll ein Gatter - z.B. der Multiplikator - mit einer längeren Laufzeit versehen werden, muss in die Schaltung ein Verzögerungsglied am Ausgang des Multiplikators eingefügt werden.

Wird eine Schaltung in eine andere Schaltung eingebettet, um ein hierarchisches Design zu erstellen, behalten die eingebetteten Schaltungen ihre Laufzeiten. Wird also ein komplexes Schaltnetz eingebettet, bei welchem ein Signal von einem Eingang bis zu einem Ausgang drei Gatter passiern muss, beträgt dessen Laufzeit auch als eingebettete Schaltung drei Gatterlaufzeiten. Es gibt keine zusätzlichen Verzögerungen durch das Einbetten einer Schaltung. Sind nicht alle Ausgangssignale einer eingebetteten Schaltung gleich schnell, werden sich die Signale in der einbettenden Schaltung ebenso verhalten. Generell verändert das Einbetten einer Schaltung in eine andere deren Laufzeitverhalten nicht. Die Schaltung verhält sich exakt so, als befänden sich alle Elemente auf derselben Ebene.

3. Analyse

3.1. Schaltungsanalyse und Synthese

Eine vorliegende Schaltung kann über den Menüeintrag *Analyse* analysiert werden. Bei rein kombinatorischen Schaltungen wird eine Wahrheitstabelle erzeugt. Diese Wahrheitstabelle kann beliebig bearbeitet werden. Aus dieser Wahrheitstabelle lässt sich nach einer Bearbeitung eine neue Schaltung erzeugen.

Neben rein kombinatorischen Schaltungen lassen sich auch Schaltwerke analysieren bzw. erzeugen. Anstelle einer einfachen Wahrheitstabelle entsteht eine sogenannte Zustandsfolgetabelle. Jedes Flipflop taucht dabei auf der Eingangsseite und der Ausgangsseite der Zustandsfolgetabelle auf. In dieser Tabelle findet sich auf der rechten Seite der abhängigen Signale der Folgezustand, der nach dem nächsten Taktsignal eintreten wird, abhängig vom aktuellen Zustand der Flipflops. Damit eine Analyse möglich ist, müssen die Flipflops benannt werden.

Dabei gilt folgende Namenskonvention: Der Folgezustand eines Bits auf der rechten Seite der Tabelle wird durch ein angehängtes kleines 'n+1' gekennzeichnet. Der dazu gehörige aktuelle Zustand wird durch ein angehängtes 'n' ausgezeichnet. Gibt es also eine Zustandsvariable 'A', dann kennzeichnet 'An' den aktuellen Zustand und 'An+1' den Folgezustand. Finden sich in der Wahrheitstabelle auf der linken und rechten Seite Signale, die diesem Muster entsprechen, wird davon ausgegangen, dass es sich um eine Zustandsfolgetabelle handelt und bei der Synthese wird entsprechend ein Schaltwerk anstelle eines Schaltnetzes erzeugt.

Zu beachten ist dabei, dass die zu analysierende Schaltung neben den eingebauten D- und JK-Flipflops nur rein kombinatorische Elemente enthalten darf. Wird ein Flipflop z.B. aus NOR-

Gattern aufgebaut, kann diese Schaltung nicht als Flipflop erkannt und daher auch nicht entsprechend analysiert werden.

3.2. Ausdrücke

Über den Menüeintrag *Ausdruck* ist es möglich, eine Boolsche Funktion einzugeben und diese dann in eine Schaltung überführen zu lassen.

3.3. Zustandsdiagramme

Über den Menüeintrag *Endlicher Automat* ist ein Editor für Zustandsdiagramme verfügbar. Dieser elaubt die grafische Erstellung von Zustandsautomaten indem Zustände und Zustandsübergänge gezeichnet werden. Dabei können in den verschiedenen Zuständen unterschiedliche Ausgänge gesetzt werden. Indem Übergänge mit Bedingungen versehen werden, können Eingangssignale erzeugt werden. Da Ausgangswerte auch an Übergängen gesetzt werden können, is es möglich, auch Mealy-Automaten zu erstellen.

Der auf diese Weise definierte Automat kann dann automatisiert in eine Zustandsübergangstabelle überführt werden, aus welcher dann in einem weiteren Schritt eine Schaltung erzeugt werden kann, welche den ursprünglichen Automaten implementiert. Wird dann die Simulation dieser Schaltung gestartet, lässt sich der aktuelle Zustand auch im noch geöffneten Zustandsdiagramm verfolgen.

4. Hardware

4.1. GAL16v8 bzw. GAL22v10

Im Menü der Schaltungserzeugung der Warheitstabelle finden sich auch Funktionen um so gen. JEDEC Dateien zu erzeugen. Dabei handelt es sich um ein spezielles Dateiformat, welches die Fuse-Map eines PLD beschreibt. Diese JEDEC Datei kann mit Hilfe eines speziellen Programmers in ein entsprechendes PLD geschrieben werden, um dieses zu konfigurieren. Zur Zeit werden Bausteine des Typs *GAL16v8* und *GAL22v10* bzw. Fuse-Map-Kompatible Bausteine unterstützt.

4.2. ATF150xAS

Die Bausteine der *ATF150x*-Familie sind einfache CPLDs mit bis zu 128 Makrozellen. Diese sind im PLCC Gehäuse verfügbar, was sie für Laborversuche gut geeignet erscheinen lässt: Sollte ein IC bei Versuchen zerstört werden, kann es einfach ersetzt werden. Zudem ist mit dem *ATDH1150USB* ein einfach zu benutzender, günstiger Programmer erhältlich, mit welchem der Baustein in seiner Schaltung (In System) programmiert werden kann. Mit dem *ATF15XX-DK3-U* ist zudem ein geeignetes Evaluations-Board verfügbar. Für die Programmierung ist die Software *ATMISP* erforderlich, welche auf der ATMEL/Microchip Webseite erhältlich ist.

Leider sind die Details der Fuse-Map nicht öffentlich zugänglich, so dass kein geeigneter Fitter für diesen Baustein in Digital integriert werden kann, wie das bei den *GAL16v8* und *GAL22v10* Bausteinen möglich ist.

Daher muss auf einen der von ATMEL bereitgestellten Fitter *fit150[x].exe* zurückgegriffen werden. Diese Programme erzeugen aus einer geeigneten *TT2* Datei eine *JEDEC*-Datei, welche dann in den entsprechenden Chip übertragen werden kann. Digital startet den Fitter automatisch, wenn eine *TT2* Datei erzeugt wird. Dazu muss in den Digital-Einstellungen der Pfad zu

den *fit150[x].exe* Dateien angegeben werden. Es wird zudem eine Projektdatei **.chn* erzeugt, welche dann direkt mit *ATMISP* geöffnet werden kann, um die *JEDEC*-Datei zu brennen.

Aus rechtlichen Gründen können die Fitter *fit150[n].exe* nicht mit Digital ausgeliefert werden. Diese finden sich jedoch nach der Installation von *WinCupl* im Ordner *WinCupl\Fitters*. *WinCupl* wiederum ist auf der ATMEL/Microchip Webseite verfügbar. Unter Linux lassen sich die Fitter ebenfalls von Digital starten, wenn *wine* installiert ist.

4.3. Export zu VHDL oder Verilog

Eine Schaltung kann zu VHDL oder Verilog exportiert werden. Dabei wird eine Datei erzeugt, welche die komplette Beschreibung der Schaltung enthält. Der erzeugte VHDL Code wurde mit Xilinx Vivado und dem Open Source VHDL Simulator ghdl getestet. Der Verilog Code mit dem Verilog Simulator Icarus Verilog.

Wenn eine Schaltung Testfälle enthält, wird anhand der Testdaten eine Test-Bench erzeugt. Diese kann verwendet werden, um die korrekte Funktion der Schaltung in einer HDL-Simulation zu überprüfen.

Für spezielle Boards können zusätzliche Dateien erzeugt werden. Zur Zeit werden nur das BASYS3 Board und die Boards Mimas und Mimas V2 unterstützt. Dabei wird eine Constraints-Datei erzeugt, welche die Zuordnung der Pins beinhaltet. Die Bezeichnung der Pins kann den entsprechenden Datenblättern entnommen werden und ist als Pinnummer bei den Ein- und Ausgängen einzutragen.

Beim BASYS3 Board wird, wenn die Taktfrequenz niedrig ist, ein Frequenzteiler in den HDL Code integriert, um den Boardtakt entsprechend zu teilen. Wenn die in der Schaltung gewählte Taktfrequenz über 4.7MHz liegt, wird die MMCM Einheit des Artix-7 zur Takterzeugung verwendet. Dies stellt sicher, dass die für die Taktverteilung vorgesehenen FPGA-Resourcen auch tatsächlich verwendet werden. Der enthaltene Beispiel-Prozessor läuft z.B. mit 20MHz, und wenn auf den Multiplizierer in der ALU verzichtet werden kann, sind auch 30MHz möglich.

Soll eine Schaltung auf einem BASYS3 Board betrieben werden, wird eine Vivado Projektdatei angelegt, die direkt mit Vivado geöffnet werden kann. Es lässt sich dann der Bitstream erzeugen und mit dem Hardware-Manager kann dieser in ein BASYS3 Board übertragen werden.

Um neben der HDL Datei auch die erforderliche Constraints-Datei erzeugen zu lassen, muss in den Einstellungen das Entsprechende Board konfiguriert werden. Dazu kann im Feld "Toolchain Konfiguration" die entspechende XML-Datei ausgewählt werden. Die verfügbaren Konfgurationen finden sich im Ordner *examples/hdl* und haben die Dateiendung *.config*. Wurde die Konfiguration erfolgreich eingebunden, erscheint ein weiteres Menü, welches die Board-Spezifischen Funktionen zugänglich macht.

5. Benutzerdefinierter Schaltungssymbole

Digital bietet zwar einige Optionen, die die Erscheinungsform einer Schaltung festlegen, wenn diese in eine andere eingebettet wird, in manchen Fällen kann es jedoch sinnvoll sein, ein spezielles eigenes Symbol für eine Teilschaltung zu verwenden. Ein Beispiel ist die Darstellung der ALU in dem Prozessor, welcher in den Beispielen enthalten ist. In diesem Kapitel wird beschrieben, wie eine solches spezielles Symbol für eine Schaltung definiert wird.

Digital bietet keinen Editor für das Erstellen eines speziellen Symbols an. Statt dessen ist ein kleiner Umweg für das Erstellen von Schaltungssymbolen erforderlich: Zunächst wird die Schaltung geöffnet, welche durch ein spezielles Symbol repräsentiert werden soll. Dann wird für diese Schaltung ein SVG-Template erstellt. In diesem Template wird die Schaltung durch ein einfaches Rechteck repräsentiert. Es finden sich darin auch alle Pins der Schaltung, reprä-

sentiert durch blaue (Eingänge) und rote (Ausgänge) Kreise. Um zu sehen, welcher Kreis zu welchem Pin gehört, kann man sich in den Objekteigenschaften die ID des Kreises ansehen. Diese ID hat die Form *pin:[Name]* oder *pin+:[Name]*. Bei der letzteren Variante wird der Pin beim Reimport zu Digital mit einem Label versehen. Wünscht man kein solches Label, kann das + entfernt werden.

Diese SVG-Datei kann jetzt bearbeitet werden. Am besten geeignet ist das Open-Source Programm Inkscape welches kostenlos verfügbar ist. Die Pins können frei verschoben werden, werden jedoch beim Reimport auf den nächstgelegenen Rasterpunkt verschoben.

Wenn schon existierende SVG Dateien verwendet werden sollen, ist es das einfachste das erstellte Template zu öffnen, und die bereits vorhandene Grafik per Copy&Paste in das Template einzufügen.

Wenn die Datei gespeichert wurde, kann diese mit Digital importiert werden. Dabei wird die Datei eingelesen und alle notwendigen Informationen werden extrahiert und in der Schaltung gespeichert. Für die weitere Verwendung der Schaltung ist die SVG-Datei also nicht mehr erforderlich.

Noch eine Anmerkung: SVG ist ein sehr mächtiges und flexibles Dateiformat. Es können extrem komplexe Grafiken damit beschrieben werden. Der in Digital integrierte Importer ist nicht in der Lage alle denkbaren SVG-Dateien fehlerfrei zu importieren. Wenn eine Datei nicht importiert werden kann, oder nicht so erscheint wie erwartet, ist evtl. einiges Experimentieren erforderlich, bevor das gewünschte Ergebnis erreicht ist.

6. Generische Schaltungen

Es kommt vor, dass eine Teilschaltung erstellt wurde, und diese soll in veschiedenen Varianten verwendet werden. Vorstellbar ist z.B. ein spezieller Zähler, der für verschiedene Bitbreiten benötigt wird. Würde man nun je eine Teilschaltung für 4,5 und 6 Bits erstellen, wäre die Pflege der Schaltung in Zukunft schwierig, da bei jeder Änderung mehrere Teilschaltungen zu bearbeiten sind, die bis auf einen Parameter - die Bitbreite - identisch sind.

Um dies zu verhindern, kann eine generische Teilschaltung erstellt werden, die sich parametrisieren lässt. Dazu muss in den Schaltungseinstellung der Schaltung das Häckchen "Schaltung ist generisch" gesetzt werden. Danach enthält der Parameter-Dialog jeder Komponente der Schaltung das zusätzliche Feld "generische Parametrisierung". In diesem Feld kann Programmcode angegeben werden, welcher die Parameter der Komponente verändern kann. Jeder Parameter hat einen Namen und kann als Attribut des Feldes *this* modifiziert werden. Wie die Parameter benannt sind, kann dem Hilfedialog der Komponente entnommen werden. Möchte man die Bitbreite eines Addierers ändern, kann die Zeile *this.Bits=1;* verwendet werden.

Auf diese Weise kann man jedoch noch keine parametrisierbare Schaltung erstellen. Es ist noch erforderlich auf Parameter zuzugreifen, die gesetzt werden, wenn die Schaltung verwendet wird. Dies geschieht über das Feld "args". Will man die Bitbreite von außen setzen, kann man schreiben: *this.Bits=args.bitWidth;*. Der Name des Argumentes - hier *bitWidth* - ist dabei beliebig. Wenn diese Teilschaltung verwendet wird, ist dieses Argument zu setzen.

Wird die Schaltung verwendet, und der Parameter-Dialog der eingebetteten Schaltung geöffnet, hat auch dieser ein Feld "generische Parametrisierung". Hier kann mit der Anweisung *bitWidth:=5;* die zu verwendende Bitbreite gesetzt werden.

Wenn eine generische Schaltung direkt gestartet werden soll, ist das ohne Weiteres nicht möglich, da die erforderlichen Argumente fehlen, die bei der Einbettung der Schaltung angegeben werden müssen. Diese fehlenden Argumente würden zu entsprechenden Fehlermeldungen

führen. Um den Test der Schaltung zu vereinfachen, kann daher die Komponente *Generische Initialisierung* der Schaltung hinzugefügt werden. In dieser Komponente lassen sich die Argumente setzen, die aus einer einbettenden Schaltung kommen würden. Auf diese Weise kann auch eine generische Schaltung direkt simuliert werden, was die Erstellung sehr erleichtert. Wird die Schaltung eingebettet, wird diese Komponente ignoriert. Sie wird nur für den direkten Start der Simulation benötigt.

Unter Umständen kann es sinnvoll sein, nicht nur die Attribute der Komponenten einer Schaltung zu verändern, sondern ganz neue Komponenten und Leitungen abhängig von den übergebenen Argumenten hinzuzufügen. Dazu kann die Komponente *Code* verwendet werden. Wird sie der Schaltung hinzugefügt, wird der enhaltene Code beim Start der Simulation ausgeführt. Hier kann mit der Funktion *addWire([x1],[y1],[x2],[y2])* eine Leitung hinzugefügt werden und über die Funktionen *addComponent([Name],[x],[y])* kann eine neue Komponente *[Name]* an der Position ([x],[y]) hinzugefügt werden. Der Rückgabewert der Funktion *addComponent([Name],[x],[y])* erlaubt das Setzen der Parameter der Komponente.

Die Beispielschaltung *examples/generic/modify/Conway/GenericConway.dig* zeigt, wie auf diese Weise auch eine komplexere Schaltung zusammengesetzt werden kann.

Eine andere Möglichkeit eine Schaltung zu erzeugen ist Rekursion: Es ist möglich, abhängig von den Argumenten, eine Schaltung durch eine andere zu ersetzen. Zu diesem Zweck steht die Funktion *setCircuit([Name])* zur Verfügung. Wird sie im Definitionsteil einer eingebetteten Schaltung aufgerufen, kann die einzufügende Schaltung durch eine andere ersetzt werden. Dies erlaubt die rekursive Definition einer Schaltung. Wie in anderen Programmiersprachen auch, ist auf eine geeignete Abbruchbedingung zu achten.

Im Ordner *examples/generic* findet sich ein Beispiel für einen Medwedew Gray-Code-Zähler, dessen Bitbreite konfigurierbar ist. Hier wird ein Gray-Code-Zähler aufgebaut, indem einer initialen Schaltung rekursiv weitere Bits hinzugefügt werden, bis die erforderliche Bitzahl des Zählers erreicht ist.

7. Skriptgesteuertes Testen

Wenn Studenten Aufgaben mit Digital erledigen sollen, kann es hilfreich sein, die von den Studenten abgegebenen Schaltungen in einem automatischen Prozess überprüfen zu können. Um diese Überprüfung durchzuführen, kann Digital über die Kommandozeile gestartet werden. Der Aufruf geschieht dabei wie folgt:

```
java -cp Digital.jar CLI test [zu testende Datei] [-tests [optionale Datei mit Testfällen]]
```

Wird nur die zu testende Datei angegeben, werden die Testfälle in dieser Datei ausgeführt. Auf diese Weise können die Testfälle ausgeführt werden, welche die Studenten selbst erstellt haben.

Wird ein zweiter Dateiname angegeben, werden die Testfälle aus der zweiten Datei entnommen und die erste Schaltung wird mit diesen Testfällen überprüft. Die zweite Datei wird also in der Regel die Musterlösung enthalten, deren Testfälle vollständig und korrekt sind. Die in der zweiten Datei enthaltene Schaltung wird dabei ignoriert. Nur die Testfälle werden daraus entnommen.

Um eine abgegebene Schaltung gegen eine Musterlösung testen zu können, müssen die Signalnamen der Ein- und Ausgänge in beiden Schaltungen übereinstimmen.

8. Häufig gestellte Fragen

Wie kann ich eine Leitung verschieben?

Mit der Rechteckauswahl einen Endpunkt auswählen und dann mit der Maus verschieben. Alternativ kann eine einzelne Leitung mit STRG+Mausklick selektiert werden.

Wie kann ich eine Leitung löschen?

Mit der Rechteckauswahl einen Endpunkt auswählen und dann *Entfernen* drücken bzw. auf den Papierkorb klicken. Alternativ kann eine einzelne Leitung mit STRG+Mausklick selektiert werden.

Wie kann ich ein Element verschieben und dabei alle angeschlossenen Leitungen mitnehmen?

Das Bauteil mit der Rechteckauswahl auswählen. Die Auswahl muss das Bauteil komplett enthalten. Dann kann das Bauteil zusammen mit den Leitungen verschoben werden.

Ich habe ein Element, das nicht mit einer Leitung verbunden ist, obwohl die Pins auf der Leitung liegen.

