Digital

Revision: v0.20
Date: 2018-09-03 14:36

Table of Contents

A General

1. Digital

1.1. Introduction

Digital is a simple simulator used to simulate digital circuits. The logic gates are connected to each other by wires and the behavior of the overall circuit can be simulated. The user can interact with the simulation by either pressing buttons or setting values to the inputs of the circuit.

In this way, most of the basic circuits used in digital electronics can be built and simulated. In the folder *examples*, users can browse for examples that includes a functional 16-bit single-cycle Harvard processor.

The simulator has two modes of operation: Editing and Simulation mode. In the editing mode, modifications to the circuit can be performed. Users can add or connect components. In this mode, simulation is disabled. The simulation mode is activated by pressing the *Start* button in the tool bar. While starting the simulation the circuit is checked for consistency. If there are errors in the circuit an appropriate message is shown and the affected components or wires are highlighted. If the circuit is error free, the simulation is enabled. Now you can interact with the running simulation. In the simulation mode it is not possible to modify the circuit. To do so you have to activate the editing mode again by stopping the simulation.

1.2. First Steps
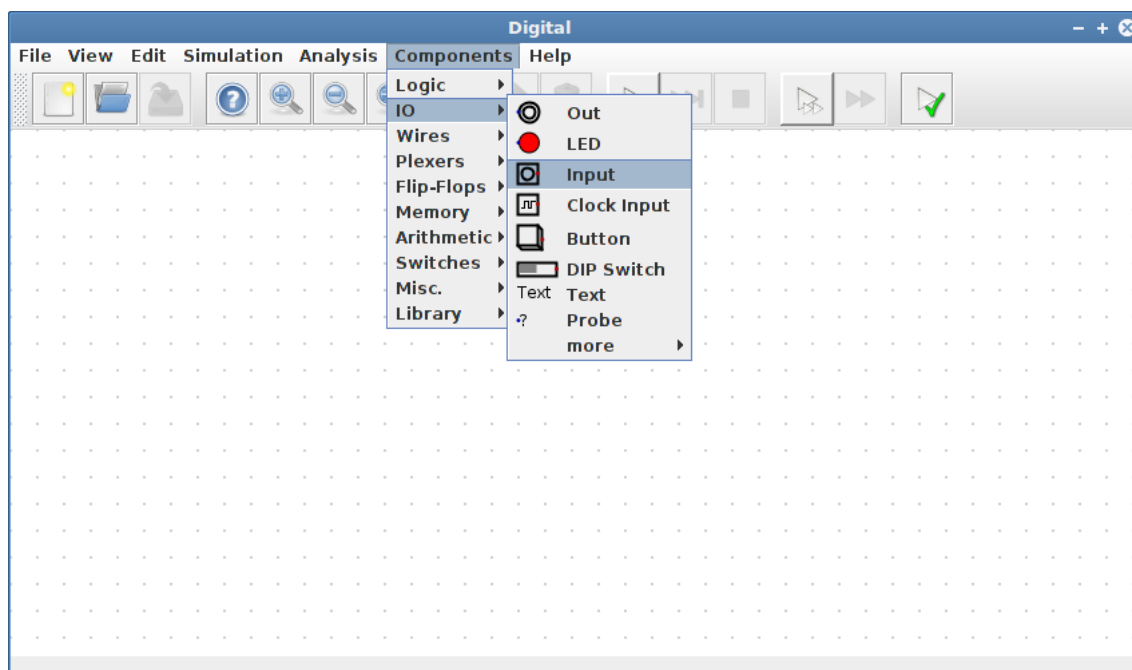
As a first example, a circuit is to be constructed with an Exclusive-Or gate. From the main window, the *Components* menu allows you to select the various components. Then they are placed on the drawing panel. This process can be canceled by pressing the ESC key at any time. Start by selecting an input component. This can later be controlled interactively by using the mouse.