Pins werden nur mit einer Leitung verbunden, wenn ein Endpunkt der Leitung auf dem Pin liegt.

Wenn die Pinnamen einer Schaltung etwas länger sind, sind die Namen nicht mehr zu lesen, wenn die Schaltung eingebettet wird. Was kann ich tun?

Unter *Bearbeiten→Schaltungsattribute bearbeiten* kann die Breite des Blockes vergrößert werden.

Die Pins in einer eingebetteten Schaltung haben eine ungünstige Reihenfolge. Wie kann diese verändert werden?

Unter *Bearbeiten→Sortieren der Eingänge* bzw. *Bearbeiten→Sortieren der Ausgänge* kann die Reihenfolge verändert werden.

Wenn die Simulation gestartet wird, ist die Leitung grau. Was bedeutet das?

Mit Hellgrün und Dunkelgrün wird eine logische 1 bzw. 0 dargestellt. Grau bedeutet, dass die Leitung hochohmig ist.

Ich habe eine Wahrheitstabelle. Wie kann ich daraus die minimalen Schaltfunktionen erzeugen?

Im Menü *Analyse→Synthese* auswählen und dann die Wahrheitstabelle eingeben. In der Statusleiste des Fensters findet sich die Schaltfunktion. Geben Sie mehr als eine abhängige Größe an, öffnet sich ein neues Fenster, in welchem alle Schaltfunktionen angezeigt werden.

Ich habe eine Wahrheitstabelle eingegeben, es wird jedoch mehr als eine Schaltfunktion für das gesuchte Signal angezeigt. Welche ist richtig?

Wenn eine Schaltfunktion minimiert wird, kann es mehrere minimierte Schaltfunktionen gleicher Komplexität geben (gleiche Anzahl von Termen). Digital zeigt alle möglichen Lösungen an und alle liefern die selbe Wahrheitstabelle. In den Don't Care-Zeilen der Wahrheitstabelle kann es natürlich Unterschiede zwischen den verschiedenen Funktionen geben.

Ich habe eine Wahrheitstabelle. Wie kann ich daraus eine Schaltung erzeugen?

Im Menü *Analyse→Synthese* auswählen und dann die Wahrheitstabelle eingeben. Unter *Neu* bzw. mit *Spalten hinzufügen* können Variablen hinzugefügt werden. Im Menü mit *Erzeugen→Schaltung* dann eine Schaltung erzeugen.

Wie kann ich die Signalnamen in der Wahrheitstabelle verändern?

Mit einem Rechtsklick auf den Signalnamen im Tabellenkopf kann der Name bearbeitet werden.

Ich habe eine Schaltfunktion. Wie kann ich daraus eine Schaltung erzeugen?

Im Menü *Analyse→Ausdruck* auswählen und dann die Funktion eingeben.

Ich habe eine Schaltfunktion. Wie kann ich daraus eine Wahrheitstabelle erzeugen?

Im Menü *Analyse→Ausdruck* auswählen und dann die Funktion eingeben. Dann die Schaltung erzeugen und im Menü *Analyse→Analyse* die Wahrheitstabelle erzeugen.

Wie kann ich aus einer Schaltung eine JEDEC-Datei erzeugen?

Im Menü *Analyse→Analyse* auswählen und dann im neuen Fenster im Menü *Erzeugen→Bausteine* den entsprechenden Baustein auswählen.

Wie kann ich bei der Erzeugung einer JEDEC-Datei die Pinnummern der Signale festlegen?

Bei den entsprechenden Eingängen und Ausgängen in der Schaltung kann im Attribute-Dialog eine Pinnummer zugewiesen werden.

Ich habe eine JEDEC Datei erzeugt. Wie bekomme ich diese in ein *GAL16v8* bzw. *GAL22v10*?

Für die Programmierung von GAL's ist eine entsprechende Zusatz-Hardware, ein so gen. GAL-Programmer erforderlich.

Ich habe eine Schaltung erstellt, welche ich in vielen anderen Schaltungen verwenden möchte. Wie kann ich das ermöglichen, ohne die Datei immer wieder in die entsprechenden Ordner zu kopieren?

Die Schaltung kann im "lib" Ordner gespeichert werden. Dann ist sie in allen anderen Schaltungen verfügbar.

9. Tastenbelegung

Leertaste	Starten und stoppen der Simulation
F6	Anzeigen der Messwerttabelle
F7	Run to Break
F8	Starten der Testfälle
C	Einen Taktschritt ausführen (nur bei gestarteter Simulation und nur, wenn es genau ein Taktelement in der Schaltung gibt)
V	Einen Gatterschritt ausführen.
B	Alle Gatterschritte ausführen, bis sich die Schaltung stabilisiert hat, oder, wenn ein Break-Element vorhanden ist, bis zum Break.
F9	Analyse der Schaltung
STRG-A	Alles auswählen
STRG-X	Ausschneiden der selektierten Elemente und kopieren in die Zwischenablage
STRG-C	Kopieren der selektierten Elemente in die Zwischenablage
STRG-V	Einfügen der Elemente aus der Zwischenablage.
STRG-D	Aktuelle Auswahl duplizieren ohne die Zwischenablage zu verändern.
R	Rotieren der Elemente beim Einfügen
L	Zuletzt eingefügtes Element noch einmal einfügen
T	Einfügen eines Tunnels
STRG-N	Neue Schaltung
STRG-O	Schaltung öffnen
STRG-S	Speichern der Schaltung
STRG-Z	Letzte Änderung zurücknehmen
STRG-Y	Zurückgenommene Änderung erneut anwenden
P	Programmieren einer Diode oder eines FG-FET
D	Beim Ziehen einer rechteckigen Leitung in den Diagonalmodus wechseln
F	Beim Ziehen einer rechteckigen Leitung die Orientierung wechseln
S	Leitung in zwei Leitungen teilen
ESC	Abbrechen der aktuellen Aktion
Entfernen	Löschen der selektierten Elemente
Rückschritt	Löschen der selektierten Elemente
+	Erhöht die Anzahl der Eingänge in dem Element, auf welches die Maus zeigt. Bei Konstanten wird der Wert erhöht.

-	Erniedrigt die Anzahl der Eingänge in dem Element, auf welches die Maus zeigt. Bei Konstanten wird der Wert verringert.
STRG +	Vergrößern
STRG -	Verkleinern
F1	Einpassen
F5	Baumansicht der Bauteile ein- oder ausblenden

B Einstellungen

Im Folgenden werden die vorhandenen Einstellungen des Simulators beschrieben.

Einstellungen

In den globalen Einstellungen des Simulators werden u.a. die Sprache, die für die elementaren Gatter zu verwendende Symbolform oder die Pfade externer Tools vorgegeben.

Veränderbare Attribute

Verwende US Symbole (IEEE 91-1984)
: Verwende IEEE 91-1984 Symbole anstelle der rechteckigen Symbole

Sprache
: Sprache der Oberfläche. Wird erst nach einem Neustart wirksam.

Format
: Anzeigeformat der Ausdrücke.

Farbschema
: Farbschema

Benutzerdefinierte Farben
: Benutzerdefinierte Farben

Baumansicht beim Start anzeigen
: Wenn gesetzt, wird die Baumansicht beim Start automatisch angezeigt.

Raster anzeigen
: Zeigt im Hauptfenster ein Raster an, um das platzieren der Elemente zu erleichtern.

Zeigt die Anzahl der Leitungen auf einem Bus
: Zeigt die Anzahl der Leitungen eines Bus an. ACHTUNG: Dieser Wert wird nur beim Start der Simulation aktualisiert.

Keine ToolTips für Bauteile auf der Arbeitsfläche.
: Wenn gesetzt, werden keine ToolTips für die Bauteile auf der Arbeitsfläche angezeigt. Vor allem in einer Präsentation können diese ToolTips sehr störend sein.

Leitungen als ToolTip
: Wenn gesetzt, werden Leitungen hervorgehoben, wenn die Maus darauf zeigt.

Bibliothek
: Ordner, in welchem sich die Bibliothek mit vordefinierten Schaltungen befindet. Enthält z.B. die ICs der 74xx Reihe. Es können auch eigene Schaltungen hinzugefügt werden, indem diese hier gespeichert werden. Es muss darauf geachtet werden, dass der Name aller Dateien in diesem Ordner und allen Unterordnern eindeutig ist.

Java Bibliothek
: Eine jar Datei, welche neue Bauteile als Java Klassen enthält.

ATF15xx Fitter
: Pfad zum Fitter für den ATF15xx. Geben Sie hier das Verzeichnis an, welches die Dateien fit15xx.exe enthält. Diese Datei wird von Microchip (früher ATMEL) zur Verfügung gestellt.

ATMISP
: Pfad zur ausführbaren Datei ATMISP.exe. Wenn gesetzt, kann die Software ATMISP automatisch gestartet werden!

GHDL
: Pfad der ausführbaren ghdl-Datei. Nur wichtig, wenn ghdl zur Interpretation von VHDL-Code verwendet werden soll.

IVerilog
: Pfad zum Icarus-Verilog-Installationsordner. Nur notwendig, wenn Sie iverilog verwenden möchten, um mit Verilog definierte Komponenten zu simulieren.

Toolchain Konfiguration
: Kann für eine Integration einer externen Toolchain verwendet werden. Erlaubt den Start externer Tools, um z.B. einen FPGA zu programmieren o.ä.

Schriftgröße im Menü [%]
: Für die Menüs kann eine abweichende Schriftgröße gewählt werden. Angabe in Prozent der Standardgröße.

Die MacOS Mausklicks verwenden.
: Das unter MacOS übliche STRG-Klick anstelle von Rechtsklick verwenden.

Die Gleich-Taste verwenden.
: Die Gleich-Taste anstelle der Plus-Taste verwenden. Dies ist immer dann sinvoll, wenn das Plus-Zeichen keine Primärtaste ist, sondern die Zweitbelegung des Gleich-Zeichens, also z.B. bei amerikanischem oder französischem Tastaturlayout.

Dialog zum automatischen umbenennen von Tunneln anzeigen
: Wenn gesetzt, wird nach dem Umbenennen eines Tunnels ein Dialog für automatisches Umbenennen aller gleichnamigen Tunnel angezeigt.

Einstellungen der Schaltung

Die Einstellungen der Schaltung beeinflussen das Verhalten der aktuell geöffneten Schaltung. So kann z.B. das Erscheiningsbild festgelegt werden, welche die Schaltung annimmt, wenn sie in andere Schaltungen eingebettet wird. Diese Einstellungen werden mit der Schaltung zusammen gespeichert.

Veränderbare Attribute

Bezeichnung
: Die Bezeichnung dieses Elementes.

Breite
: Breite des Symbols, wenn diese Schaltung in eine andere eingefügt wird.

Hintergrundfarbe
: Hintergrundfarbe der Schaltung, wenn sie eingebettet wird. Wird für DILs nicht verwendet.

Beschreibung
: Eine kurze Beschreibung des Elementes.

Bearbeitung gesperrt
: Die Schaltung ist für die Bearbeitung gesperrt. Dioden und FG-FETs können jedoch konfiguriert werden.

Form
: Die Form, welche für die Repräsentation der Schaltung in einer einbettenden Schaltung verwendet werden soll. Bei der Form "Einfach" werden die Eingänge auf der linken und die Ausgänge auf der rechten Seite eines einfachen Rechtecks angezeigt. Bei "Layout" bestimmt die Lage und Orientierung der Ein- und Ausgänge in der Schaltung die Position der Pins. Hier sind auch Pins oben und unten möglich. Bei der Wahl von "DIL-Gehäuse" wird ein DIL-Gehäuse zur Darstellung verwendet. Die Pin-Nummern der Ein- und Ausgänge bestimmen hier die Position der Pins.

Benutzerdefinierte Form
: Importieren einer SVG-Datei

Höhe
: Höhe des Symbols, wenn diese Schaltung in eine andere eingefügt wird.

Pinanzahl DIL
: Anzahl der Pins des DILs. Wird hier eine 0 eingetragen, wird die Anzahl automatisch bestimmt.

ROM Inhalte
: Inhalt aller ROM Bausteine

Zeige Messwertetabelle bei Simulationsstart
: Beim Start der Simulation wird eine Tabelle mit den Messwerten angezeigt.

Zeige Messwertegraph bei Simulationsstart
: Beim Start der Simulation wird ein Graph mit den Messwerten angezeigt.

Zeige Messwertegraph im Gatterschrittmodus bei Simulationsstart
: Beim Start der Simulation wird ein Graph mit den Messwerten im Gatterschrittmodus angezeigt. Dabei werden alle Gatterwechsel angezeigt.

Datei beim Start in den Programmspeicher laden.
: Wird ein Prozessor simuliert, der einen RAM-Baustein als Programmspeicher verwendet, ist es schwierig, diesen Prozessor zu starten, da der RAM Inhalt beim Start der Simulation immer mit Nullen initialisiert wird. Diese Einstellung erlaubt das Laden von Daten in den Programmspeicher. Der Programmspeicher in der Simulation muss als solcher markiert sein.

Programdatei
: Datei welche beim Start der Simulation in den Programspeicher geladen werden soll.

Schaltung ist generisch
: Erlaubt die Erzeugung von generischen Schaltungen.

C Steuerung per Kommandozeile

java -cp Digital.jar CLI

test -circ [String] [-tests [String]] [-allowMissingInputs]:

Der erste Dateiname gibt die zu testende Schaltung an. Wenn ein zweiter Dateiname angegeben wird, werden die Testfälle aus dieser Datei ausgeführt. Wird kein zweiter Dateiname angegeben, werden die Tests aus der ersten Datei ausgeführt.

Optionen:

-circ [String(def:)]

Name der zu testenden Datei.

[-tests [String(def:)]]

Name einer Datei mit Testfällen.

[-allowMissingInputs(def: false)]

Erlaubt das Fehlen von Eingängen in der Schaltung die im Testfall definiert sind. Dies kann sinnvoll sein, wenn es mehrere mögliche Lösungen gibt, die von verschiedenen Eingängen abhängig sein können.

svg -dig [String] [-svg [String]] [-ieee] [-LaTeX] [-pinsInMathMode] [-hideTest] [-noShapeFilling] [-smallIO] [-noPinMarker] [-thinnerLines] [-highContrast] [-monochrome]:

Kann verwendet werden, um aus einer Schaltung eine SVG-Datei zu erzeugen.

Optionen:

-dig [String(def:)]

Der Dateiname der Schaltung.

[-svg [String(def:)]]

Der Name der zu schreibenden SVG-Datei.

[-ieee(def: false)]

Verwendung der IEEE Symbole.

[-LaTeX(def: false)]

Text wird in LaTeX-Notation eingefügt. Inkscape ist für die Weiterverabeitung erforderlich.

[-pinsInMathMode(def: false)]

Für Pin-Labels auch dann den Math-Mode verwenden, wenn keine Indizes enthalten sind.

[-hideTest(def: false)]

Testfälle verbergen

[-noShapeFilling(def: false)]

Polygone werden nicht ausgefüllt.

[-smallIO(def: false)]

Ein- und Ausgänge werden als kleine Kreise dargestellt.

[-noPinMarker(def: false)]

Die blauen und roten Pin-Marker an den Symbolen entfallen.

[-thinnerLines(def: false)]

Wenn gesetzt, werden die Linen etwas dünner gezeichnet.

[-highContrast(def: false)]

Leitungen und der Text der Pins werden in Schwarz ausgegeben.

[-monochrome(def: false)]

Es werden nur Graustufen verwendet.

stats -dig [String] [-csv [String]]:

Erzeugt eine CSV Datei welche die Schaltungsstatistik enthält. Aufgeführt sind alle verwendeten Komponenten.

Optionen:

-dig [String(def:)]

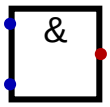
Name der Schaltung.

[-csv [String(def:)]]

Name der Ausgabedatei. Wenn diese Option fehlt, erfolgt eine Ausgabe auf die Konsole.

D Bauteile

1. Logisch



1.1. Und

Binäres Und-Gatter. Gibt eine 1 aus, wenn alle Eingänge 1 sind. Es können auch Busse mit mehreren Bits pro Eingang verknüpft werden. In diesem Fall wird ein bitweises Und ausgeführt. Das heißt, dass das jeweils unterste Bit aller Eingänge mit Und verknüpft wird, und als unterstes Bit am Ausgang ausgegeben wird. Das gleiche passiert mit Bit 1, Bit 2 u.s.w. Exportierbar zu VHDL/Verilog.

Eingänge

In_1
: Der 1. Eingangswert für die Verknüpfung.

In_2
: Der 2. Eingangswert für die Verknüpfung.

Ausgänge

out
: Gibt das Ergebnis der Verknüpfung zurück.

Veränderbare Attribute

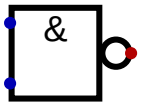
Daten-Bits
: Anzahl der Daten-Bits, die verwendet werden.

Anzahl der Eingänge
: Legt die Anzahl der Eingänge fest. Alle Eingänge müssen beschaltet werden.

inverse Eingänge
: Es können die Eingänge ausgewählt werden, welche invertiert werden sollen.

Rotation
: Legt die Ausrichtung des Elementes in der Schaltung fest.

Breites Symbol
: Verwendet ein breiteres Symbol zur Darstellung des Gatters.



1.2. Nicht Und

Eine Kombination aus Und- und Nicht-Verknüpfung. Alle Eingangssignale werden mit Und verknüpft, und das Ergebnis wird vor der Ausgabe invertiert. Es können auch Busse mit mehreren Bits pro Eingang verknüpft werden. In diesem Fall wird die Verknüpfung auf jedes Bit der Eingänge angewendet. Exportierbar zu VHDL/Verilog.

Eingänge

In_1
: Der 1. Eingangswert für die Verknüpfung.

In_2
: Der 2. Eingangswert für die Verknüpfung.

Ausgänge

out
: Gibt das Ergebnis der Verknüpfung zurück.

Veränderbare Attribute

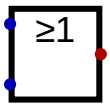
Daten-Bits
: Anzahl der Daten-Bits, die verwendet werden.

Anzahl der Eingänge
: Legt die Anzahl der Eingänge fest. Alle Eingänge müssen beschaltet werden.

inverse Eingänge
: Es können die Eingänge ausgewählt werden, welche invertiert werden sollen.

Rotation
: Legt die Ausrichtung des Elementes in der Schaltung fest.

Breites Symbol
: Verwendet ein breiteres Symbol zur Darstellung des Gatters.



1.3. Oder

Binäres Oder-Gatter. Gibt eine 1 aus, wenn mindestens einer der Eingänge 1 ist. Es können auch Busse mit mehreren Bits pro Eingang verknüpft werden. In diesem Fall wird ein bitweises Oder ausgeführt. Das heißt, dass das jeweils unterste Bit aller Eingänge mit Oder verknüpft wird, und als unterstes Bit am Ausgang ausgegeben wird. Das gleiche passiert mit Bit 1, Bit 2 u.s.w. Exportierbar zu VHDL/Verilog.

Eingänge

In_1
: Der 1. Eingangswert für die Verknüpfung.

In_2
: Der 2. Eingangswert für die Verknüpfung.

Ausgänge

out
: Gibt das Ergebnis der Verknüpfung zurück.

Veränderbare Attribute

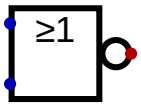
Daten-Bits
: Anzahl der Daten-Bits, die verwendet werden.

Anzahl der Eingänge
: Legt die Anzahl der Eingänge fest. Alle Eingänge müssen beschaltet werden.

inverse Eingänge
: Es können die Eingänge ausgewählt werden, welche invertiert werden sollen.

Rotation
: Legt die Ausrichtung des Elementes in der Schaltung fest.

Breites Symbol
: Verwendet ein breiteres Symbol zur Darstellung des Gatters.



1.4. Nicht Oder

Eine Kombination aus Oder und Nicht-Verknüpfung. Alle Eingangssignale werden mit Oder verknüpft, und das Ergebnis wird vor der Ausgabe invertiert. Es können auch Busse mit mehreren Bits pro Eingang verknüpft werden. In diesem Fall wird die Verknüpfung auf jedes Bit der Eingänge angewendet. Exportierbar zu VHDL/Verilog.

Eingänge

In_1
: Der 1. Eingangswert für die Verknüpfung.

In_2
: Der 2. Eingangswert für die Verknüpfung.

Ausgänge

out
: Gibt das Ergebnis der Verknüpfung zurück.

Veränderbare Attribute

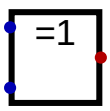
Daten-Bits
: Anzahl der Daten-Bits, die verwendet werden.

Anzahl der Eingänge
: Legt die Anzahl der Eingänge fest. Alle Eingänge müssen beschaltet werden.

inverse Eingänge
: Es können die Eingänge ausgewählt werden, welche invertiert werden sollen.

Rotation
: Legt die Ausrichtung des Elementes in der Schaltung fest.

Breites Symbol
: Verwendet ein breiteres Symbol zur Darstellung des Gatters.



1.5. Exklusiv Oder

Bei zwei Eingängen wird eine 1 ausgegeben, wenn genau einer der Eingänge auf 1 gesetzt ist. Werden mehr als zwei Eingänge verwendet, werden die XOR-Gater kaskadiert (A XOR B XOR C = (A XOR B) XOR C). Es können auch Busse mit mehreren Bits pro

Eingang verknüpft werden. In diesem Fall wird die Verknüpfung auf jedes Bit der Eingänge angewendet. Exportierbar zu VHDL/Verilog.

Eingänge

In_1
: Der 1. Eingangswert für die Verknüpfung.

In_2
: Der 2. Eingangswert für die Verknüpfung.

Ausgänge

out
: Gibt das Ergebnis der Verknüpfung zurück.

Veränderbare Attribute

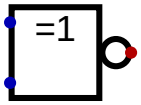
Daten-Bits
: Anzahl der Daten-Bits, die verwendet werden.

Anzahl der Eingänge
: Legt die Anzahl der Eingänge fest. Alle Eingänge müssen beschaltet werden.

inverse Eingänge
: Es können die Eingänge ausgewählt werden, welche invertiert werden sollen.

Rotation
: Legt die Ausrichtung des Elementes in der Schaltung fest.

Breites Symbol
: Verwendet ein breiteres Symbol zur Darstellung des Gatters.



1.6. Nicht Exklusiv Oder

Eine Kombination aus XOR und NOT. Die Eingänge werden per XOR verknüpft und das Ergebnis wird vor der Ausgabe invertiert. Es können auch Busse mit mehreren Bits pro Eingang verknüpft werden. In diesem Fall wird die Verknüpfung auf jedes Bit der Eingänge angewendet. Exportierbar zu VHDL/Verilog.

Eingänge

In_1
: Der 1. Eingangswert für die Verknüpfung.

In_2
: Der 2. Eingangswert für die Verknüpfung.

Ausgänge

out
: Gibt das Ergebnis der Verknüpfung zurück.

Veränderbare Attribute

Daten-Bits
: Anzahl der Daten-Bits, die verwendet werden.

Anzahl der Eingänge
: Legt die Anzahl der Eingänge fest. Alle Eingänge müssen beschaltet werden.

inverse Eingänge
: Es können die Eingänge ausgewählt werden, welche invertiert werden sollen.

Rotation
: Legt die Ausrichtung des Elementes in der Schaltung fest.

Breites Symbol
: Verwendet ein breiteres Symbol zur Darstellung des Gatters.

1.7. Nicht

Invertiert den Eingang und gibt diesen Wert aus. Es können auch Busse mit mehreren Bits pro Eingang verwendet werden. In diesem Fall wird jedes einzelne Bit des Eingangs invertiert. Exportierbar zu VHDL/Verilog.

Eingänge

in
: Der Eingangswert des Inverters.

Ausgänge

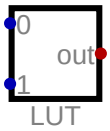
out
: Gibt den negierten Eingangswert aus.

Veränderbare Attribute

Daten-Bits
: Anzahl der Daten-Bits, die verwendet werden.

Rotation
: Legt die Ausrichtung des Elementes in der Schaltung fest.

Breites Symbol
: Verwendet ein breiteres Symbol zur Darstellung des Gatters.



1.8. LookUpTable

Erzeugt einen Ausgabewert aus einer Tabelle, indem aus den Eingangsbits ein Tabellenindex gebildet wird. Der Wert an dieser Position wird ausgegeben. Auf diese Weise kann jedes kombinatorische Gatter erzeugt werden. Exportierbar zu VHDL/Verilog.

Eingänge

0
: Eingang 0. Der Eingang legt, in Kombination mit den anderen Eingängen, die Adresse des gewünschten Ausgabewertes fest.

1
: Eingang 1. Der Eingang legt, in Kombination mit den anderen Eingängen, die Adresse des gewünschten Ausgabewertes fest.

Ausgänge

out
: Gibt den Wert an der ausgewählten Adresse aus.

Veränderbare Attribute

Daten-Bits
: Anzahl der Daten-Bits, die verwendet werden.

Anzahl der Eingänge
: Legt die Anzahl der Eingänge fest. Alle Eingänge müssen beschaltet werden.

Bezeichnung
: Die Bezeichnung dieses Elementes.

Daten
: Die Daten, welche in diesem Element gespeichert sind.

Rotation
: Legt die Ausrichtung des Elementes in der Schaltung fest.

2. IO

2.1. Ausgang

Kann genutzt werden, um ein Ausgangssignal in einer Schaltung anzuzeigen. Dieses Element wird auch genutzt, um eine Verbindung zu einer einbettenden Schaltung herzustellen. In diesem Fall ist der Anschluss bidirektional. Wird zudem verwendet, um einen Hardware Pin zuzuordnen, wenn Code für ein CPLD oder FPGA erzeugt wird. Exportierbar zu VHDL/Verilog.

Eingänge

in
: Der hier angelegte Wert wird als Ausgangswert bereit gestellt.

Veränderbare Attribute

Daten-Bits
: Anzahl der Daten-Bits, die verwendet werden.

Bezeichnung
: Die Bezeichnung dieses Elementes.

Beschreibung
: Eine kurze Beschreibung des Elementes.

Rotation
: Legt die Ausrichtung des Elementes in der Schaltung fest.

Zahlenformat
: Das Zahlenformat für die Anzeige der Werte.

Pinnummer
: Nummer dieses Pins. Wird für die Darstellung einer Schaltung als DIL-Gehäuse und die Pinzuordnung bei der Programmierung eines CPLD verwendet. Bei mehreren Bits, können alle Pinnummern als kommagetrennte Liste angegeben werden.

Im Messwertegraph anzeigen
: Zeigt den Wert im Messwertegraph an.

2.2. LED

Eine Leuchtdiode, welche beispielsweise zur Visualisierung eines Ausgangswertes verwendet werden kann. Nimmt ein Bit entgegen. Leuchtet, wenn der Eingang auf 1 gesetzt ist.

Eingänge

in
: LED Eingang. LED leuchtet, wenn Eingang auf 1 gesetzt ist.

Veränderbare Attribute

Bezeichnung
: Die Bezeichnung dieses Elementes.

Farbe
: Die Farbe des Elementes.

Rotation
: Legt die Ausrichtung des Elementes in der Schaltung fest.

Größe
: Die Größe der Darstellung in der Schaltung.

2.3. Eingang

Kann genutzt werden, um ein Eingangssignal in einer Schaltung interaktiv per Maus zu verändern. Dieses Element wird auch genutzt, um eine Verbindung zu einer einbettenden Schaltung herzustellen. In diesem Fall ist der Anschluss bidirektional. Wird zudem verwendet, um einen Hardware Pin zuzuordnen, wenn Code für ein CPLD oder FPGA erzeugt wird. Exportierbar zu VHDL/Verilog.

Ausgänge

out
: An diesem Ausgang wird der anliegende Wert ausgegeben.

Veränderbare Attribute

Daten-Bits
: Anzahl der Daten-Bits, die verwendet werden.

Bezeichnung
: Die Bezeichnung dieses Elementes.

Beschreibung
: Eine kurze Beschreibung des Elementes.

Rotation
: Legt die Ausrichtung des Elementes in der Schaltung fest.

Vorgabe
: Dieser Wert wird beim Schaltungsstart gesetzt. Ein "Z" steht für "hochohmig".

Eingang kann hochohmig sein
: Wenn gesetzt, ist ein hochohmiger Eingang erlaubt. Bei einem Eingangselement ist ein hochohmiger Eingang auch erlaubt, wenn der Vorgabewert auf hochohmig ("Z") gesetzt wird.

Keine Ausgabe von Null.
: Vermeidet die Ausgabe von Null. Ist vor allem beim Aufbau von Relaisschaltungen hilfreich. Kann nur aktiviert werden, wenn ein hochohmiger Ausgang erlaubt ist.

Zahlenformat
: Das Zahlenformat für die Anzeige der Werte.

Pinnummer
: Nummer dieses Pins. Wird für die Darstellung einer Schaltung als DIL-Gehäuse und die Pinzuordnung bei der Programmierung eines CPLD verwendet. Bei mehreren Bits, können alle Pinnummern als kommagetrennte Liste angegeben werden.

Im Messwertegraph anzeigen
: Zeigt den Wert im Messwertegraph an.

2.4. Takteingang

Ein Taktsignal. Dieses Taktsignal kann über die Echtzeituhr gesteuert werden. Abhängig von der Komplexität der Schaltung kann die erreichte Taktfrequenz geringer sein als der vorgegebene Wert. Ist die Frequenz größer als 50Hz, wird nicht mehr bei jedem Taktsignal die grafische Darstellung der Schaltung aktualisiert, sodass die Leitungsfarben nicht mehr aktualisiert werden. Ist die Echtzeituhr nicht aktiviert, kann der Takt durch Mausklicks weiter geschaltet werden. Wird auch verwendet, um einen Hardware Pin zuzuordnen, wenn Code für ein CPLD oder FPGA erzeugt wird. Exportierbar zu VHDL/Verilog.

Ausgänge

C
: Wechselt im vorgegebenen Takt zwischen 0 und 1.

Veränderbare Attribute

Bezeichnung
: Die Bezeichnung dieses Elementes.

Echtzeittakt starten
: Wenn eingeschaltet, wird beim Start der Schaltung der Echtzeittakt gestartet.

Frequenz/Hz
: Gibt die Frequenz an, wenn der Echtzeittakt aktiviert ist

Rotation
: Legt die Ausrichtung des Elementes in der Schaltung fest.

Pinnummer
: Nummer dieses Pins. Wird für die Darstellung einer Schaltung als DIL-Gehäuse und die Pinzuordnung bei der Programmierung eines CPLD verwendet. Bei mehreren Bits, können alle Pinnummern als kommagetrennte Liste angegeben werden.

2.5. Taster

Ein einfacher Taster, welcher in seinen Ausgangszustand zurückkehrt, wenn er nicht gehalten wird.

Ausgänge

out
: Das Ausgangssignal des Tasters.

Veränderbare Attribute

Bezeichnung
: Die Bezeichnung dieses Elementes.

Active Low
: Wenn gesetzt, ist der Ausgang im aktiven Zustand Low.

Auf Tastatur legen
: Taste wird durch die Tastatur bedienbar. Um die Cursor-Tasten zu nutzen, kann als Bezeichnung UP, DOWN, LEFT oder RIGHT verwendet werden.

Rotation
: Legt die Ausrichtung des Elementes in der Schaltung fest.

Im Messwertegraph anzeigen
: Zeigt den Wert im Messwertegraph an.

2.6. DIP-Schalter

Einfacher Schiebeschalter, der High oder Low ausgeben kann.

Ausgänge

out
: Der Ausgabewert des Schalters.

Veränderbare Attribute

Bezeichnung
: Die Bezeichnung dieses Elementes.

Beschreibung
: Eine kurze Beschreibung des Elementes.

Rotation
: Legt die Ausrichtung des Elementes in der Schaltung fest.

Ausgabe ist High
: Der Vorgabewert des DIP-Schalters, wenn die Simulation gestartet wird.

Text

2.7. Text

Zeigt einen einfachen Text in der Schaltung an. Hat keine weitere Funktion für die Simulation. Der Text kann im Attribute-Dialog geändert werden.

Veränderbare Attribute

Beschreibung
: Eine kurze Beschreibung des Elementes.

Schriftgröße
: Legt die für diesen Text zu verwendende Schriftgröße fest.

Rotation
: Legt die Ausrichtung des Elementes in der Schaltung fest.

Ausrichtung
: Lage der Koordinate relativ zum Text.

An Gitter ausrichten
: Wenn gesetzt, wird das Element am Gitter ausgerichtet.

•?

2.8. Messwert

Ein Messwert, welcher im Messwertegraphen bzw. in der Messwertetabelle dargestellt wird. Dieses Element kann verwendet werden, um auf einfache Weise Werte aus eingebetteten Schaltungen zu beobachten. Hat keine weitere Funktion für die Simulation.

Eingänge

in
: Hier wird der Messwert angeschlossen.

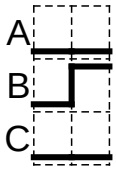
Veränderbare Attribute

Bezeichnung
: Die Bezeichnung dieses Elementes.

Rotation
: Legt die Ausrichtung des Elementes in der Schaltung fest.

Zahlenformat
: Das Zahlenformat für die Anzeige der Werte.

Im Messwertegraph anzeigen
: Zeigt den Wert im Messwertegraph an.



2.9. Messwertegraph

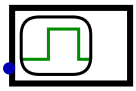
Zeigt einen Messwertegraphen innerhalb des Schaltkreisbereichs. Es können sowohl komplette Taktschritte als auch einzelne Gatter-Veränderungen angezeigt werden. Hat keine weitere Funktion für die Simulation.

Veränderbare Attribute

Zeige Einzelgatterschritte
: Zeigt in der Grafik alle Einzelgatterschritte an.

Maximale Messpunktezahl
: Die maximale Anzahl an Messpunkten, die gespeichert werden, bevor die ältesten Messungen verworfen werden.

An Gitter ausrichten
: Wenn gesetzt, wird das Element am Gitter ausgerichtet.



2.10. Getriggerter Messwertegraph

Zeigt einen Messwertegraph, wobei nur dann Messwerte gespeichert werden, wenn sich das Eingangssignal verändert. Das Speichern findet statt, wenn sich die Schaltung stabilisiert hat. Der Trigger startet nicht die Messung wie bei einem echten Oszilloscop, sondern jedes Triggerereignis speichert einen einzigen Messwert für jedes der angezeigten Signale. Als direkten Eingang gibt es nur den Trigger. Als Signale können die Ein- und Ausgänge der Schaltung, Flipflops und Register und der Messwert verwendet werden. In den jeweiligen Komponenten kann dies jeweils aktiviert werden.

Eingänge

T
: Ein Änderung an diesem Eingang veranlasst das Speichern von Messwerten.

Veränderbare Attribute

Bezeichnung
: Die Bezeichnung dieses Elementes.

Trigger
: Triggerbedingung für die Datenaufzeichnung.

Maximale Messpunktezahl
: Die maximale Anzahl an Messpunkten, die gespeichert werden, bevor die ältesten Messungen verworfen werden.

3. IO - Anzeigen

3.1. RGB-LED

Eine RGB-Leuchtdiode dessen Farbe über drei Eingänge kontrolliert werden kann. An jedem der drei Eingängen wird ein Farbkanal angeschlossen.

Eingänge

R
: Der rote Farbkanal.

G
: Der grüne Farbkanal.

B
: Der blaue Farbkanal.

Veränderbare Attribute

Daten-Bits
: Anzahl der Daten-Bits, die verwendet werden.

Bezeichnung
: Die Bezeichnung dieses Elementes.

Rotation
: Legt die Ausrichtung des Elementes in der Schaltung fest.

Größe
: Die Größe der Darstellung in der Schaltung.

3.2. LED mit zwei Anschlüssen

LED mit zwei Anschlüssen für die Kathode und die Anode. Die LED leuchtet nur, wenn die Anode auf High und die Kathode auf Low gelegt wird. Diese LED kann nicht als Pull-Down-Widerstand verwendet werden. Es handelt sich ausschließlich um ein Anzeigeelement.
Der abgebildete Widerstand soll lediglich den erforderlichen Serienwiderstand zur Strombegrenzung symbolisieren.

Eingänge

A
: Die Anode der LED.

C
: Die Kathode der LED.

Veränderbare Attribute

Bezeichnung
: Die Bezeichnung dieses Elementes.

Farbe
: Die Farbe des Elementes.

Rotation
: Legt die Ausrichtung des Elementes in der Schaltung fest.

3.3. Taster mit LED

Ein einfacher Taster, welcher in seinen Ausgangszustand zurückkehrt, wenn er nicht gehalten wird. Der Taster verfügt über eine LED, die über ein Eingangssignal geschaltet werden kann.

Eingänge

in
: Eingang zur Steuerung der LED.

Ausgänge

out
: Das Ausgangssignal des Tasters.

Veränderbare Attribute

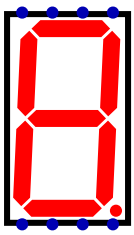
Bezeichnung
: Die Bezeichnung dieses Elementes.

Active Low
: Wenn gesetzt, ist der Ausgang im aktiven Zustand Low.

Auf Tastatur legen
: Taste wird durch die Tastatur bedienbar. Um die Cursor-Tasten zu nutzen, kann als Bezeichnung UP, DOWN, LEFT oder RIGHT verwendet werden.

Farbe
: Die Farbe des Elementes.

Rotation
: Legt die Ausrichtung des Elementes in der Schaltung fest.



3.4. Siebensegmentanzeige

Siebensegmentanzeige, bei der jedes Segment über einen eigenen Eingang gesteuert werden kann.

Eingänge

a
: Dieser Eingang steuert die obere horizontale Linie.

b
: Dieser Eingang steuert die obere rechte vertikale Linie.

c
: Dieser Eingang steuert die untere rechte vertikale Linie.

d
: Dieser Eingang steuert die untere horizontale Linie.

e
: Dieser Eingang steuert die untere linke vertikale Linie.

f
: Dieser Eingang steuert die obere linke vertikale Linie.

g
: Dieser Eingang steuert die mittlere horizontale Linie.

dp
: Dieser Eingang steuert den Dezimalpunkt.

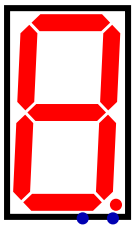
Veränderbare Attribute

Farbe
: Die Farbe des Elementes.

Gemeinsamer Anschluss
: Wenn gesetzt, hat die Anzeige einen Anschluss mit einer gemeinsamen Kathode oder Anode.

Gemeinsame
: Art des gemeinsamen Anschlusses.