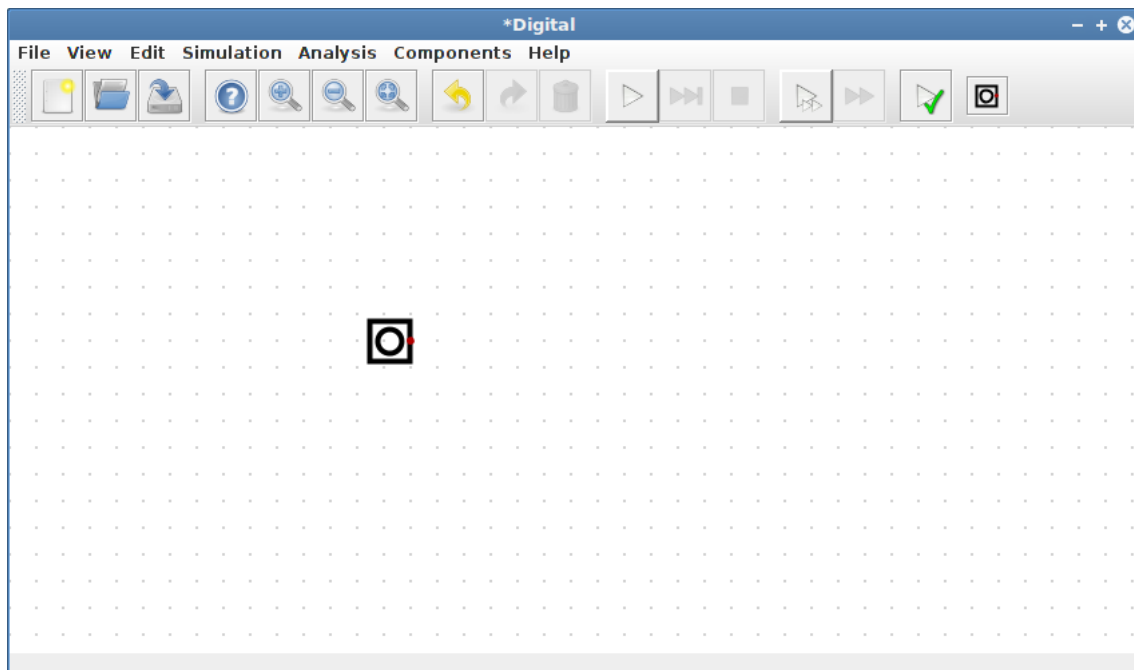
After selection, the first input can be placed on the drawing panel. The red dot on the input component symbol is a connection point between the component and a wire, which will be connected later on. The red color indicates an output. This means that the port defines a signal value or can drive a wire.
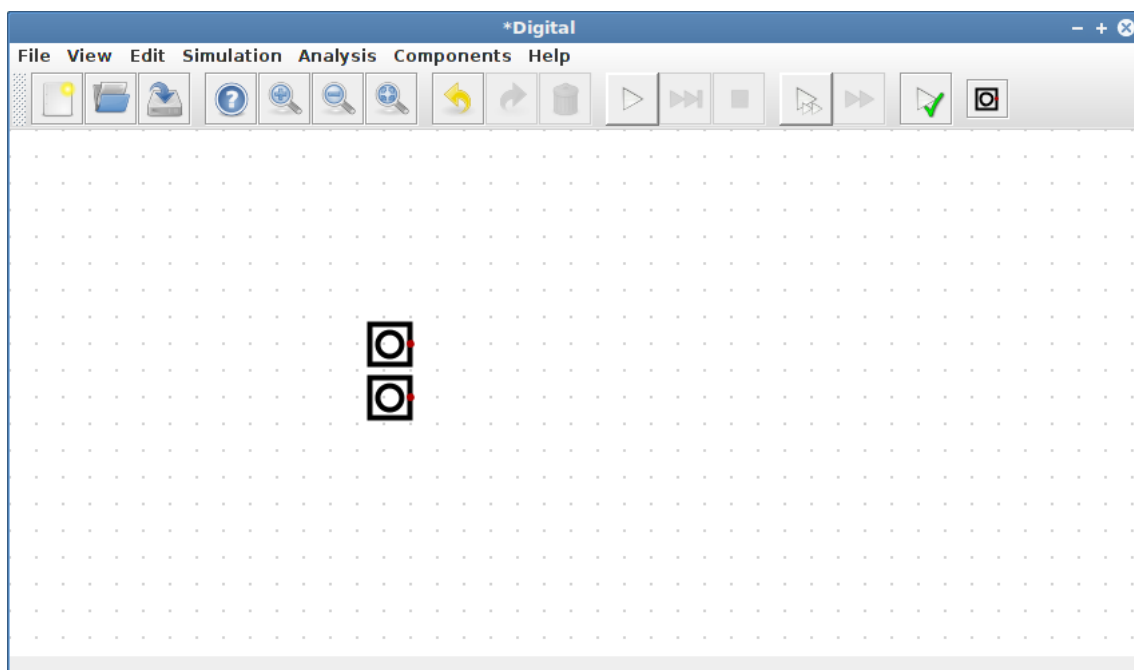
In the same way, a second input is added. It is best to place it directly below the first input.