Flimmern vermeiden
: Die Schaltfrequenz in der Simulation kann nicht so hoch werden, dass das menschliche Auge kein Flimmern mehr wahrnimmt. Um dennoch das Flackern zu unterdrücken, kann bei den LEDs mit dieser Option ein "nachleuchten" eingeschaltet werden. Dabei leuchten die LEDs weiter, auch wenn einer der Anschlüsse zu High-Z wechselt.



3.5. Siebensegmentanzeige Hex

Siebensegmentanzeige mit einem 4 Bit hexadezimalen Eingang.

Eingänge

d
: Der 4-Bit-Wert dieses Eingangs wird als Hex-Ziffer dargestellt.

dp
: Dieser Eingang steuert den Dezimalpunkt.

Veränderbare Attribute

Farbe
: Die Farbe des Elementes.

Größe
: Die Größe der Darstellung in der Schaltung.

3.6. 16-Segement Anzeige

Der LED Eingang verfügt über 16 Bits welche die Segmente steuern. Der zweite Eingang steuert den Dezimalpunkt.

Eingänge

led
: Bus mit 16-Bits zur Ansteuerung der LEDs.

dp
: Dieser Eingang steuert den Dezimalpunkt.

Veränderbare Attribute

Farbe
: Die Farbe des Elementes.

Größe
: Die Größe der Darstellung in der Schaltung.

3.7. Glühlämpchen

Glühlämpchen mit zwei Anschlüssen: Wenn ein Strom fließt, leuchtet das Lämpchen.
Die Stromrichtung spielt keine Rolle. Das Lämpchen leuchtet also, wenn die Eingänge unterschiedliche Werte haben. Das Lämpchen verhält sich damit ähnlich wie ein XOr Gatter.

Eingänge

A
: Anschluss

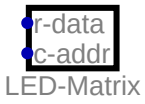
B
: Anschluss

Veränderbare Attribute

Bezeichnung
: Die Bezeichnung dieses Elementes.

Farbe
: Die Farbe des Elementes.

Rotation
: Legt die Ausrichtung des Elementes in der Schaltung fest.



3.8. LED-Matrix

Eine Matrix aus LEDs. Die LEDs werden in einem eigenen Fenster dargestellt. Die LEDs einer Spalte der Anzeige werden durch ein Datenwort gesteuert. An einem weiteren Eingang wird die aktuelle Spalte ausgewählt. Es Wird damit eine gemultiplexte Anzeige realisiert. Die LEDs können in der Simulation unendlich lange nachleuchten, um ein Flimmern der Anzeige zu verhindern.

Eingänge

r-data
: Der Zeilen-Zustand der LEDs einer Spalte. Jedes Bit in diesem Datenwort repräsentiert den Zustand einer Zeile der aktuellen Spalte.

c-addr
: Die Nummer der aktuellen Spalte, dessen Zustand gerade am anderen Eingang anliegt.

Veränderbare Attribute

Bezeichnung
: Die Bezeichnung dieses Elementes.

Zeilen
: Gibt direkt die Zahl der Zeilen an, indem die Anzahl der Bits des Zeilenwortes festgelegt wird.

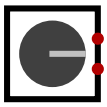
Adressbits der Spalten
: Adressiert die einzelnen Spalten. Drei Bits bedeuten also acht Spalten.

Farbe
: Die Farbe des Elementes.

Flimmern vermeiden
: Die Schaltfrequenz in der Simulation kann nicht so hoch werden, dass das menschliche Auge kein Flimmern mehr wahrnimmt. Um dennoch das Flackern zu unterdrücken, kann bei den LEDs mit dieser Option ein "nachleuchten" eingeschaltet werden. Dabei leuchten die LEDs weiter, auch wenn einer der Anschlüsse zu High-Z wechselt.

Rotation
: Legt die Ausrichtung des Elementes in der Schaltung fest.

4. IO - Mechanisch



4.1. Drehencoder

Drehknopf mit Drehencoder zur Erfassung einer Drehbewegung.

Ausgänge

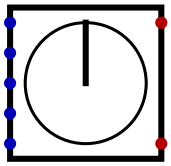
A
: Encodersignal A

B
: Encodersignal B

Veränderbare Attribute

Bezeichnung
: Die Bezeichnung dieses Elementes.

Rotation
: Legt die Ausrichtung des Elementes in der Schaltung fest.



4.2. Schrittmotor, unipolar

Unipolarer Schrittmotor mit zwei Endlagenschaltern. Die Endlagenschalter können 'überfahren' werden. Unterstützt wird das Vollschrittverfahren, das Halbschrittverfahren und Wave-Drive.

Eingänge

P0
: Phase 0

P1
: Phase 1

P2
: Phase 2

P3
: Phase 3

com
: Gemeinsamer Mittenanschluss

Ausgänge

S0
: Endlagenschalter 0, wird zu Eins, wenn der Motorwinkel 0° beträgt.

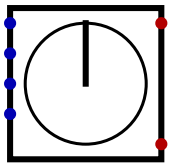
S1
: Endlagenschalter 1, wird zu Eins, wenn der Motorwinkel 180° beträgt.

Veränderbare Attribute

Bezeichnung
: Die Bezeichnung dieses Elementes.

invertierter Ausgang
: Wenn gesetzt, wird der Ausgang invertiert.

Rotation
: Legt die Ausrichtung des Elementes in der Schaltung fest.



4.3. Schrittmotor, bipolar

Bipolarer Schrittmotor mit zwei Endlagenschaltern. Die Endlagenschalter können 'überfahren' werden. Unterstützt wird das Vollschrittverfahren, das Halbschrittverfahren und Wave-Drive.

Eingänge

A+
: Wicklung A, Pluspol

A-
: Wicklung A, Minuspol

B+
: Wicklung B, Pluspol

B-
: Wicklung B, Minuspol

Ausgänge

S0
: Endlagenschalter 0, wird zu Eins, wenn der Motorwinkel 0° beträgt.

S1
: Endlagenschalter 1, wird zu Eins, wenn der Motorwinkel 180° beträgt.

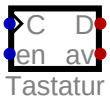
Veränderbare Attribute

Bezeichnung
: Die Bezeichnung dieses Elementes.

invertierter Ausgang
: Wenn gesetzt, wird der Ausgang invertiert.

Rotation
: Legt die Ausrichtung des Elementes in der Schaltung fest.

5. IO - Peripherie



5.1. Tastatur

Eine Tastatur, welche zur Eingabe von Text verwendet werden kann. Dieses Element puffert die Eingaben, welche dann ausgelesen werden können. Es wird ein eigenes Fenster für die Eingabe geöffnet.

Eingänge

C
: Takt. Eine steigende Flanke entfernt das älteste Zeichen aus dem Buffer.

en
: Wenn gesetzt, ist der Ausgang D aktiv und ein Tastendruck wird ausgegeben. Ist gleichzeitig die Freigabe für den Takt.

Ausgänge

D
: Die letzte Taste oder 0, wenn keine Taste gedrückt. Ausgegeben wird der 16 Bit Java char-Wert.

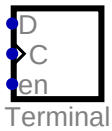
av
: Dieser Ausgang zeigt an, dass Zeichen vorhanden sind. Er kann genutzt werden, um einen Interrupt auszulösen.

Veränderbare Attribute

Bezeichnung
: Die Bezeichnung dieses Elementes.

inverse Eingänge
: Es können die Eingänge ausgewählt werden, welche invertiert werden sollen.

Rotation
: Legt die Ausrichtung des Elementes in der Schaltung fest.



5.2. Terminal

Ein Terminal, auf dem ASCII-Zeichen ausgegeben werden können. Öffnet ein eigenes Fenster, um die Ausgabe anzuzeigen.

Eingänge

D
: Über diesen Eingang werden die anzuzeigenden Daten an das Terminal weitergegeben.

C
: Takteingang. Eine steigende Flanke gibt das anliegende Datenwort auf dem Terminal aus.

en
: Ein High an diesem Eingang aktiviert den Takteingang.

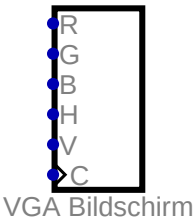
Veränderbare Attribute

Zeichen pro Zeile
: Die Anzahl der Zeichen, die in einer Zeile angezeigt werden können.

Zeilen
: Die Anzahl der anzuzeigenden Zeilen.

Bezeichnung
: Die Bezeichnung dieses Elementes.

Rotation
: Legt die Ausrichtung des Elementes in der Schaltung fest.



5.3. VGA Bildschirm

Analysiert die eingehenden Video-Signale und zeigt die entsprechende Grafik an. Da die Simulation nicht in Echtzeit laufen kann, wird zusätzlich zu den Videosignalen der Pixeltakt benötigt.

Eingänge

R
: Der rote Farbanteil

G
: Der grüne Farbanteil

B
: Der blaue Farbanteil

H
: Das horizontale Synchronisationssignal

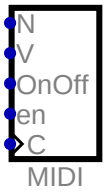
V
: Das vertikale Synchronisationssignal

C
: Der Pixeltakt

Veränderbare Attribute

Bezeichnung
: Die Bezeichnung dieses Elementes.

Rotation
: Legt die Ausrichtung des Elementes in der Schaltung fest.



5.4. MIDI

Nutzt das MIDI-System, um Noten abzuspielen.

Eingänge

N
: Note

V
: Lautstärke

OnOff
: Wenn gesetzt, enspricht das dem Drücken einer Keyboard-Taste (key down). Wenn nicht gesetzt, entspricht das einem Loslassen der Taste (key up).

en
: Wenn dieser eingang high ist, kann der Baustein angesprochen werden.

C
: Takteingang

Veränderbare Attribute

Bezeichnung
: Die Bezeichnung dieses Elementes.

MIDI-Kanal
: Legt den MIDI-Kanal fest.

MIDI-Instrument
: Das MIDI-Instrument, welches verwendet werden soll.

Programmwechsel erlauben
: Fügt einen weiteren Eingang PC hinzu. Wird dieser Eingang auf High gesetzt, wird mit dem Wert am Eingang N das Programm (Instrument) gewechselt.

Rotation
: Legt die Ausrichtung des Elementes in der Schaltung fest.

6. Leitungen

6.1. Masse

Eine Masseverbindung. Gibt immer Null aus. Exportierbar zu VHDL/Verilog.

Ausgänge

out
: Dieser Ausgang gibt immer 0 aus.

Veränderbare Attribute

Daten-Bits
: Anzahl der Daten-Bits, die verwendet werden.

Bezeichnung
: Die Bezeichnung dieses Elementes.

Rotation
: Legt die Ausrichtung des Elementes in der Schaltung fest.

6.2. Betriebsspannung

Anschluss zur Betriebsspannung. Gibt immer Eins aus. Exportierbar zu VHDL/Verilog.

Ausgänge

out
: Dieser Ausgang gibt immer 1 aus.

Veränderbare Attribute

Daten-Bits
: Anzahl der Daten-Bits, die verwendet werden.

Bezeichnung
: Die Bezeichnung dieses Elementes.

Rotation
: Legt die Ausrichtung des Elementes in der Schaltung fest.

6.3. Konstante

Ein Element, welches einen konstanten Wert ausgibt. Die Konstante kann über den Attribute-Dialog festgelegt werden. Exportierbar zu VHDL/Verilog.

Ausgänge

out
: Gibt den gesetzten Wert als Konstante aus.

Veränderbare Attribute

Daten-Bits
: Anzahl der Daten-Bits, die verwendet werden.

Wert
: Der Wert der Konstanten.

Rotation
: Legt die Ausrichtung des Elementes in der Schaltung fest.

Zahlenformat
: Das Zahlenformat für die Anzeige der Werte.

6.4. Tunnel

Verbindet Elemente, ohne eine Leitung zu ziehen. Alle Tunnelelemente, welche denselben Netznamen tragen, sind miteinander verbunden. Der Tunnel ist immer lokal, es können keine Verbindungen über Schaltungsgrenzen hinaus erzeugt werden. Unbenannte Tunnel haben keine Funktion. Exportierbar zu VHDL/Verilog.

Eingänge

in
: Anschluss des Tunnels.

Veränderbare Attribute

Netzname
: Alle Netze mit identischem Namen werden miteinander verbunden.

Rotation
: Legt die Ausrichtung des Elementes in der Schaltung fest.

6.5. Splitter/Merger

Führt mehrere Leitungen zu einem gemeinsamen Bus zusammen, bzw. splittet diesen wieder auf. Mit Bussen wird es möglich z.B. 16 Bit Verbindungen zu erzeugen, ohne 16 einzelne Leitungen verlegen zu müssen. Alle 16 Verbindungen können zu einer Leitung zusammengefasst werden. Der Splitter hat eine Richtung, er kann Signale also nur in eine Richtung weitergeben. Exportierbar zu VHDL/Verilog.

Eingänge

0-3
: Die Eingangsbits 0-3 des Splitters.

4-7
: Die Eingangsbits 4-7 des Splitters.

Ausgänge

0-7
: Die Ausgangsbits 0-7 des Splitters.

Veränderbare Attribute

Eingangsaufteilung
: Sollen z.B. je vier Bits, zwei Bits und noch einmal zwei Bits als Eingänge verwendet werden, kann das mit der Eingabe "4,2,2" konfiguriert werden. Die Zahl gibt dabei jeweils die Bitanzahl an. Als Vereinfachung kann das *-Zeichen verwendet werden: Mit "[Bits]*[Anzahl]" wie "1*16" können 16 einzelne Bits konfiguriert werden. Es können auch die zu verwendenden Bits direkt und in beliebiger Reihenfolge angegeben werden. So werden mit der Eingabe"4-7,0-3" die Bits 4-7 und 0-3 konfiguriert. Diese Notation erlaubt eine beliebige Bit-Anordnung. Die Eingangsbits müssen vollständig und eindeutig angegeben werden.

Ausgangsaufteilung
: Sollen z.B. je vier Bits, zwei Bits und noch einmal zwei Bits als Ausgänge verwendet werden, kann das mit der Eingabe "4,2,2" konfiguriert werden. Die Zahl gibt dabei jeweils die Bitanzahl an. Als Vereinfachung kann das *-Zeichen verwendet werden: Mit "[Bits]*[Anzahl]" wie "1*16" können 16 einzelne Bits konfiguriert werden. Es können auch die zu verwendenden Bits direkt und in beliebiger Reihenfolge angegeben werden. So werden mit der Eingabe"4-7,0-3" die Bits 4-7 und 0-3 konfiguriert. Diese Notation erlaubt eine beliebige Bit-Anordnung. Ausgangsbits können auch mehrfach ausgegeben werden: "0-7,1-6,4-7"

Rotation
: Legt die Ausrichtung des Elementes in der Schaltung fest.

Spiegeln
: Spiegelt das Element in der Schaltung.

Spreizung
: Bestimmt die Spreizung der Ein- und Ausgänge in der Schaltung.

6.6. Treiber

Ein Treiber kann dazu verwendet werden, ein Datenwort nur unter speziellen Voraussetzungen auf eine andere Leitung weiterzureichen. Gesteuert wird der Treiber durch den sel Eingang. Ist der sel Eingang auf 1 gesetzt, wird der am Eingang anliegende Wert zum Ausgang gereicht. Ist der Eingang 0, ist der Ausgang hochohmig. Exportierbar zu VHDL/Verilog.

Eingänge

in
: Das Eingangssignal des Treibers.

sel
: Eingang zum Steuern des Treibers. Ist dieser Eingang auf 1 gesetzt, wird der am Eingang anliegende Wert zum Ausgang gereicht. Ist der Eingang 0, ist der Ausgang hochohmig.

Ausgänge

out
: Gibt den am Eingang anliegenden Wert aus, wenn sel auf 1 gesetzt ist. Ist sel auf Null gesetzt, ist der Ausgang hochohmig.

Veränderbare Attribute

Daten-Bits
: Anzahl der Daten-Bits, die verwendet werden.

Tausche Selektorposition
: Mit dieser Option kann der Anschluss des Selektors auf die andere Seite des Multiplexers verschoben werden.

Rotation
: Legt die Ausrichtung des Elementes in der Schaltung fest.

6.7. Treiber, invertierte Auswahl

Ein Treiber kann dazu verwendet werden, ein Datenwort nur unter speziellen Voraussetzungen auf eine andere Leitung weiterzureichen. Gesteuert wird der Treiber durch den sel Eingang. Ist der sel Eingang auf 0 gesetzt, wird der am Eingang anliegende Wert zum Ausgang gereicht. Ist der Eingang 1, ist der Ausgang hochohmig. Exportierbar zu VHDL/Verilog.

Eingänge

in
: Das Eingangssignal des Treibers.

sel
: Eingang zum Steuern des Treibers. Ist dieser Eingang auf 0 gesetzt, wird der am Eingang anliegende Wert zum Ausgang gereicht. Ist der Eingang 1, ist der Ausgang hochohmig.

Ausgänge

out
: Gibt den am Eingang anliegenden Wert aus, wenn sel auf 1 gesetzt ist. Ist sel auf Null gesetzt, ist der Ausgang hochohmig.

Veränderbare Attribute

Daten-Bits
: Anzahl der Daten-Bits, die verwendet werden.

Tausche Selektorposition
: Mit dieser Option kann der Anschluss des Selektors auf die andere Seite des Multiplexers verschoben werden.

Rotation
: Legt die Ausrichtung des Elementes in der Schaltung fest.

6.8. Verzögerung

Verzögert ein Signal für eine einstellbare Anzahl an Gatterlaufzeiten. Alle übrigen Bauteile in Digital haben genau eine Gatterlaufzeit Verzögerung. Mit diesem Bauteil lassen sich damit beliebige Laufzeiten realisieren.

Eingänge

in
: Eingang für das zu verzögernde Signal.

Ausgänge

out
: Das um eine Gatterlaufzeit verzögerte Eingangssignal.

Veränderbare Attribute

Daten-Bits
: Anzahl der Daten-Bits, die verwendet werden.

Dauer
: Dauer der Verzögerung in Gatterlaufzeiten.

Rotation
: Legt die Ausrichtung des Elementes in der Schaltung fest.

6.9. Pull-Up Widerstand

Ist eine Leitung hochohmig, zieht sie ein Pull-Up Widerstand zu High. Ist eine Leitung nicht hochohmig, hat das Element keine Wirkung.

Ausgänge

out
: Ein "Weak High".

Veränderbare Attribute

Daten-Bits
: Anzahl der Daten-Bits, die verwendet werden.

Rotation
: Legt die Ausrichtung des Elementes in der Schaltung fest.

6.10. Pull-Down Widerstand

Ist eine Leitung hochohmig, zieht sie ein Pull-Down Widerstand zu Low. Ist eine Leitung nicht hochohmig, hat das Element keine Wirkung.

Ausgänge

out
: Ein "Weak Low".

Veränderbare Attribute

Daten-Bits
: Anzahl der Daten-Bits, die verwendet werden.

Rotation
: Legt die Ausrichtung des Elementes in der Schaltung fest.

6.11. Nicht verbunden

Dieses Element kann verwendet werden, um eine Leitung auf High-Z zu legen. Wird ein Eingang eines logischen Gatters auf High-Z gesetzt, ist der gelesene Wert undefiniert. Zu beachten ist, dass es in der Realität in vielen Fällen zu überhöhter Stromaufnahme und gar zu Beschädigungen kommen kann, wenn ein Digitaleingang nicht auf Null oder Eins gesetzt wird, sondern unbeschaltet bleibt.