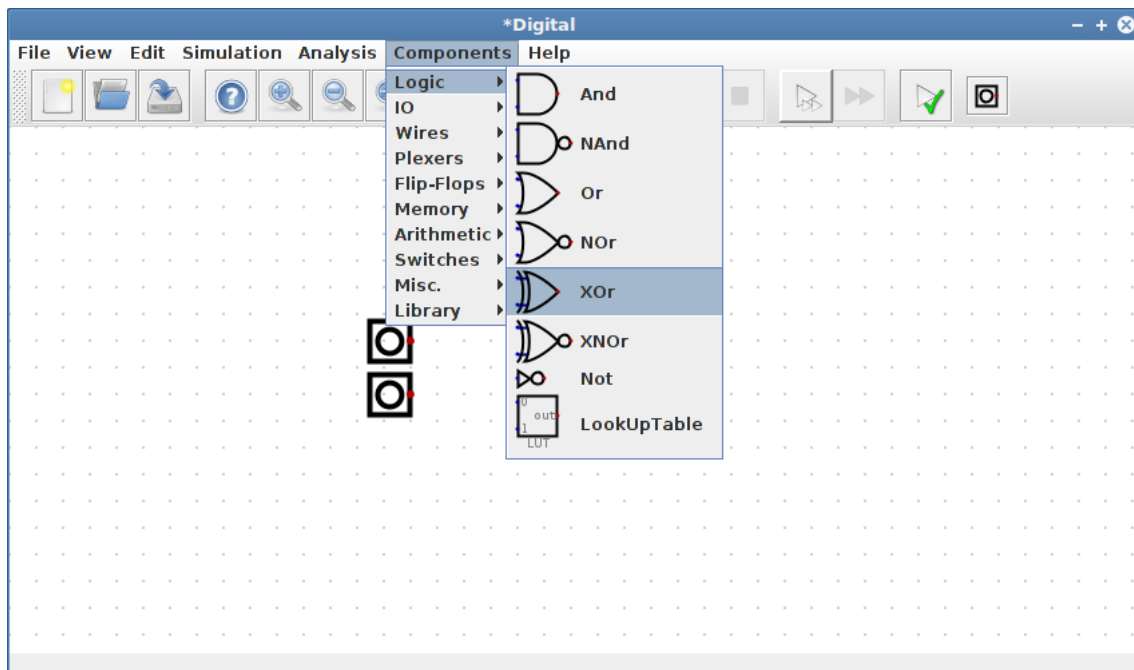
After adding the inputs, the Exclusive-Or gate is selected. This gate represents the actual logical function.
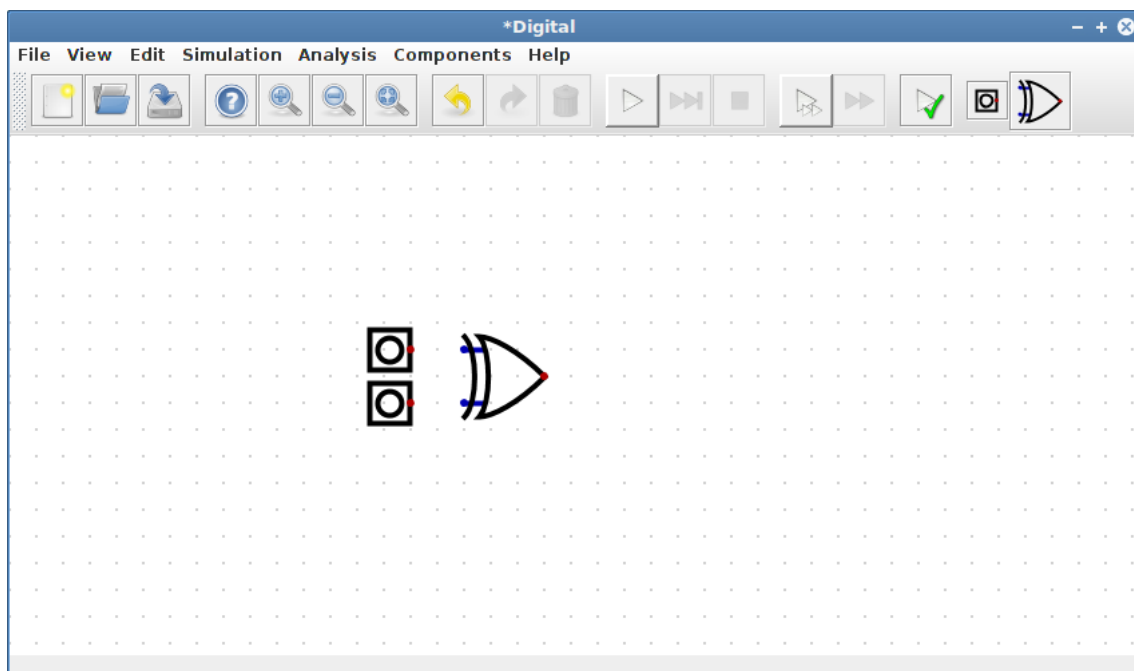
This gate can now also be added to the circuit. It is best to place it in a way that the subsequent wiring is made as simple as possible. The blue dots indicate the input terminals of the gate.
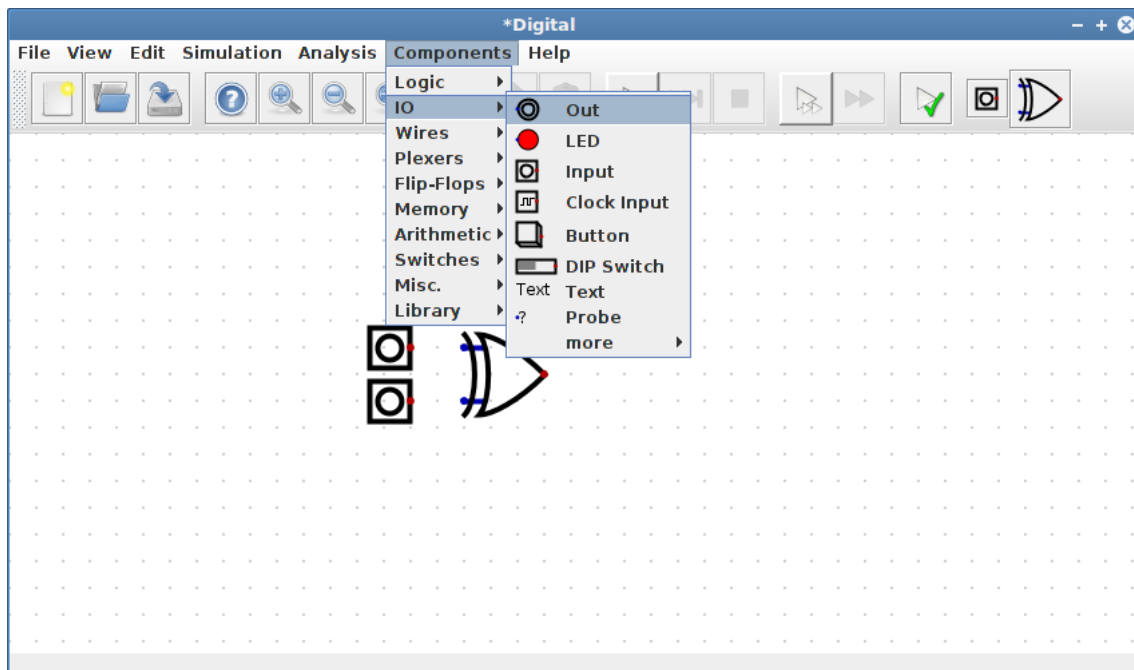
Now, select an output which could be used to display a signal state or to later pass signals to an embedding circuit.