Ausgänge

out
: Dieser Ausgang gibt immer High-Z aus.

Veränderbare Attribute

Daten-Bits
: Anzahl der Daten-Bits, die verwendet werden.

7. Multiplexer

7.1. Multiplexer

Ein Baustein, welcher den Wert eines der Eingänge am Ausgang ausgibt. Über den sel-Eingang wird ausgewählt, welcher der Eingänge ausgegeben werden soll. Exportierbar zu VHDL/Verilog.

Eingänge

sel
: Mit dieser Leitung wird der Dateneingang ausgewählt, welcher am Ausgang ausgegeben werden soll.

in_0
: Der 0. Dateneingang des Multiplexers.

in_1
: Der 1. Dateneingang des Multiplexers.

Ausgänge

out
: Ausgeben wird der Wert, der am gewählten Dateneingang anliegt.

Veränderbare Attribute

Daten-Bits
: Anzahl der Daten-Bits, die verwendet werden.

Anzahl der Auswahlbits
: Anzahl der Bits, die in der Auswahlleitung vorhanden sind.

Tausche Selektorposition
: Mit dieser Option kann der Anschluss des Selektors auf die andere Seite des Multiplexers verschoben werden.

Rotation
: Legt die Ausrichtung des Elementes in der Schaltung fest.

7.2. Demultiplexer

Ein Baustein, welcher einen Eingangswert auf verschiedene Ausgänge ausgeben kann. Gibt das Eingangssignal auf einem wählbaren Ausgang aus. Die anderen Ausgänge werden auf den Vorgabewert gesetzt. Exportierbar zu VHDL/Verilog.

Eingänge

sel
: Mit dieser Leitung wird der Datenausgang ausgewählt, auf welchem der Eingangswert ausgegeben werden soll.

in
: Der Wert dieses Eingangs wird auf einen der Datenausgänge geschaltet.

Ausgänge

out_0
: Datenausgang 0.

out_1
: Datenausgang 1.

Veränderbare Attribute

Daten-Bits
: Anzahl der Daten-Bits, die verwendet werden.

Anzahl der Auswahlbits
: Anzahl der Bits, die in der Auswahlleitung vorhanden sind.

Tausche Selektorposition
: Mit dieser Option kann der Anschluss des Selektors auf die andere Seite des Multiplexers verschoben werden.

Rotation
: Legt die Ausrichtung des Elementes in der Schaltung fest.

Vorgabe
: Dieser Wert wird beim Schaltungsstart gesetzt. Beim Demultiplexer wird dieser Wert für die nicht gewählten Ausgänge gesetzt.

7.3. Dekoder

Eine wählbare Ausgangsleitung geht auf Eins, alle anderen sind Null. Exportierbar zu VHDL/Verilog.

Eingänge

sel
: Mit dieser Leitung wird der zu aktivierende Ausgang ausgewählt. Sein Wert gibt den auf 1 zu schaltenden Ausgang an.

Ausgänge

out_0
: Ausgang 0. Wenn über sel ausgewählt, ist dieser Ausgang 1, sonst 0.

out_1
: Ausgang 1. Wenn über sel ausgewählt, ist dieser Ausgang 1, sonst 0.

Veränderbare Attribute

Anzahl der Auswahlbits
: Anzahl der Bits, die in der Auswahlleitung vorhanden sind.

Tausche Selektorposition
: Mit dieser Option kann der Anschluss des Selektors auf die andere Seite des Multiplexers verschoben werden.

Rotation
: Legt die Ausrichtung des Elementes in der Schaltung fest.

7.4. Bitwähler

Wählt aus einem Datenbus ein einzelnes Bit aus. Exportierbar zu VHDL/Verilog.

Eingänge

in
: Der Eingangsbus

sel
: Dieser Eingang wählt das Bit aus.

Ausgänge

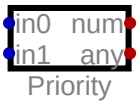
out
: Das ausgewählte Bit.

Veränderbare Attribute

Anzahl der Auswahlbits
: Anzahl der Bits, die in der Auswahlleitung vorhanden sind.

Tausche Selektorposition
: Mit dieser Option kann der Anschluss des Selektors auf die andere Seite des Multiplexers verschoben werden.

Rotation
: Legt die Ausrichtung des Elementes in der Schaltung fest.



7.5. Prioritätsencoder

Ist einer der Eingänge gesetzt, wird dessen Nummer ausgegeben. Sind mehrere Eingänge gleichzeitig gesetzt, wird die höchsten Nummer ausgegeben. Exportierbar zu VHDL/Verilog.

Eingänge

in0
: Der 0. Eingang des PriorityEncoder

in1
: Der 1. Eingang des PriorityEncoder

Ausgänge

num
: Nummer des gesetzten Eingangs.

any
: Wenn dieser Ausgang gesetzt ist, ist mindestens einer der Eingänge gesetzt. Dieser Ausgang ist die Oder-Verknüpfung aller Eingänge.

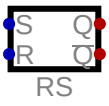
Veränderbare Attribute

Bezeichnung
: Die Bezeichnung dieses Elementes.

Anzahl der Auswahlbits
: Anzahl der Bits, die in der Auswahlleitung vorhanden sind.

Rotation
: Legt die Ausrichtung des Elementes in der Schaltung fest.

8. FlipFlops



8.1. RS-FlipFlop

Ein Baustein, welcher ein einzelnes Bit speichern kann. Bietet die Funktionen "Setzen" und "Rücksetzen". Werden beide Eingänge auf Eins geschaltet, geben auch beide Ausgänge eine Eins aus. Wenn beide Eingänge gleichzeitig zurück auf Null wechseln, ist der Endzustand unbestimmt.

Eingänge

S
: Der Setzen-Eingang. Mit diesem Eingang wird das gespeicherte Bit gesetzt.

R
: Der Rücksetzen-Eingang. Mit diesem Eingang wird das gespeicherte Bit zurückgesetzt.

Ausgänge

Q
: Gibt den gespeicherten Wert aus.

¬Q
: Gibt den gespeicherten Wert negiert zurück.

Veränderbare Attribute

Bezeichnung
: Die Bezeichnung dieses Elementes.

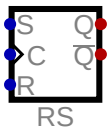
inverse Eingänge
: Es können die Eingänge ausgewählt werden, welche invertiert werden sollen.

Rotation
: Legt die Ausrichtung des Elementes in der Schaltung fest.

Spiegeln
: Spiegelt das Element in der Schaltung.

Vorgabe
: Dieser Wert wird beim Schaltungsstart gesetzt. Beim Demultiplexer wird dieser Wert für die nicht gewählten Ausgänge gesetzt.

Als Messwert verwenden
: Wenn gesetzt, taucht der Wert als Messwert in Graph und Tabelle auf. Zusätzlich muss eine Bezeichnung angegeben werden, welche als Identifikation des Messwertes dienen kann.



8.2. RS-FlipFlop, getaktet

Ein Baustein, welcher ein einzelnes Bit speichern kann. Bietet die Funktionen "Setzen" und "Rücksetzen". Sind bei der steigenden Flanke des Taktes beide Eingänge (S,R) gesetzt, nimmt das Flipflop einen zufälligen Zustand ein.

Eingänge

S
: Der Setzen-Eingang. Mit diesem Eingang wird das gespeicherte Bit gesetzt.

C
: Der Takteingang. Eine steigende Flanke veranlasst die Zustandsänderung.

R
: Der Rücksetzen-Eingang. Mit diesem Eingang wird das gespeicherte Bit zurückgesetzt.

Ausgänge

Q
: Gibt den gespeicherten Wert aus.

¬Q
: Gibt den gespeicherten Wert negiert zurück.

Veränderbare Attribute

Bezeichnung
: Die Bezeichnung dieses Elementes.

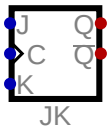
inverse Eingänge
: Es können die Eingänge ausgewählt werden, welche invertiert werden sollen.

Rotation
: Legt die Ausrichtung des Elementes in der Schaltung fest.

Spiegeln
: Spiegelt das Element in der Schaltung.

Vorgabe
: Dieser Wert wird beim Schaltungsstart gesetzt. Beim Demultiplexer wird dieser Wert für die nicht gewählten Ausgänge gesetzt.

Als Messwert verwenden
: Wenn gesetzt, taucht der Wert als Messwert in Graph und Tabelle auf. Zusätzlich muss eine Bezeichnung angegeben werden, welche als Identifikation des Messwertes dienen kann.



8.3. JK-FlipFlop

Bietet die Funktionen zum Speichern (J=K=0), Setzen (J=1, K=0), Rücksetzen (J=0, K=1) und Wechseln (J=K=1). Ein Zustandswechsel findet nur bei einer steigenden Flanke am Eingang C statt. Exportierbar zu VHDL/Verilog.

Eingänge

J
: Der Setzen-Eingang des Flipflops.

C
: Takteingang. Eine steigende Flanke initiiert den Zustandswechsel.

K
: Der Rücksetzen-Eingang des Flipflops.

Ausgänge

Q
: Gibt den gespeicherten Wert aus.

¬Q
: Gibt den gespeicherten Wert negiert aus.

Veränderbare Attribute

Bezeichnung
: Die Bezeichnung dieses Elementes.

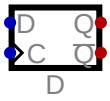
inverse Eingänge
: Es können die Eingänge ausgewählt werden, welche invertiert werden sollen.

Rotation
: Legt die Ausrichtung des Elementes in der Schaltung fest.

Spiegeln
: Spiegelt das Element in der Schaltung.

Vorgabe
: Dieser Wert wird beim Schaltungsstart gesetzt. Beim Demultiplexer wird dieser Wert für die nicht gewählten Ausgänge gesetzt.

Als Messwert verwenden
: Wenn gesetzt, taucht der Wert als Messwert in Graph und Tabelle auf. Zusätzlich muss eine Bezeichnung angegeben werden, welche als Identifikation des Messwertes dienen kann.



8.4. D-FlipFlop

Ein Baustein zum Speichern eines Bits. Der an Eingang D anliegende Wert wird bei einer steigenden Flanke an Eingang C gespeichert. Die Bitbreite kann gewählt werden, so dass mehrere Bits gespeichert werden können. Exportierbar zu VHDL/Verilog.

Eingänge

D
: Das zu speichernde Bit.

C
: Takt des Flipflops. Wechselt dieser Wert auf 1, wird der an D anliegende Wert abgespeichert.

Ausgänge

Q
: Gibt den gespeicherten Wert zurück.

¬Q
: Gibt den gespeicherten Wert negiert zurück.

Veränderbare Attribute

Daten-Bits
: Anzahl der Daten-Bits, die verwendet werden.

Bezeichnung
: Die Bezeichnung dieses Elementes.

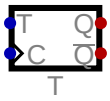
inverse Eingänge
: Es können die Eingänge ausgewählt werden, welche invertiert werden sollen.

Rotation
: Legt die Ausrichtung des Elementes in der Schaltung fest.

Spiegeln
: Spiegelt das Element in der Schaltung.

Vorgabe
: Dieser Wert wird beim Schaltungsstart gesetzt. Beim Demultiplexer wird dieser Wert für die nicht gewählten Ausgänge gesetzt.

Als Messwert verwenden
: Wenn gesetzt, taucht der Wert als Messwert in Graph und Tabelle auf. Zusätzlich muss eine Bezeichnung angegeben werden, welche als Identifikation des Messwertes dienen kann.



8.5. T-FlipFlop

Speichert ein Bit, welches sich mit einem Eingang umschalten lässt.

Eingänge

T
: Aktiviert die Umschaltfunktion.

C
: Dieser Eingang schaltet das Flipflop um. Wenn ein T-Eingang vorhanden ist, passiert das nur, wenn T auf 1 gesetzt ist.

Ausgänge

Q
: Gibt den gespeicherten Wert aus.

¬Q
: Gibt den gespeicherten Wert negiert aus.

Veränderbare Attribute

Bezeichnung
: Die Bezeichnung dieses Elementes.

Enable Eingang
: Wenn gesetzt, ist ein Enable-Eingang (T) vorhanden.

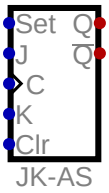
inverse Eingänge
: Es können die Eingänge ausgewählt werden, welche invertiert werden sollen.

Rotation
: Legt die Ausrichtung des Elementes in der Schaltung fest.

Spiegeln
: Spiegelt das Element in der Schaltung.

Vorgabe
: Dieser Wert wird beim Schaltungsstart gesetzt. Beim Demultiplexer wird dieser Wert für die nicht gewählten Ausgänge gesetzt.

Als Messwert verwenden
: Wenn gesetzt, taucht der Wert als Messwert in Graph und Tabelle auf. Zusätzlich muss eine Bezeichnung angegeben werden, welche als Identifikation des Messwertes dienen kann.



8.6. JK-FlipFlop, asynchron

Bietet die Funktionen zum Speichern (J=K=0), Setzen (J=1, K=0), Rücksetzen (J=0, K=1) und Wechseln (J=K=1). Ein Zustandswechsel findet nur bei einer steigenden Flanke am Eingang C statt. Es gibt zwei Eingänge, welche ein sofortiges Setzen bzw. Rücksetzen ohne Taktsignal erlauben. Exportierbar zu VHDL/Verilog.

Eingänge

Set
: asynchrones Setzen. Eine Eins an diesem Eingang setzt das Flipflop auf Eins.

J
: Der Setzen-Eingang des Flipflops.

C
: Takteingang. Eine steigende Flanke initiiert den Zustandswechsel.

K
: Der Rücksetzen-Eingang des Flipflops.

Clr
: asynchrones Löschen. Eine Eins an diesem Eingang setzt das Flipflop auf Null.

Ausgänge

Q
: Gibt den gespeicherten Wert aus.

¬Q
: Gibt den gespeicherten Wert negiert aus.

Veränderbare Attribute

Bezeichnung
: Die Bezeichnung dieses Elementes.

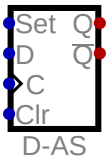
inverse Eingänge
: Es können die Eingänge ausgewählt werden, welche invertiert werden sollen.

Rotation
: Legt die Ausrichtung des Elementes in der Schaltung fest.

Spiegeln
: Spiegelt das Element in der Schaltung.

Vorgabe
: Dieser Wert wird beim Schaltungsstart gesetzt. Beim Demultiplexer wird dieser Wert für die nicht gewählten Ausgänge gesetzt.

Als Messwert verwenden
: Wenn gesetzt, taucht der Wert als Messwert in Graph und Tabelle auf. Zusätzlich muss eine Bezeichnung angegeben werden, welche als Identifikation des Messwertes dienen kann.



8.7. D-FlipFlop, asynchron

Ein Baustein zum Speichern eines Bits. Der an Eingang D anliegende Wert wird bei einer steigenden Flanke an Eingang C gespeichert. Es gibt zwei Eingänge, welche ein sofortiges Setzen bzw. Rücksetzen ohne Taktsignal erlauben. Die Bitbreite kann gewählt werden, so dass mehrere Bits gespeichert werden können. Exportierbar zu VHDL/Verilog.

Eingänge

Set
: asynchrones Setzen. Eine Eins an diesem Eingang setzt alle gespeicherten Bits auf Eins.

D
: Das zu speichernde Bit.

C
: Takt des Flipflops. Wechselt dieser Wert auf 1, wird der an D anliegende Wert abgespeichert.

Clr
: asynchrones Löschen. Eine Eins an diesem Eingang setzt alle gespeicherten Bits auf Null.

Ausgänge

Q
: Gibt den gespeicherten Wert zurück.

¬Q
: Gibt den gespeicherten Wert negiert zurück.

Veränderbare Attribute

Daten-Bits
: Anzahl der Daten-Bits, die verwendet werden.

Bezeichnung
: Die Bezeichnung dieses Elementes.

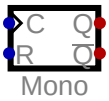
inverse Eingänge
: Es können die Eingänge ausgewählt werden, welche invertiert werden sollen.

Rotation
: Legt die Ausrichtung des Elementes in der Schaltung fest.

Spiegeln
: Spiegelt das Element in der Schaltung.

Vorgabe
: Dieser Wert wird beim Schaltungsstart gesetzt. Beim Demultiplexer wird dieser Wert für die nicht gewählten Ausgänge gesetzt.

Als Messwert verwenden
: Wenn gesetzt, taucht der Wert als Messwert in Graph und Tabelle auf. Zusätzlich muss eine Bezeichnung angegeben werden, welche als Identifikation des Messwertes dienen kann.



8.8. Monoflop

Ein Monoflop wird bei einer steigenden Flanke am Takteingang gesetzt. Nach einer einstellbaren Verzögerungszeit wird das Monoflop automatisch wieder gelöscht. Das Monoflop ist retriggerbar. Es kann nur genutzt werden, wenn es genau ein Taktelement in der Schaltung gibt. Dieses Taktelement wird dann für die Zeitbestimmung verwendet.

Eingänge

C
: Eine steigende Flanke an diesem Eingang setzt das Monoflop. Der Ausgang Q wird auf Eins gesetzt. Nach Ablauf der Verzögerungszeit fällt Q wieder auf Null zurück.

R
: Reset Eingang. Eine Eins setzt das Monoflop auf Null zurück.

Ausgänge

Q
: Ausgang des Monoflops

¬Q
: Invertierter Ausgang

Veränderbare Attribute

Bezeichnung
: Die Bezeichnung dieses Elementes.

Impulsdauer
: Die Impulsdauer wird in Taktzyklen angegeben.

inverse Eingänge
: Es können die Eingänge ausgewählt werden, welche invertiert werden sollen.

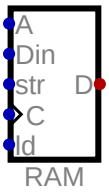
Rotation
: Legt die Ausrichtung des Elementes in der Schaltung fest.

Spiegeln
: Spiegelt das Element in der Schaltung.

Vorgabe
: Dieser Wert wird beim Schaltungsstart gesetzt. Beim Demultiplexer wird dieser Wert für die nicht gewählten Ausgänge gesetzt.

Als Messwert verwenden
: Wenn gesetzt, taucht der Wert als Messwert in Graph und Tabelle auf. Zusätzlich muss eine Bezeichnung angegeben werden, welche als Identifikation des Messwertes dienen kann.

9. Speicher - RAM



9.1. RAM, getrennte Ports

Ein RAM Modul mit getrennten Daten-Anschlüssen für Lesen und Schreiben. Es gibt einen Eingang für das Beschreiben und einen Ausgang für das Auslesen der gespeicherten Daten. Exportierbar zu VHDL/Verilog.

Eingänge

A
: Die Adresse, an der gelesen bzw. geschrieben wird.

Din
: Die Daten, die gespeichert werden sollen.

str
: Ist diese Leitung high, wird das Datenwort gespeichert, wenn der Takt ansteigt.

C
: Der Takt. Eine steigende Flanke aktiviert das Speichern.

ld
: Ist diese Leitung high, wird der Ausgang aktiviert, und die Daten liegen dort an.

Ausgänge

D
: Ausgabe der gespeicherten Daten.

Veränderbare Attribute

Daten-Bits
: Anzahl der Daten-Bits, die verwendet werden.

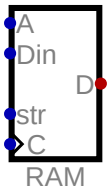
Adress-Bits
: Anzahl der Adress-Bits, die verwendet werden.

Bezeichnung
: Die Bezeichnung dieses Elementes.

Rotation
: Legt die Ausrichtung des Elementes in der Schaltung fest.

Zahlenformat
: Das Zahlenformat für die Anzeige der Werte.

Programmspeicher
: Zeichnet dieses ROM als Programmspeicher aus. Es kann damit von einer externen IDE beschrieben werden.



9.2. Block-RAM, getrennte Ports

Ein RAM Modul mit getrennten Daten-Anschlüssen für Lesen und Schreiben. Es gibt einen Eingang für das Beschreiben und einen Ausgang für das Auslesen der gespeicherten Daten. Dieser RAM-Baustein aktualisiert seinen Ausgang nur bei einer steigenden Flanke an der Clock. Das erlaubt die Verwendung von Block-RAM in einem FPGA. Exportierbar zu VHDL/ Verilog.

Eingänge

A
: Die Adresse, an der gelesen bzw. geschrieben wird.

Din
: Die Daten, die gespeichert werden sollen.

str
: Ist diese Leitung high, wird das Datenwort gespeichert, wenn der Takt ansteigt.

C
: Der Takt. A rising edge activates storing and reading.

Ausgänge

D
: Ausgabe der gespeicherten Daten.

Veränderbare Attribute

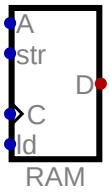
Daten-Bits
: Anzahl der Daten-Bits, die verwendet werden.

Adress-Bits
: Anzahl der Adress-Bits, die verwendet werden.

Bezeichnung
: Die Bezeichnung dieses Elementes.

Rotation
: Legt die Ausrichtung des Elementes in der Schaltung fest.

Programmspeicher
: Zeichnet dieses ROM als Programmspeicher aus. Es kann damit von einer externen IDE beschrieben werden.



9.3. RAM, bidirektionaler Port

Ein RAM Module mit einem bidirektionalem Anschluss für das Lesen und Schreiben von Daten.

Eingänge

A
: Die Adresse, an der gelesen und geschrieben wird.

str
: Ist dieser Eingang 1, wird mit steigendem Takt das Datenwort gespeichert.

C
: Der Takt. Eine steigende Flanke aktiviert das Speichern.

ld
: Ist dieser Eingang 1, wird das Datenwort ausgegeben.

Ausgänge

D
: Der bidirektionale Datenanschluss.

Veränderbare Attribute

Daten-Bits
: Anzahl der Daten-Bits, die verwendet werden.

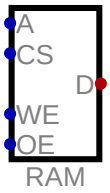
Adress-Bits
: Anzahl der Adress-Bits, die verwendet werden.

Bezeichnung
: Die Bezeichnung dieses Elementes.

Rotation
: Legt die Ausrichtung des Elementes in der Schaltung fest.

Zahlenformat
: Das Zahlenformat für die Anzeige der Werte.

Programmspeicher
: Zeichnet dieses ROM als Programmspeicher aus. Es kann damit von einer externen IDE beschrieben werden.



9.4. RAM, Chip Select

Ein RAM-Baustein mit einem bidirektionalem Anschluss für das Lesen und Schreiben von Daten. Es gibt einen CS-Eingang. Ist dieser Eingang low, ist der Baustein deaktiviert. Mit diesem können mehrere solcher Bausteine mit einem Adressdekoder zu einem größeren RAM zusammen geschaltet werden. Der Schreibzyklus läuft wie folgt ab: Indem CS auf high gesetzt wird, wird der Baustein ausgewählt. Eine steigende Flanke an WE übernimmt die Adresse, und die folgende fallende Flanke an WE speichert die Daten.

Eingänge

A
: Die Adresse, an der gelesen und geschrieben wird.

CS
: Ist dieser Eingang 1, ist der Baustein aktiv.

WE
: Bei einer 1 werden die Daten in das RAM geschrieben.

OE
: Ist dieser Eingang 1, wird das Datenwort ausgegeben.

Ausgänge

D
: Der bidirektionale Datenanschluss.

Veränderbare Attribute

Daten-Bits
: Anzahl der Daten-Bits, die verwendet werden.

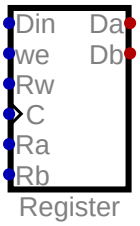
Adress-Bits
: Anzahl der Adress-Bits, die verwendet werden.

Bezeichnung
: Die Bezeichnung dieses Elementes.

inverse Eingänge
: Es können die Eingänge ausgewählt werden, welche invertiert werden sollen.

Rotation
: Legt die Ausrichtung des Elementes in der Schaltung fest.

Programmspeicher
: Zeichnet dieses ROM als Programmspeicher aus. Es kann damit von einer externen IDE beschrieben werden.



9.5. Registerspeicher

Speicher mit einem Schreib- und zwei Leseports. Kann verwendet werden um Prozessorregister zu implementieren. Es können so gleichzeitig zwei Register gelesen und ein Drittes beschrieben werden. Exportierbar zu VHDL/Verilog.

Eingänge

Din
: Die zu schreibenden Daten.

we
: Bei einer 1 werden die Daten in das Register Rw übernommen.

Rw
: Nummer des zu beschreibenden Registers.

C
: Takteingang

Ra
: Nummer des Registers a.

Rb
: Nummer des Registers b.

Ausgänge

Da
: Inhalt des Registers a.

Db
: Inhalt des Registers b.

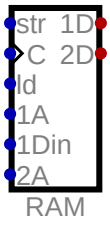
Veränderbare Attribute

Daten-Bits
: Anzahl der Daten-Bits, die verwendet werden.

Adress-Bits
: Anzahl der Adress-Bits, die verwendet werden.

Bezeichnung
: Die Bezeichnung dieses Elementes.

Rotation
: Legt die Ausrichtung des Elementes in der Schaltung fest.



9.6. RAM, Dual Port

RAM mit einem Port, der das Beschreiben und Lesen des RAMs ermöglicht, und einem zweiten Leseport. Dieser zweite Port kann verwendet werden, um einer Grafik-Logik Zugriff auf den Speicherinhalt zu geben. Auf diese Weise kann ein Prozessor in das RAM schreiben, und eine Grafik-Logik kann das RAM gleichzeitig auslesen. Exportierbar zu VHDL/Verilog.

Eingänge

str
: Ist diese Leitung high, wird das Datenwort an D1 gespeichert, wenn der Takt ansteigt.

C
: Der Takt. Eine steigende Flanke aktiviert das Speichern.

ld
: Ist diese Leitung high, wird der Ausgang D1 aktiviert, und die Daten liegen dort an.

1A
: Die Adresse, an der über Port 1 gelesen bzw. geschrieben wird.

1Din
: Die Daten, die gespeichert werden sollen.

2A
: Die Adresse, an der über Port 2 gelesen wird.

Ausgänge

1D
: Ausgabeport 1

2D
: Ausgabeport 2

Veränderbare Attribute

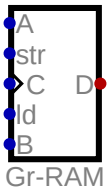
Daten-Bits
: Anzahl der Daten-Bits, die verwendet werden.

Adress-Bits
: Anzahl der Adress-Bits, die verwendet werden.

Bezeichnung
: Die Bezeichnung dieses Elementes.

Rotation
: Legt die Ausrichtung des Elementes in der Schaltung fest.

Programmspeicher
: Zeichnet dieses ROM als Programmspeicher aus. Es kann damit von einer externen IDE beschrieben werden.



9.7. Grafik-RAM

Wird verwendet, um Bitmap Grafiken anzuzeigen. Der Baustein verhält sich wie ein RAM-Baustein mit dem Unterschied, dass der RAM-Inhalt als Grafik angezeigt wird. Jede Speicherstelle definiert die Farbe eines Pixels, wobei eine feste Farbpalette verwendet wird. Es werden zwei Grafikseiten angelegt, sodass ein Screen-Switching möglich ist (siehe Eingang B). Die gesamte Speichergröße beträgt damit damit dx*dy*2 Speicherworte. Die verwendete Palette ist wie folgt aufgebaut: Die Indizes 0-9 entsprechen den Farben Weiß, Schwarz, Rot, Grün, Blau, Gelb, Türkis, Magenta, Orange und Pink. Die Indizes 32-63 bilden Grauwerte ab und die Indizes 64-127 repräsentieren 64 Farbwerte mit je zwei Bit pro Farbkanal. Dadurch ergibt sich eine einfache Palette, die mit nur 7-Bit angesprochen werden kann. Wird von der Architektur ein 16-Bit Index unterstützt, kann ab Index 0x8000 ein High-Color-Modus mit 5 Bit pro Farbkanal verwendet werden, welcher 32768 Farben ermöglicht.

Eingänge

A
: Die Adresse, an der gelesen und geschrieben wird.

str
: Ist dieser Eingang 1, wird mit steigendem Takt das Datenwort gespeichert.

C
: Der Takt. Eine steigende Flanke aktiviert das Speichern.

ld
: Ist dieser Eingang 1, wird das Datenwort ausgegeben.

B
: Auswahl der anzuzeigenden Seite. Mit diesem Eingang kann zwischen zwei Speicherseiten umgeschaltet werden.

Ausgänge

D
: Der bidirektionale Datenanschluss.

Veränderbare Attribute

Daten-Bits
: Anzahl der Daten-Bits, die verwendet werden.

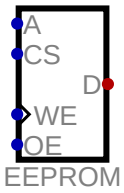
Bezeichnung
: Die Bezeichnung dieses Elementes.

Breite in Pixel
: Breite des Grafik-Bildschirmes in Pixel

Höhe in Pixel
: Höhe des Grafik-Bildschirmes in Pixel

Rotation
: Legt die Ausrichtung des Elementes in der Schaltung fest.

10. Speicher - EEPROM



10.1. EEPROM

Ein EEPROM-Baustein mit einem bidirektionalem Anschluss für das Lesen und Schreiben von Daten. Es gibt einen CS-Eingang. Ist dieser Eingang low, ist der Baustein deaktiviert. Der Dateninhalt wird gespeichert wie bei einem ROM. Er bleibt also erhalten, wenn die Simulation beendet und neu gestartet wird. Der Schreibzyklus läuft wie folgt ab: Indem CS auf high gesetzt wird, wird der Baustein ausgewählt. Eine steigende Flanke an WE übernimmt die Adresse, und die folgende fallende Flanke an WE speichert die Daten.

Eingänge

A
: Die Adresse, an der gelesen und geschrieben wird.

CS
: Ist dieser Eingang 1, ist der Baustein aktiv.

WE
: Bei einer 1 werden die Daten in das EEPROM geschrieben.

OE
: Ist dieser Eingang 1, wird das Datenwort ausgegeben.

Ausgänge

D
: Der bidirektionale Datenanschluss.

Veränderbare Attribute

Daten-Bits
: Anzahl der Daten-Bits, die verwendet werden.

Adress-Bits
: Anzahl der Adress-Bits, die verwendet werden.

Bezeichnung
: Die Bezeichnung dieses Elementes.

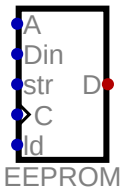
inverse Eingänge
: Es können die Eingänge ausgewählt werden, welche invertiert werden sollen.

Daten
: Die Daten, welche in diesem Element gespeichert sind.

Rotation
: Legt die Ausrichtung des Elementes in der Schaltung fest.

Zahlenformat
: Das Zahlenformat für die Anzeige der Werte.

Programmspeicher
: Zeichnet dieses ROM als Programmspeicher aus. Es kann damit von einer externen IDE beschrieben werden.



10.2. EEPROM, getrennte Ports

Ein EEPROM Modul mit getrennten Daten-Anschlüssen für Lesen und Schreiben. Es gibt einen Eingang für das Beschreiben und einen Ausgang für das Auslesen der gespeicherten Daten.

Eingänge

A
: Die Adresse, an der gelesen bzw. geschrieben wird.

Din
: Die Daten, die gespeichert werden sollen.

str
: Ist diese Leitung high, wird das Datenwort gespeichert, wenn der Takt ansteigt.

C
: Der Takt. Eine steigende Flanke aktiviert das Speichern.

ld
: Ist diese Leitung high, wird der Ausgang aktiviert, und die Daten liegen dort an.

Ausgänge

D
: Ausgabe der gespeicherten Daten.

Veränderbare Attribute

Daten-Bits
: Anzahl der Daten-Bits, die verwendet werden.

Adress-Bits
: Anzahl der Adress-Bits, die verwendet werden.

Bezeichnung
: Die Bezeichnung dieses Elementes.

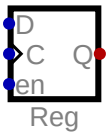
Daten
: Die Daten, welche in diesem Element gespeichert sind.

Rotation
: Legt die Ausrichtung des Elementes in der Schaltung fest.

Zahlenformat
: Das Zahlenformat für die Anzeige der Werte.

Programmspeicher
: Zeichnet dieses ROM als Programmspeicher aus. Es kann damit von einer externen IDE beschrieben werden.

11. Speicher



11.1. Register

Ein Baustein zum Speichern von Werten. Die Bitbreite kann gewählt werden. Der an Eingang D anliegende Wert wird bei einer steigenden Flanke an Eingang C gespeichert. Im Unterschied zu einem D Flipflop verfügt das Register über einen Freigabeeingang für den Takt. Exportierbar zu VHDL/Verilog.

Eingänge

D
: Das zu speichernde Datenwort liegt hier an.

C
: Der Takteingang. Eine steigende Flanke speichert den an D anliegenden Wert ab.

en
: Nur wenn dieser Eingang 1 ist, kann gespeichert werden.

Ausgänge

Q
: Gibt den abgespeicherten Wert aus.

Veränderbare Attribute

Daten-Bits
: Anzahl der Daten-Bits, die verwendet werden.

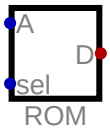
Bezeichnung
: Die Bezeichnung dieses Elementes.

inverse Eingänge
: Es können die Eingänge ausgewählt werden, welche invertiert werden sollen.

Rotation
: Legt die Ausrichtung des Elementes in der Schaltung fest.

Programmzähler
: Weist dieses Register als Programmzähler aus. Der Wert dieses Registers wird an die externe Assembler-IDE zurückgegeben, um während des Debuggings die aktuelle Codezeile zu markieren.

Als Messwert verwenden
: Wenn gesetzt, taucht der Wert als Messwert in Graph und Tabelle auf. Zusätzlich muss eine Bezeichnung angegeben werden, welche als Identifikation des Messwertes dienen kann.



11.2. ROM

Ein nichtflüchtiger Speicherbaustein welcher einen zur Simulationszeit festen Inhalt hat. Die gespeicherten Daten können im Attribute-Dialog bearbeitet werden. Exportierbar zu VHDL/ Verilog.

Eingänge

A
: Dieser Eingang bestimmt die Speicheradresse des Datenwortes, welches ausgegeben werden soll.

sel
: Ist dieser Pin high (1), ist der Ausgang aktiviert. Ist er low (0), ist der Ausgang hochohmig.

Ausgänge

D
: Hier wird das Datenwort ausgegeben, wenn am "Select"-Eingang eine 1 anliegt.

Veränderbare Attribute

Daten-Bits
: Anzahl der Daten-Bits, die verwendet werden.

Adress-Bits
: Anzahl der Adress-Bits, die verwendet werden.

Bezeichnung
: Die Bezeichnung dieses Elementes.

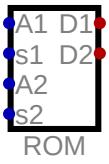
Daten
: Die Daten, welche in diesem Element gespeichert sind.

Rotation
: Legt die Ausrichtung des Elementes in der Schaltung fest.

Zahlenformat
: Das Zahlenformat für die Anzeige der Werte.

Programmspeicher
: Zeichnet dieses ROM als Programmspeicher aus. Es kann damit von einer externen IDE beschrieben werden.

Bei jedem Start automatisch neu laden.
: Lädt das HEX-File bei jedem Modelstart neu.



11.3. ROM Dual Port

Ein nichtflüchtiger Speicherbaustein welcher einen zur Simulationszeit festen Inhalt hat. Die gespeicherten Daten können im Attribute-Dialog bearbeitet werden. Im Gegensatz zum einfachen ROM verfügt dieser Baustein über zwei Leseports.

Eingänge

A1
: Dieser Eingang bestimmt die Speicheradresse des Datenwortes, welches and Port 1 ausgegeben werden soll.

s1
: Ist dieser Pin high (1), ist der Ausgang 1 aktiviert. Ist er low (0), ist der Ausgang 1 hochohmig.

A2
: Dieser Eingang bestimmt die Speicheradresse des Datenwortes, welches and Port 2 ausgegeben werden soll.

s2
: Ist dieser Pin high (1), ist der Ausgang 2 aktiviert. Ist er low (0), ist der Ausgang 2 hochohmig.

Ausgänge

D1
: Hier wird das Datenwort ausgegeben, wenn am "Select"-Eingang 1 eine 1 anliegt.

D2
: Hier wird das Datenwort ausgegeben, wenn am "Select"-Eingang 2 eine 1 anliegt.

Veränderbare Attribute

Daten-Bits
: Anzahl der Daten-Bits, die verwendet werden.

Adress-Bits
: Anzahl der Adress-Bits, die verwendet werden.

Bezeichnung
: Die Bezeichnung dieses Elementes.

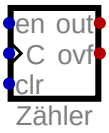
Daten
: Die Daten, welche in diesem Element gespeichert sind.

Rotation
: Legt die Ausrichtung des Elementes in der Schaltung fest.

Zahlenformat
: Das Zahlenformat für die Anzeige der Werte.

Programmspeicher
: Zeichnet dieses ROM als Programmspeicher aus. Es kann damit von einer externen IDE beschrieben werden.

Bei jedem Start automatisch neu laden.
: Lädt das HEX-File bei jedem Modelstart neu.



11.4. Zähler

Ein einfacher Zähler-Baustein. Zählt jede steigende Flanke am C Eingang und kann über den clr Eingang zurückgesetzt werden. Die Bitbreite des Zählers kann im Attribute-Dialog festgelegt werden. Exportierbar zu VHDL/Verilog.

Eingänge

en
: Der Zähler zählt nur, wenn dieser Eingang auf 1 gesetzt ist.

C
: Eingang des Taktsignals.

clr
: Setzt den Zähler synchron auf 0 zurück, wenn dieser Eingang auf 1 gesetzt wird.

Ausgänge

out
: Gibt den gezählten Wert aus.

ovf
: Overflow Ausgang. Wird auf 1 gesetzt, wenn der Zähler seinen Maximalwert hat und der en Eingang auf 1 gesetzt ist.

Veränderbare Attribute

Daten-Bits
: Anzahl der Daten-Bits, die verwendet werden.

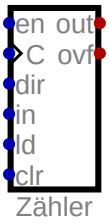
inverse Eingänge
: Es können die Eingänge ausgewählt werden, welche invertiert werden sollen.

Bezeichnung
: Die Bezeichnung dieses Elementes.

Rotation
: Legt die Ausrichtung des Elementes in der Schaltung fest.

Als Messwert verwenden
: Wenn gesetzt, taucht der Wert als Messwert in Graph und Tabelle auf. Zusätzlich muss eine Bezeichnung angegeben werden, welche als Identifikation des Messwertes dienen kann.

Programmzähler
: Weist dieses Register als Programmzähler aus. Der Wert dieses Registers wird an die externe Assembler-IDE zurückgegeben, um während des Debuggings die aktuelle Codezeile zu markieren.



11.5. Zähler, beschreibbar

Zähler, dessen Wert gesetzt werden kann. Zudem kann ein Maximalwert und eine Zählrichtung vorgegeben werden. Exportierbar zu VHDL/Verilog.