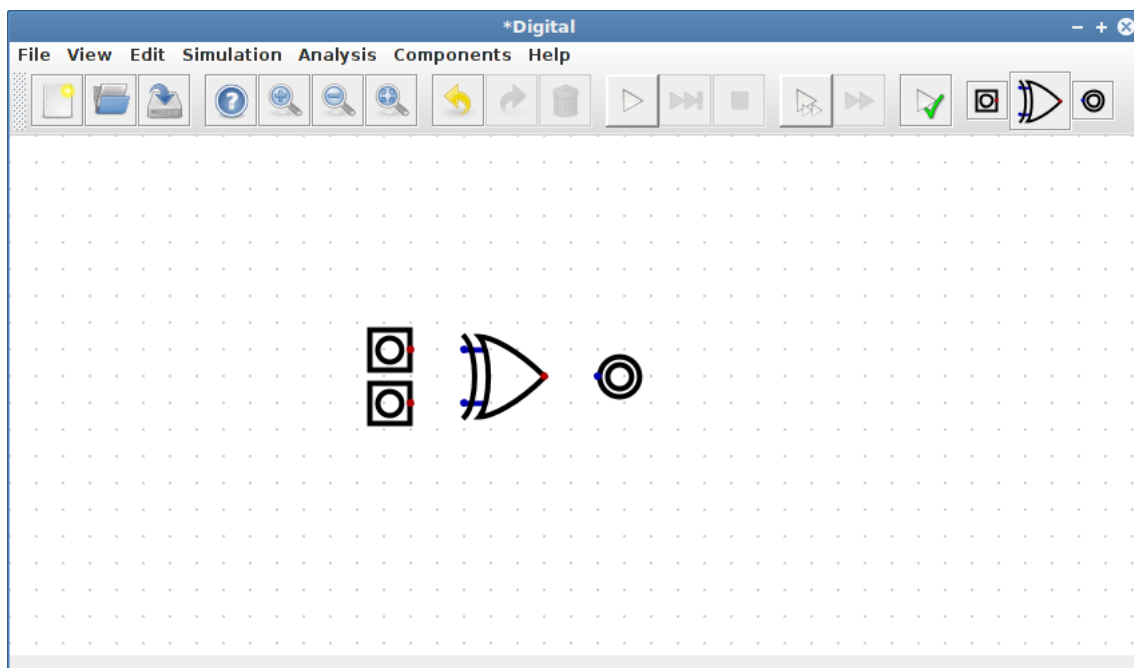
This is placed in a way that it can be wired easily. The output has a blue dot, which indicates an input terminal. Here you can feed in the value which is then exported.
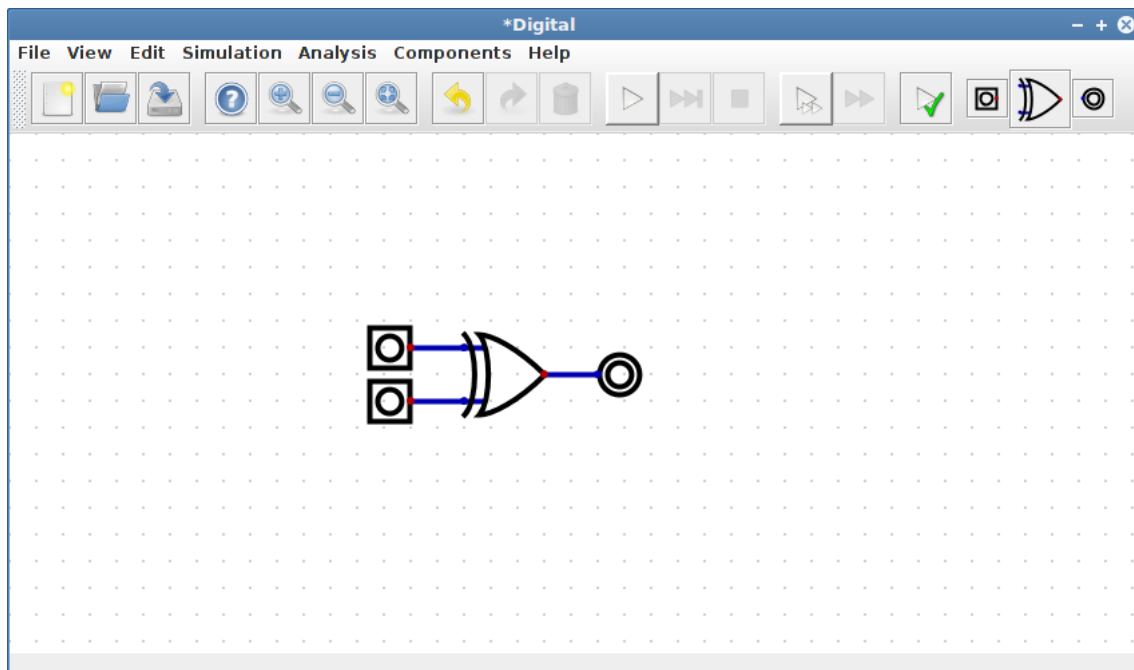
After all components are selected and in place, use the mouse to wire a connection between the blue and red dots. Make sure that exactly one red dot is connected to any number of blue dots. Only the usage of three-state outputs makes it possible to deviate from this rule and to interconnect several red dots. If all wires have been drawn, the circuit is complete.
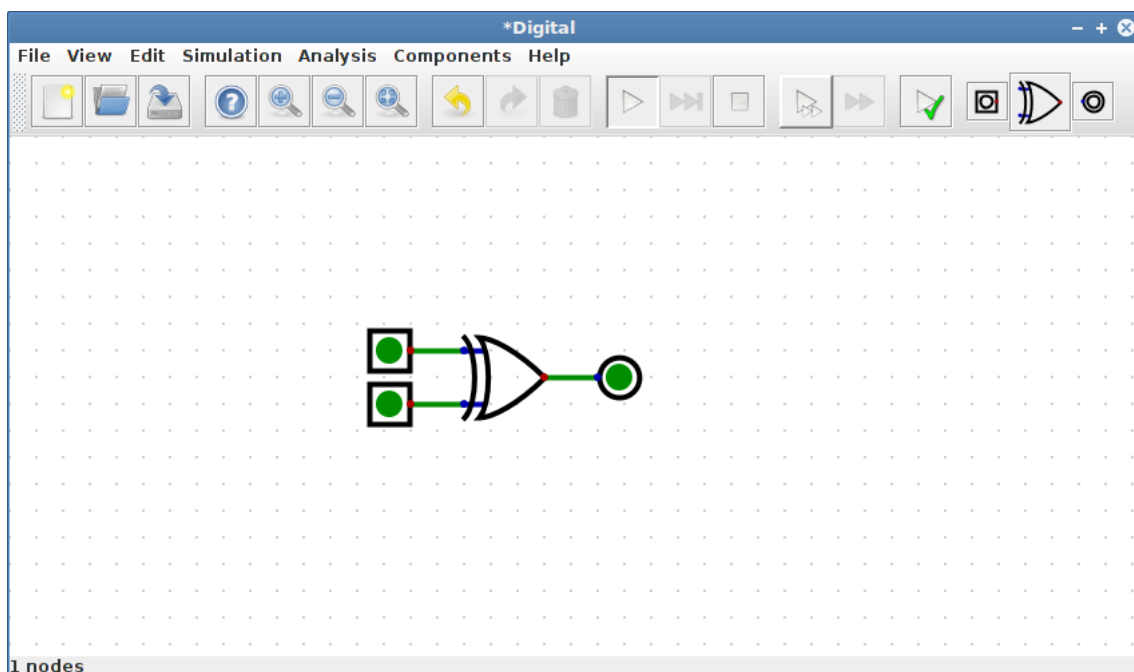
Interaction with the circuit is possible when simulation is started. This is done by clicking on the play button located in the toolbar. After starting the simulation, the color of the wires changes and the inputs and outputs are now filled. Bright green indicates a logical '1' and dark green a logical '0'. In the figure above, all wires have a '0' value.
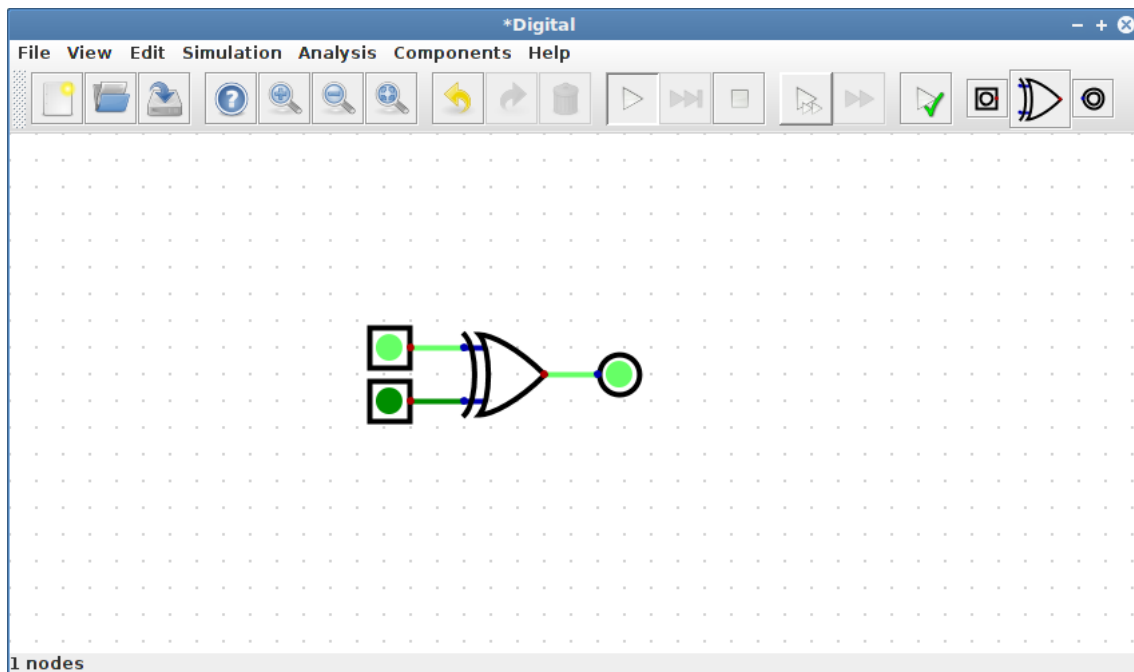
By clicking with the mouse, the inputs can be switched. Since the simulation is now active, the output changes according to the current input states. The circuit behaves like an Exclusive-Or gate as expected .