Eingänge

- en
  Der Zähler zählt nur, wenn dieser Eingang auf 1 gesetzt ist.
- C
  Eingang des Taktsignals.
- dir
  Gibt die Zählrichtung an. Eine 0 bedeutet aufwärts.
- in
  Dieses Datenwort wird im Zähler gespeichert, wenn ld=1 gesetzt ist.
- ld
  Wenn gesetzt wird beim nächsten Taktsignal der Wert am Eingang 'in' in den Zähler übernommen.
- clr
  Setzt den Zähler synchron auf 0 zurück, wenn dieser Eingang auf 1 gesetzt wird.

Ausgänge

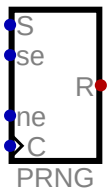
- out
  Gibt den gezählten Wert aus.
- ovf
  Overflow Ausgang. Wird auf 1 gesetzt, der 'en' Eingang auf 1 gesetzt ist und wenn der Zähler beim Aufwärtszählen seinen Maximalwert erreicht, bzw. beim Abwärtszählen die 0 erreicht hat.

Veränderbare Attribute

- Daten-Bits
  Anzahl der Daten-Bits, die verwendet werden.
- Maximalwert
  Wird hier eine Null eingetragen, wird der maximal mögliche Wert verwendet (Alle Bits sind Eins).
- inverse Eingänge
  Es können die Eingänge ausgewählt werden, welche invertiert werden sollen.
- Bezeichnung
  Die Bezeichnung dieses Elementes.
- Rotation
  Legt die Ausrichtung des Elementes in der Schaltung fest.

Als Messwert verwenden
: Wenn gesetzt, taucht der Wert als Messwert in Graph und Tabelle auf. Zusätzlich muss eine Bezeichnung angegeben werden, welche als Identifikation des Messwertes dienen kann.

Programmzähler
: Weist dieses Register als Programmzähler aus. Der Wert dieses Registers wird an die externe Assembler-IDE zurückgegeben, um während des Debuggings die aktuelle Codezeile zu markieren.



11.6. Zufallszahlengenerator

Kann verwendet werden um Zufallszahlen zu erzeugen. Beim Starten der Simulation wird der Generator neu initialisiert, so dass bei jedem Start eine neue Pseudozufallszahlenfolge erzeugt wird. Der Generator kann in der laufenden Simulation mit einem definierten SEED-Wert initialisiert werden, um eine definierte Pseudozufallszahlenfolge erzeugen zu lassen.

Eingänge

S
: Startwert des Generators

se
: Wenn gesetzt wird der Zufallsszahlengenerator bei der nächsten steigenden Taktflanke mit dem neuen Startwert reinitialisiert.

ne
: Wenn gesetzt wird bei der nächsten steigenden Taktflanke eine neue Zufallsszahl ausgegeben.

C
: Der Takteingang.

Ausgänge

R
: Ausgabe der Pseudozufallszahl.

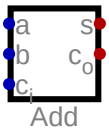
Veränderbare Attribute

Daten-Bits
: Anzahl der Daten-Bits, die verwendet werden.

Bezeichnung
: Die Bezeichnung dieses Elementes.

Rotation
: Legt die Ausrichtung des Elementes in der Schaltung fest.

12. Arithmetik



12.1. Addierer

Ein Bauteil für einfache Additionen. Führt eine Addition der Ganzzahlen an Eingang a und Eingang b durch (a+b). Ist der Carry-Eingang gesetzt, wird das Ergebnis um Eins erhöht. Exportierbar zu VHDL/Verilog.

Eingänge

a
: Erster Eingang für die Addition.

b
: Zweiter Eingang für die Addition.

c_i
: Carry-Eingang. Wenn gesetzt, wird das Ergebnis um Eins erhöht. Kann von vorhergehenden Bausteinen über den Carry-Out Ausgang gesetzt werden.

Ausgänge

s
: Das Ergebnis der Addition

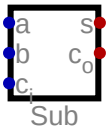
c_o
: Ist gesetzt, wenn bei der Addition ein Übertrag aufgetreten ist.

Veränderbare Attribute

Bezeichnung
: Die Bezeichnung dieses Elementes.

Daten-Bits
: Anzahl der Daten-Bits, die verwendet werden.

Rotation
: Legt die Ausrichtung des Elementes in der Schaltung fest.



12.2. Subtrahierer

Ein Bauteil für einfache Subtraktionen. Führt eine Subtraktion der Ganzzahlen an Eingang a und Eingang b durch (a-b). Ist der Carry-Eingang gesetzt, wird das Ergebnis nochmals um 1 verringert. Exportierbar zu VHDL/Verilog.

Eingänge

a
: Eingang a für die Subtraktion.

b
: Eingang b für die Subtraktion.

c_i
: Carry-Eingang. Wenn gesetzt wird das Ergebnis um Eins verringert.

Ausgänge

s
: Ausgang gibt das Ergebnis der Subtraktion aus.

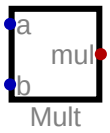
c_o
: Carry-Ausgang. Gibt 1 aus, wenn bei der Subtraktion ein Überlauf aufgetreten ist.

Veränderbare Attribute

Bezeichnung
: Die Bezeichnung dieses Elementes.

Daten-Bits
: Anzahl der Daten-Bits, die verwendet werden.

Rotation
: Legt die Ausrichtung des Elementes in der Schaltung fest.



12.3. Multiplizierer

Ein Baustein für Multiplikation. Multipliziert die an Eingang a und Eingang b anliegenden Ganzzahlen. Exportierbar zu VHDL/Verilog.

Eingänge

a
: Eingang a für Multiplikation.

b
: Eingang b für Multiplikation.

Ausgänge

mul
: Ausgang mit dem Ergebnis der Multiplikation.

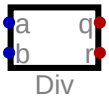
Veränderbare Attribute

Bezeichnung
: Die Bezeichnung dieses Elementes.

Operation mit Vorzeichen
: Wenn gesetzt, findet die Operation vorzeichenrichtig (2-er Komplement) statt.

Daten-Bits
: Anzahl der Daten-Bits, die verwendet werden.

Rotation
: Legt die Ausrichtung des Elementes in der Schaltung fest.



12.4. Dividierer

Ein Baustein für Divisionen. Dividiert die an Eingang a anliegende Ganzzahl durch die an Eingang b anliegenden Ganzzahl. Ist der Divisor Null, wird stattdessen durch Eins geteilt. Bei vorzeichenbehafteter Division ist der Rest der Division immer positiv.

Eingänge

a
: Der Dividend

b
: Der Divisor

Ausgänge

q
: Der ganzzahlige Quotient.

r
: Der Rest der Division.

Veränderbare Attribute

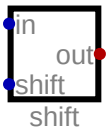
Bezeichnung
: Die Bezeichnung dieses Elementes.

Daten-Bits
: Anzahl der Daten-Bits, die verwendet werden.

Operation mit Vorzeichen
: Wenn gesetzt, findet die Operation vorzeichenrichtig (2-er Komplement) statt.

Rest immer positiv
: Wenn gesetzt, ist der Rest einer vorzeichenbehafteten Division immer positiv.

Rotation
: Legt die Ausrichtung des Elementes in der Schaltung fest.



12.5. Bit-Schieber

Ein Baustein zum Schieben von Bits. Verschiebt einen Wert um die am Eingang shift angegebene Anzahl von Bits.

Eingänge

in
: Eingang mit zu verschiebenden Bits.

shift
: Eingang mit Weite der Verschiebung.

Ausgänge

out
: Ausgang mit dem Ergebnis der Verschiebeoperation.

Veränderbare Attribute

Bezeichnung
: Die Bezeichnung dieses Elementes.

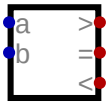
Daten-Bits
: Anzahl der Daten-Bits, die verwendet werden.

Verschiebeweite hat Vorzeichen
: Verschiebeweite verwendet Zweierkomplement

Richtung
: Richtungsangabe.

Modus
: Modus der Verschiebung.

Rotation
: Legt die Ausrichtung des Elementes in der Schaltung fest.



12.6. Komparator

Ein Baustein zum Vergleichen von Datenworten. Vergleicht die Datenworte an Eingang a und Eingang b und setzt die Ausgänge entsprechend. Exportierbar zu VHDL/Verilog.

Eingänge

a
: Eingang a für den Vergleich.

b
: Eingang b für den Vergleich.

Ausgänge

>
: Ausgang ist 1, wenn Eingang a größer ist als Eingang b.

=
: Ausgang ist 1, wenn die Werte an beiden Eingängen gleich sind.

<
: Ausgang ist 1, wenn Eingang a kleiner ist als Eingang b

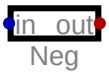
Veränderbare Attribute

Bezeichnung
: Die Bezeichnung dieses Elementes.

Daten-Bits
: Anzahl der Daten-Bits, die verwendet werden.

Operation mit Vorzeichen
: Wenn gesetzt, findet die Operation vorzeichenrichtig (2-er Komplement) statt.

Rotation
: Legt die Ausrichtung des Elementes in der Schaltung fest.



12.7. Negation

Ein Baustein für die Negation von Datenworten im 2-er Komplement. Exportierbar zu VHDL/ Verilog.

Eingänge

in
: Eingang des Datenworts, welches im 2-er Komplement negiert werden soll.

Ausgänge

out
: Gibt das Ergebnis der Negation im 2-er Komplement zurück.

Veränderbare Attribute

Daten-Bits
: Anzahl der Daten-Bits, die verwendet werden.

Rotation
: Legt die Ausrichtung des Elementes in der Schaltung fest.

12.8. Biterweiterung

Erweitert die Bitanzahl eines vorzeichenbehafteten Eingangswertes unter Erhalt des Vorzeichens. Ist der Eingang ein einzelnes Bit, wird dieses Bit auf allen Ausgangsbits ausgegeben. Exportierbar zu VHDL/Verilog.

Eingänge

in
: Eingangswert. Die Eingangsbitzahl muss kleiner sein als die Ausgangsbitzahl!

Ausgänge

out
: Vorzeichenrichtig erweiterter Eingangswert. Die Eingangsbitzahl muss kleiner sein als die Ausgangsbitzahl!

Veränderbare Attribute

Bezeichnung
: Die Bezeichnung dieses Elementes.

Anzahl Eingangsbits
: Die Anzahl der Ausgangsbits muss größer sein als die Anzahl der Eingangsbits.

Anzahl Ausgangsbits
: Die Anzahl der Ausgangsbits muss größer sein als die Anzahl der Eingangsbits.

Rotation
: Legt die Ausrichtung des Elementes in der Schaltung fest.

12.9. Bitzähler

Gibt die Anzahl der 1-Bits im Eingangswert aus.

Eingänge

in
: Die 1-Bits in diesem Datenwort werden gezählt.

Ausgänge

out
: Ausgang mit der Anzahl der gezählten 1-Bits.

Veränderbare Attribute

Daten-Bits
: Anzahl der Daten-Bits, die verwendet werden.

Rotation
: Legt die Ausrichtung des Elementes in der Schaltung fest.

13. Schalter

13.1. Schalter

Einfacher Schalter. Der Schalter hat keine Gatterlaufzeit. Eine Signaländerung wird instantan von einem Ende des Schalters zum anderen propagiert.

Ausgänge

A1
: Einer der Anschlüsse des Schalters.

B1
: Einer der Anschlüsse des Schalters.

Veränderbare Attribute

Daten-Bits
: Anzahl der Daten-Bits, die verwendet werden.

Bezeichnung
: Die Bezeichnung dieses Elementes.

Pole
: Anzahl der Schaltkontakte des Relais.

Geschlossen
: Gibt an, ob der Schalter bei Modelstart offen oder geschlossen ist.

Rotation
: Legt die Ausrichtung des Elementes in der Schaltung fest.

Spiegeln
: Spiegelt das Element in der Schaltung.

Schalter als Eingang betrachten
: Wird das Model analysiert, wird der Schalter wie ein Eingang betrachtet, wobei "offen" einer '0' und "geschlossen" einer '1' entspricht.

13.2. Wechselschalter

Einfacher Wechselschalter. Der Schalter hat keine Gatterlaufzeit. Eine Signaländerung wird instantan von einem Ende des Schalters zum anderen propagiert.

Ausgänge

A1
: Einer der Anschlüsse des Schalters.

B1
: Einer der Anschlüsse des Schalters.

C1
: Einer der Anschlüsse des Schalters.

Veränderbare Attribute

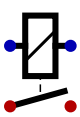
Daten-Bits
: Anzahl der Daten-Bits, die verwendet werden.

Bezeichnung
: Die Bezeichnung dieses Elementes.

Pole
: Anzahl der Schaltkontakte des Relais.

Rotation
: Legt die Ausrichtung des Elementes in der Schaltung fest.

Spiegeln
: Spiegelt das Element in der Schaltung.



13.3. Relais

Ein Relais ist ein Schalter, welcher über eine Spule umgeschaltet werden kann. Wenn ein Strom durch das Relais fließt, wird der Schalter geöffnet bzw. geschlossen. Es gibt keine Freilaufdiode, sodass die Stromrichtung keine Rolle spielt. Der Schalter wird betätigt, wenn

die Eingänge unterschiedliche Werte haben. Das Relais verhält sich damit ähnlich wie ein XOr Gatter.

Eingänge

in1
: Ein Steuereingang des Relais.

in2
: Ein Steuereingang des Relais.

Ausgänge

A1
: Einer der Anschlüsse des Schalters.

B1
: Einer der Anschlüsse des Schalters.

Veränderbare Attribute

Daten-Bits
: Anzahl der Daten-Bits, die verwendet werden.

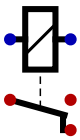
Bezeichnung
: Die Bezeichnung dieses Elementes.

Pole
: Anzahl der Schaltkontakte des Relais.

Relais ist ein Öffner
: Wenn gesetzt, ist das Relais unbestromt geschlossen.

Rotation
: Legt die Ausrichtung des Elementes in der Schaltung fest.

Spiegeln
: Spiegelt das Element in der Schaltung.



13.4. Relais mit Wechselkontakt

Ein Relais ist ein Schalter, welcher über eine Spule umgeschaltet werden kann. Wenn ein Strom durch das Relais fließt, wird der Schalter geöffnet bzw. geschlossen. Es gibt keine Freilaufdiode, sodass die Stromrichtung keine Rolle spielt. Der Schalter wird betätigt, wenn die Eingänge unterschiedliche Werte haben. Das Relais verhält sich damit ähnlich wie ein XOr Gatter.

Eingänge

in1
: Ein Steuereingang des Relais.

in2
: Ein Steuereingang des Relais.

Ausgänge

A1
: Einer der Anschlüsse des Schalters.

B1
: Einer der Anschlüsse des Schalters.

C1
: Einer der Anschlüsse des Schalters.

Veränderbare Attribute

Daten-Bits
: Anzahl der Daten-Bits, die verwendet werden.

Bezeichnung
: Die Bezeichnung dieses Elementes.

Pole
: Anzahl der Schaltkontakte des Relais.

Rotation
: Legt die Ausrichtung des Elementes in der Schaltung fest.

Spiegeln
: Spiegelt das Element in der Schaltung.

13.5. P-Kanal FET

P-Kanal Feldeffekttransistor. Der Bulk ist mit der pos. Versorgung verbunden. Der Transistor wird ohne eine Body-Diode simuliert.

Eingänge

G
: Gate

Ausgänge

S
: Source

D
: Drain

Veränderbare Attribute

Daten-Bits
: Anzahl der Daten-Bits, die verwendet werden.

Unidirektional
: Unidirektionale Transistoren wirken nur von Source zu Drain. Sie können deutlich schneller simuliert werden als bidirektionale Transistoren. Da es keine Rückwirkung von Drain zu Source gibt, können Transistoren in diesem Modus auch keine Kurzschlüsse über die Drain-Source Strecke verursachen. Daher kann dieser Modus in manchen CMOS-Schaltungen erforderlich sein.

Bezeichnung
: Die Bezeichnung dieses Elementes.

Rotation
: Legt die Ausrichtung des Elementes in der Schaltung fest.

Spiegeln
: Spiegelt das Element in der Schaltung.

13.6. N-Kanal FET

N-Kanal Feldeffekttransistor. Der Bulk ist mit Masse verbunden. Der Transistor wird ohne eine Body-Diode simuliert.

Eingänge

G
: Gate

Ausgänge

D
: Drain

S
: Source

Veränderbare Attribute

Daten-Bits
: Anzahl der Daten-Bits, die verwendet werden.

Unidirektional
: Unidirektionale Transistoren wirken nur von Source zu Drain. Sie können deutlich schneller simuliert werden als bidirektionale Transistoren. Da es keine Rückwirkung von Drain zu Source gibt, können Transistoren in diesem Modus auch keine Kurzschlüsse über die Drain-Source Strecke verursachen. Daher kann dieser Modus in manchen CMOS-Schaltungen erforderlich sein.

Bezeichnung
: Die Bezeichnung dieses Elementes.

Rotation
: Legt die Ausrichtung des Elementes in der Schaltung fest.

Spiegeln
: Spiegelt das Element in der Schaltung.

13.7. Fuse

Eine Sicherung, die verwendet werden kann, um einen einmal programmierbaren Speicher aufzubauen.

Ausgänge

out1
: Einer der Anschlüsse des Schalters.

out2
: Einer der Anschlüsse des Schalters.

Veränderbare Attribute

Programmiert
: Wenn gesetzt, ist die Diode "durchgebrannt" bzw "programmiert". Bei Floating Gate FETs wird das Gate geladen. Diese Einstellung kann direkt über die Taste 'P' verändert werden.

Rotation
: Legt die Ausrichtung des Elementes in der Schaltung fest.

13.8. Diode zu Plus

Vereinfachte unidirektionale Diode, die genutzt werden kann, um eine Leitung auf Plus zu ziehen. Wird verwendet, um ein "Wired Or" zu implementieren. Daher ist am Ausgang ein Pull Down Widerstand erforderlich! Innerhalb der Simulation verhält sich eine Diode wie ein aktives Gatter mit dreiwerteiger Wertetabelle: Ist der Eingang 1, ist auch der Ausgang 1. In den anderen Fällen (0 und hochohmig) ist der Ausgang hochohmig (High Z). Damit können sich antiparallel zusammen geschaltete Dioden gegenseitig im high-Zustand halten, was mit realen Dioden nicht möglich ist. Es handelt sich um eine ideale Diode: In Durchlassrichtung gibt es keinen Spannungsabfall über der Diode.

Eingänge

in
: Ist der Eingang 1, ist auch der Ausgang 1. In allen anderen Fällen ist der Ausgang hochohmig.

Ausgänge

out
: Ist der Eingang 1, ist auch der Ausgang 1. In allen anderen Fällen ist der Ausgang hochohmig.

Veränderbare Attribute

Programmiert
: Wenn gesetzt, ist die Diode "durchgebrannt" bzw "programmiert". Bei Floating Gate FETs wird das Gate geladen. Diese Einstellung kann direkt über die Taste 'P' verändert werden.

Rotation
: Legt die Ausrichtung des Elementes in der Schaltung fest.

13.9. Diode zu Masse

Vereinfachte unidirektionale Diode, die genutzt werden kann, um eine Leitung auf Masse zu ziehen. Wird verwendet, um ein "Wired And" zu implementieren. Daher ist am Ausgang ein Pull Up Widerstand erforderlich! Innerhalb der Simulation verhält sich eine Diode wie ein aktives Gatter mit dreiwerteiger Wertetabelle: Ist der Eingang 0, ist auch der Ausgang 0. In den anderen Fällen (1 und hochohmig) ist der Ausgang hochohmig (High Z). Damit können sich antiparallel zusammen geschaltete Dioden gegenseitig im low-Zustand halten, was mit realen Dioden nicht möglich ist. Es handelt sich um eine ideale Diode: In Durchlassrichtung gibt es keinen Spannungabfall über der Diode.