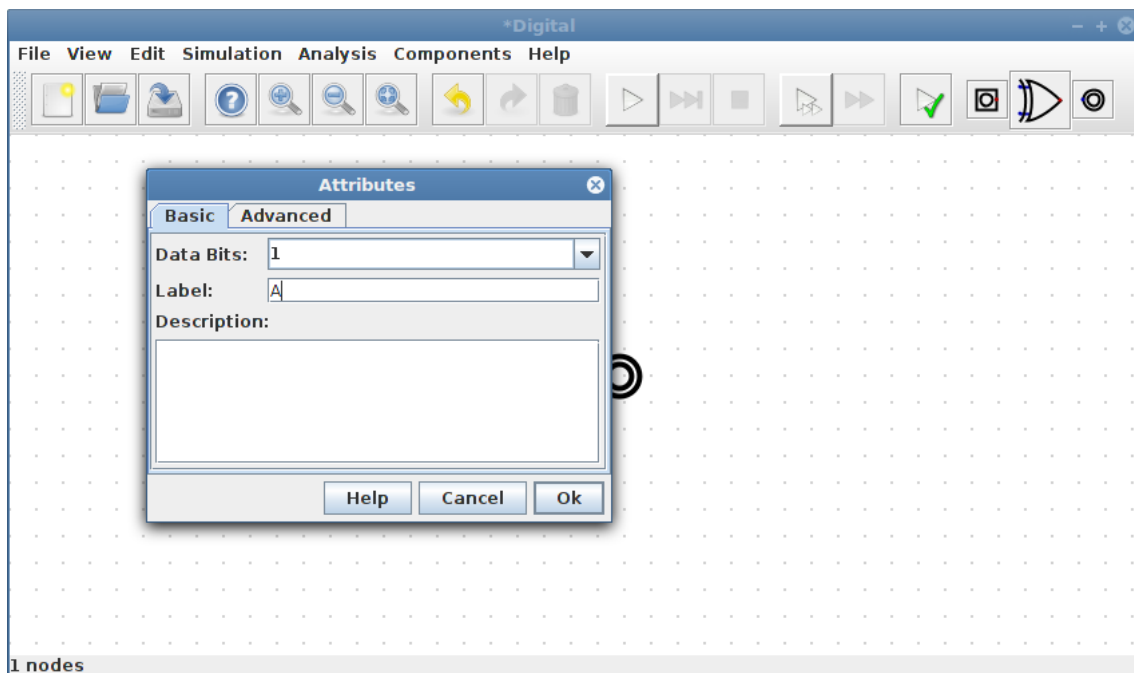
To further process the circuit, the simulation must first be stopped. The easiest way to do this is with the Stop button in the tool bar. Clicking on a component with the right mouse button opens a dialog which displays the component's properties. The label 'A' can be defined for the first input via this dialog.