Eingänge

in
: Ist der Eingang 0, ist auch der Ausgang 0. In allen anderen Fällen ist der Ausgang hochohmig.

Ausgänge

out
: Ist der Eingang 0, ist auch der Ausgang 0. In allen anderen Fällen ist der Ausgang hochohmig.

Veränderbare Attribute

Programmiert
: Wenn gesetzt, ist die Diode "durchgebrannt" bzw "programmiert". Bei Floating Gate FETs wird das Gate geladen. Diese Einstellung kann direkt über die Taste 'P' verändert werden.

Rotation
: Legt die Ausrichtung des Elementes in der Schaltung fest.

13.10. P-Kanal Floating Gate FET

P-Kanal Feldeffekttransistor mit Floating Gate. Der Bulk ist mit Masse verbunden. Der Transistor wird ohne eine Body-Diode simuliert. Ist das Floating Gate geladen, ist der Transistor immer sperrend. Die Programmierung kann direkt mit der Taste [P] verändert werden.

Eingänge

G
: Gate

Ausgänge

S
: Source

D
: Drain

Veränderbare Attribute

Daten-Bits
: Anzahl der Daten-Bits, die verwendet werden.

Bezeichnung
: Die Bezeichnung dieses Elementes.

Programmiert
: Wenn gesetzt, ist die Diode "durchgebrannt" bzw "programmiert". Bei Floating Gate FETs wird das Gate geladen. Diese Einstellung kann direkt über die Taste 'P' verändert werden.

Rotation
: Legt die Ausrichtung des Elementes in der Schaltung fest.

Spiegeln
: Spiegelt das Element in der Schaltung.

13.11. N-Kanal Floating Gate FET

N-Kanal Feldeffekttransistor mit Floating Gate. Der Bulk ist mit Masse verbunden. Der Transistor wird ohne eine Body-Diode simuliert. Ist das Floating Gate geladen, ist der Transistor immer sperrend. Die Programmierung kann direkt mit der Taste [P] verändert werden.

Eingänge

G
: Gate

Ausgänge

D
: Drain

S
: Source

Veränderbare Attribute

Daten-Bits
: Anzahl der Daten-Bits, die verwendet werden.

Bezeichnung
: Die Bezeichnung dieses Elementes.

Programmiert
: Wenn gesetzt, ist die Diode "durchgebrannt" bzw "programmiert". Bei Floating Gate FETs wird das Gate geladen. Diese Einstellung kann direkt über die Taste 'P' verändert werden.

Rotation
: Legt die Ausrichtung des Elementes in der Schaltung fest.

Spiegeln
: Spiegelt das Element in der Schaltung.

13.12. Transmissionsgatter

Ein reales Transmissionsgatter ist aus nur zwei Transistoren aufgebaut. Daher wird es oft eingesetzt, um Transistoren einzusparen.

Eingänge

S
: Steuereingang

¬S
: Steuereingang, invertiert

Ausgänge

A
: Eingang A

B
: Eingang B

Veränderbare Attribute

Daten-Bits
: Anzahl der Daten-Bits, die verwendet werden.

Rotation
: Legt die Ausrichtung des Elementes in der Schaltung fest.

14. Sonstige

Test

14.1. Testfall

Beschreibt einen Testfall. In einem Testfall kann beschrieben werden, wie sich eine Schaltung verhalten soll. Es kann dann automatisch überprüft werden, ob das Verhalten der Schaltung tatsächlich dieser Beschreibung entspricht. Ist das nicht der Fall, wird eine entsprechende Fehlermeldung angezeigt. Der Hilfetext des Testfalleditors beschreibt, wie im Detail ein solcher Testfall erstellt werden kann. Exportierbar zu VHDL/Verilog.

Veränderbare Attribute

Bezeichnung
: Die Bezeichnung dieses Elementes.

Testdaten
: Hier wird der eigentliche Testfall angelegt. Details dazu finden sich in der Hilfe des Testdateneditors.

Aktiviert
: Aktiviert oder deaktiviert diese Komponente.

14.2. Generische Initialisierung

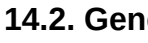
Code der ausgeführt wird, um eine generische Schaltung direkt starten zu können. Soll eine generische Schaltung direkt gestartet werden, muss eine solche Komponente vorhanden sein. Exportierbar zu VHDL/Verilog.

Veränderbare Attribute

Bezeichnung
: Die Bezeichnung dieses Elementes.

Aktiviert
: Aktiviert oder deaktiviert diese Komponente.

generische Parametrisierung
: Anweisung um eine generische Schaltung anzupassen.

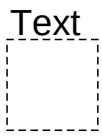


14.3. Code

Code der ausgeführt wird, wenn eine generische Schaltung konkretisiert wird. Kann z.B. benutzt werden, um einer Schaltung Komponenten oder Leitungen hinzuzufügen. Exportierbar zu VHDL/Verilog.

Veränderbare Attribute

generische Parametrisierung
: Anweisung um eine generische Schaltung anzupassen.



14.4. Rechteck

Zeigt ein einfaches Rechteck in der Schaltung an. Hat keine weitere Funktion für die Simulation. Wird als Überschrift ein Minuszeichen eingegeben, entfällt die Überschrift.

Veränderbare Attribute

Bezeichnung
: Die Bezeichnung dieses Elementes.

Breite
: Breite in Rastereinheiten

Höhe
: Höhe in Rastereinheiten

Schriftgröße
: Legt die für diesen Text zu verwendende Schriftgröße fest.

Text innen
: Text ins Innere des Rechtecks setzen.

Text unten
: Text unten ans Rechteck setzen.

Text rechts
: Text rechts ans Rechteck setzen.

An Gitter ausrichten
: Wenn gesetzt, wird das Element am Gitter ausgerichtet.

14.5. Versorgung

Hat keine weitere Funktion. Stellt nur sicher, dass VDD und GND angeschlossen sind. Kann im Zusammenhang mit den 74xx Bausteinen verwendet werden, um Anschlüsse für die Spannungsversorgung zu erzeugen, welche dann auch auf korrekte Beschaltung geprüft werden.

Eingänge

VDD
: Muss mit VDD verbunden werden!

GND
: Muss mit GND verbunden werden!

Veränderbare Attribute

Bezeichnung
: Die Bezeichnung dieses Elementes.

Rotation
: Legt die Ausrichtung des Elementes in der Schaltung fest.

14.6. Bidirektionaler Splitter

Kann für Datenbusse verwendet werden und erleichtert vor allem den Aufbau von Speicherbausteinen im DIL-Gehäuse, da die Implementierung des Datenbusses erleichtert wird.

Eingänge

OE
: Wenn gesetzt, wird der gemeinsame Datenanschluss D an den Bitausgängen D[i] ausgegeben, wenn nicht, werden die Einzelbits D[i] am gemeinamen Ausgang D ausgegeben.

Ausgänge

D
: Der gemeinsame Datenanschluss.

D0
: Das Datenbit 0 des Bus-Splitters.

Veränderbare Attribute

Daten-Bits
: Anzahl der Daten-Bits, die verwendet werden.

Rotation
: Legt die Ausrichtung des Elementes in der Schaltung fest.

Spreizung
: Bestimmt die Spreizung der Ein- und Ausgänge in der Schaltung.

14.7. Reset

Der Ausgang dieses Elements ist Eins, solange sich die Schaltung nach dem Einschalten in der Stabilisierungsphase befindet. Hat sich die Schaltung stabilisiert wird der Ausgang auf Null gesetzt. Bei einem invertiertem Ausgang verhält es sich genau anders herum. Exportierbar zu VHDL/Verilog.

Ausgänge

Reset
: Reset Ausgang.

Veränderbare Attribute

Bezeichnung
: Die Bezeichnung dieses Elementes.

invertierter Ausgang
: Wenn gesetzt, wird der Ausgang invertiert.

Rotation
: Legt die Ausrichtung des Elementes in der Schaltung fest.

14.8. Break

Wenn dieses Element in der Schaltung verwendet wird, wird der "Run To Break" Knopf zwischen "Start" und "Stop" aktiviert. Dieser Knopf taktet die Schaltung bis eine steigende Flanke am Eingang des "Break" Elements erkannt wird. Dieses Element kann zur

Fehlersuche eingesetzt werden, indem die Schaltung bis zu einem beliebigen Haltepunkt vorgetaktet wird. Auch kann damit ein Assemblerbefehl BRK implementiert werden. Dies erlaubt dann, ein Programm bis zum nächsten BRK-Befehl auszuführen. Diese Funktion kann nur genutzt werden, wenn der Echtzeittakt deaktiviert ist!

Eingänge

brk
: Stoppt den schnellen Simulationsvorlauf, bei einer steigenden Flanke.

Veränderbare Attribute

Bezeichnung
: Die Bezeichnung dieses Elementes.

Aktiviert
: Aktiviert oder deaktiviert diese Komponente.

Timeout Zyklen
: Wenn nach dieser Anzahl von Takten kein Break eingegangen ist, wird ein Fehler erzeugt

Rotation
: Legt die Ausrichtung des Elementes in der Schaltung fest.

14.9. Stop

Eine steigende Flanke am Eingang beendet die Simulation. Hat den gleichen Effekt wie das Drücken des Stop-Knopfes in der Werkzeugleiste.

Eingänge

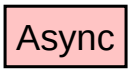
stop
: Eine steigende Flanke beendet die Simulation.

Veränderbare Attribute

Bezeichnung
: Die Bezeichnung dieses Elementes.

inverse Eingänge
: Es können die Eingänge ausgewählt werden, welche invertiert werden sollen.

Rotation
: Legt die Ausrichtung des Elementes in der Schaltung fest.



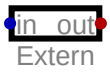
14.10. Asynchrones Timing

Erlaubt die Konfiguration des Timings eines asynchronen Automaten wie z.B einer Muller-Pipeline. Die Schaltung muss im Gatterschrittmodus gestartet werden und muss zunächst einen stabilen Zustand einnehmen. Interaktiv oder mit einem Reset-Element kann dann der Automat gestartet werden. Die Taktelemente können in diesem Modus nicht verwendet werden.

Veränderbare Attribute

Echtzeittakt starten
: Wenn eingeschaltet, wird beim Start der Schaltung der Echtzeittakt gestartet.

Frequenz/Hz
: Gibt die Frequenz an, wenn der Echtzeittakt aktiviert ist



14.11. Extern

Element zur Anbindung von externen Programmen zur Berechnung der Logik. Wird verwendet, um das Verhalten eines Elements mit einer Hardwarebeschreibungssprache wie VHDL oder Verilog zu beschreiben. Die eigentliche Simulation des Verhaltens muss mit einem externen Simulator erfolgen. Zur Zeit wird nur der VHDL-Simulator ghdl und der Verilog-Simulator Icarus Verilog unterstützt. Exportierbar zu VHDL/Verilog.

Eingänge

in

Ausgänge

out

Veränderbare Attribute

Bezeichnung
: Die Bezeichnung dieses Elementes.

Breite
: Breite des Symbols, wenn diese Schaltung in eine andere eingefügt wird.

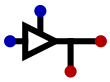
Eingänge
: Die Eingänge des externen Prozesses. Es handelt sich um eine kommaseparierte Liste mit Signalnamen. Bei jedem Signalnamen kann, mit einem Doppelpunkt getrennt, eine Bitanzahl angegeben werden. Die Eingänge eines 8-Bit Addierers könnten also mit "a:8,b:8,c_in" beschrieben werden.

Ausgänge
: Die Ausgänge des externen Prozesses. Es handelt sich um eine kommaseparierte Liste mit Signalnamen. Bei jedem Signalnamen kann, mit einem Doppelpunkt getrennt, eine Bitanzahl angegeben werden. Die Ausgänge eines 8-Bit Addierers könnten also mit "s:8,c_out" beschrieben werden.

Programmcode
: Der Programmcode welcher ausgeführt werden soll.

Anwendung
: Gibt an, welche Anwendung gestartet werden soll, um den Code auszuführen.

GHDL Optionen
: Optionen, die für alle Bearbeitungsschritte durch GHDL verwendet werden.



14.12. Pinsteuerung

Steuerlogik für einen bidirektionalen Pin. Diese ist nur im Rahmen der VHDL- oder Verilog-Generierung erforderlich, um einen bidirektionalen HDL-Port zu erstellen! Wenn Sie keinen bidirektionalen IO-Port auf einem FPGA verwenden wollen, verwenden Sie diese Komponente nicht! Die Pinsteuerung kann nicht in einer eingebetteten Schaltung verwendet werden! Sie ist nur auf der obersten Schaltungsebene erlaubt! Exportierbar zu VHDL/Verilog.

Eingänge

wr
: Die auszugebenden Daten

oe
: Aktiviert die Ausgabe

Ausgänge

rd
: Die zu lesenden Daten.

pin
: Der Anschluss für den eigentlichen Pin. Hier sollte nur noch ein einzelner Ausgang angeschlossen werden.

Veränderbare Attribute

Daten-Bits
: Anzahl der Daten-Bits, die verwendet werden.

Rotation
: Legt die Ausrichtung des Elementes in der Schaltung fest.

Spiegeln
: Spiegelt das Element in der Schaltung.

E Bibliothek

27c801: 8 Mbit (1Mb x 8) UV EPROM
28c010: 1-Megabit (128K x 8) Paged Parallel EEPROM; DATA Polling for End of Write Detection not implemented!
28c16: 16K (2K x 8) Parallel EEPROM; DATA Polling for End of Write Detection not implemented!
28c64: 64K (8K x 8) Parallel EEPROM; DATA Polling for End of Write Detection not implemented!
28c256: 256K (32K x 8) Paged Parallel EEPROM; DATA Polling for End of Write Detection not implemented!
28c512: 512K-Bit (64K x 8) CMOS Parallel EEPROM; DATA Polling for End of Write Detection not implemented!
7400: quad 2-input NAND gate
7401: quad 2-input NAND gate with open-collector outputs
7402: quad 2-input NOR gate
7403: quad 2-input NAND gate with open-collector outputs, different pinout than 7401
7404: hex inverter
7405: hex inverter, open-collector output
7406: hex inverter buffer, open-collector output
7407: hex buffer, open-collector output
7408: quad 2-input AND gate
7409: quad 2-input AND gate with open-collector outputs
7410: triple 3-input NAND gate
7411: triple 3-input AND gate
7412: triple 3-input NAND gate with open-collector outputs
7413: dual 4-input NAND gate, Schmitt trigger
7414: hex inverter, Schmitt trigger
7415: triple 3-input AND gate with open-collector outputs
7416: hex inverter buffer, open-collector output, same as 7406
7417: hex buffer, open-collector output, same as 7407
7420: dual 4-input NAND gate
7421: dual 4-input AND gate
7425: dual 4-input NOR gate
7427: triple 3-input NOR gate
7428: quad 2-input NOR buffer
7430: 8-input NAND gate
7432: quad 2-input OR gate
7440: dual 4-input NAND buffer
7442: 4-line BCD to 10-line decimal decoder
7447: BCD to 7-segment decoder, active low
7448: BCD to 7-segment decoder, active high
7451: 2-input/3-input AND-NOR gate
7454: 2-3-2-3-line AND NOR gate
7455: 2 wide 4-input AND-NOR gate
7458: dual AND OR gate
7474: dual D-flip-flop
7476: dual J-K flip-flops with preset and clear
7480: Gated Full Adder with Complementary Inputs and Complementary Sum Outputs
7482: 2-bit binary full adder

7483: 4-bit binary full adder
7483Real: 4-bit binary full adder, real gates
7485: 4-bit comparator
7486: quad 2-input XOR gate
7489: 64-bit RAM
74107: dual J-K flip-flops with clear
74109: Dual J-NOT-K flip-flop with set and reset; positive-edge-trigger
74112: Dual J-K negative-edge-triggered flip-flop, clear and preset
74116: dual 4-bit D-type latches
74133: 13-input NAND gate
74138: 3-line to 8-line decoder/demultiplexer, inverted out
74139: dual 2-line to 4-line decoder/demultiplexer
74147: 10-line to 4-line priority encoder
74148: 8-line to 3-Line priority encoder
74150: 4-line to 16-line data selectors/multiplexers
74151: 3-line to 8-line data selectors/multiplexers
74153: dual 4-line to 1-line data selectors/multiplexers
74154: 4-line to 16-line decoders/demultiplexers
74157: quad 2-line to 1-line data selectors/multiplexers
74160: decimal synchronous counter, async clear
74161: hex synchronous counter, async clear
74162: decimal synchronous counter
74162Real: decimal synchronous counter, real gates
74163: hex synchronous counter
74164: 8-bit parallel-out serial shift register, asynchronous clear
74165: parallel-load 8-bit shift register
74166: 8-Bit Parallel-In/Serial-Out Shift Register
74173: quad 3-state D flip-flop with common clock and reset
74174: hex D-flip-flop
74181: 4-bit arithmetic logic unit
74182: look-ahead carry generator
74189: 64-Bit Random Access Memory with 3-STATE Outputs
74190: Presettable synchronous 4-bit bcd up/down counter
74191: Presettable synchronous 4-bit binary up/down counter
74193: Synchronous 4-Bit Up/Down Binary Counter with Dual Clock
74198: 8-bit shift register
74238: 3-line to 8-line decoder/demultiplexer
74244: octal 3-state buffer/line driver/line receiver
74245: octal bus transceivers with 3-state outputs
74247: BCD to 7-segment decoder, active low, tails on 6 and 9
74248: BCD to 7-segment decoder, active high, tails on 6 and 9
74253: dual tri state 4-line to 1-line data selectors/multiplexers
74260: dual 5-input NOR gate
74266: quad 2-input XNOR gate
74273: octal D-type flip-flop with clear
74280: 9 bit Odd-Even Parity Generator-Checker
74283: 4-bit binary full adder, alternative pinning
74373: octal transparent latches
74374: octal positive-edge-triggered flip-flops
74377: Octal D Flip-Flop with enable

74382: 4-Bit Arithmetic Logic Unit
74540: octal buffer/line driver, inverted
74541: octal buffer/line driver
74573: octal transparent latches, different pinout compared to 74373
74574: octal positive-edge-triggered flip-flops, different pinout compared to 74374
74590: 8-bit binary counter with tri-state output registers
74595: 8-Bit Shift Registers with 3-State Output Registers
74670: 3-state 4-by-4 Register File
74682: 8-bit digital comparator
74688: 8-bit identity comparator
74779: 8-Bit Bidirectional Binary Counter with 3-STATE Outputs
74804: hex 2-input NAND gate https://www.ti.com/lit/ds/symlink/sn74as804b.pdf
74805: hex 2-input NOR gate http://www.ti.com/lit/ds/symlink/sn54as805b.pdf
74808: hex 2-input AND gate http://www.ti.com/lit/ds/symlink/sn54as808b.pdf
74832: hex 2-input OR gate http://www.ti.com/lit/ds/symlink/sn54as832b.pdf
744017: Johnson decade counter with 10 decoded outputs
744075: triple 3-input OR gate
A623308A: 8K X 8 BIT CMOS SRAM
RAM32Bit: A 32-bit memory that allows byte access and can handle non-aligned memory addresses.