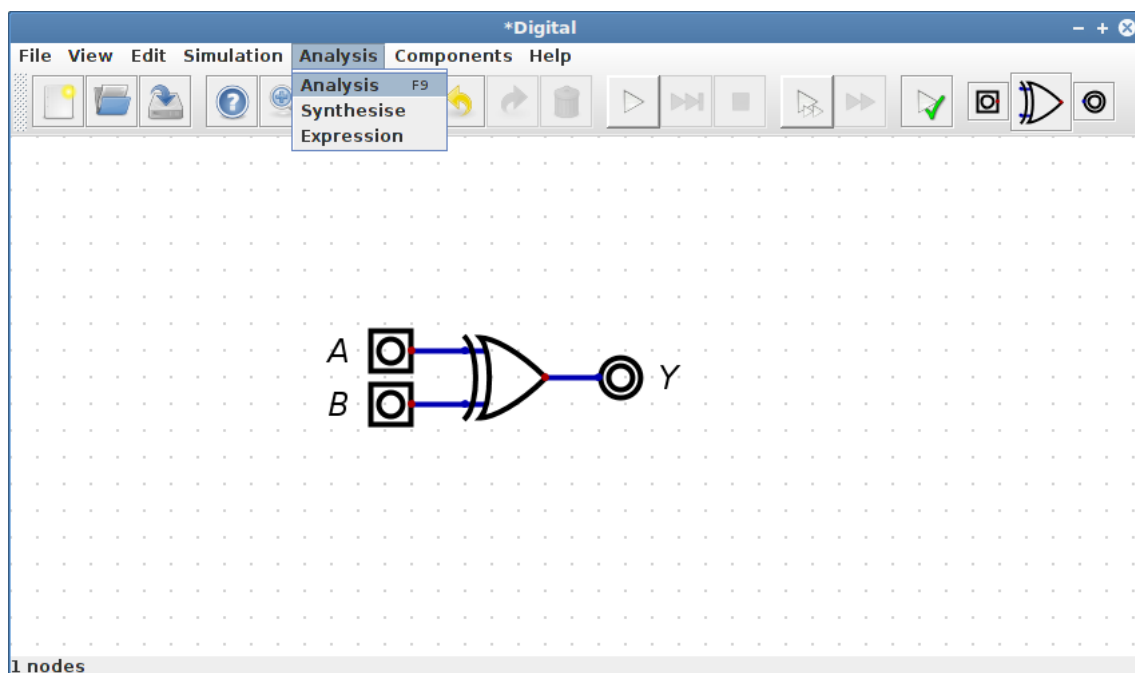
In this way, the labels for the remaining inputs and outputs can be configured. The menu item *Analysis* also contains a menu item *Analysis.* This function performs an analysis of the current circuit. However, this is only possible if all inputs and outputs are labeled properly.
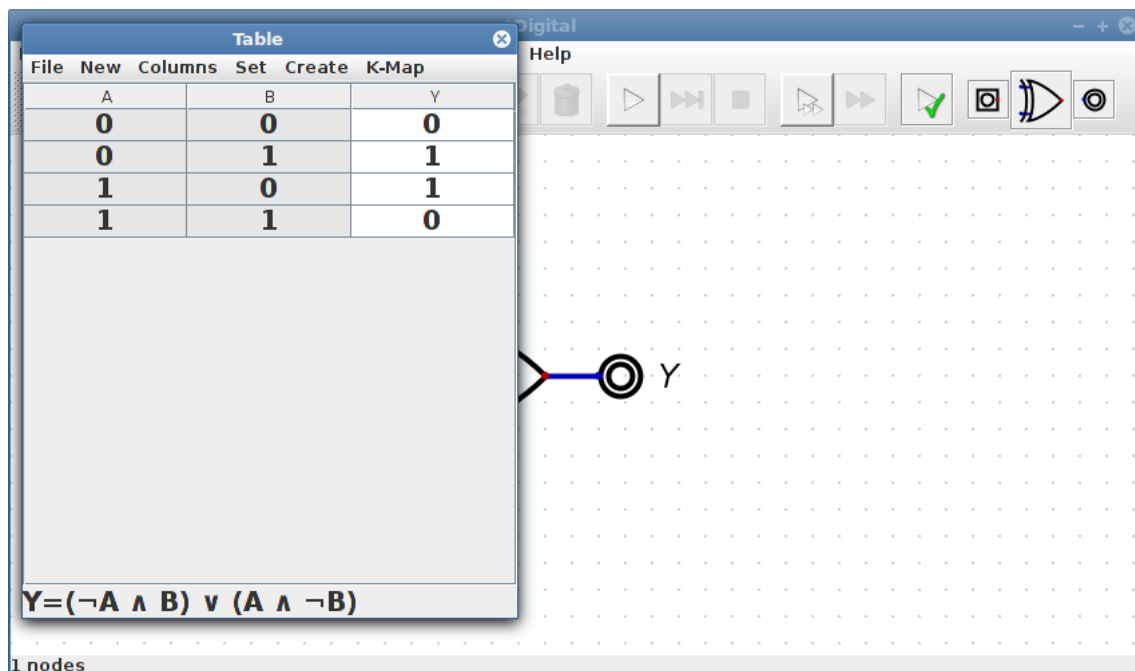
The truth table of the simulated circuit appears in a new window. Below the table you can find the algebraic expression associated with the circuit. If there are several possible algebraic expressions, a separate window will open, showing all possible expressions.
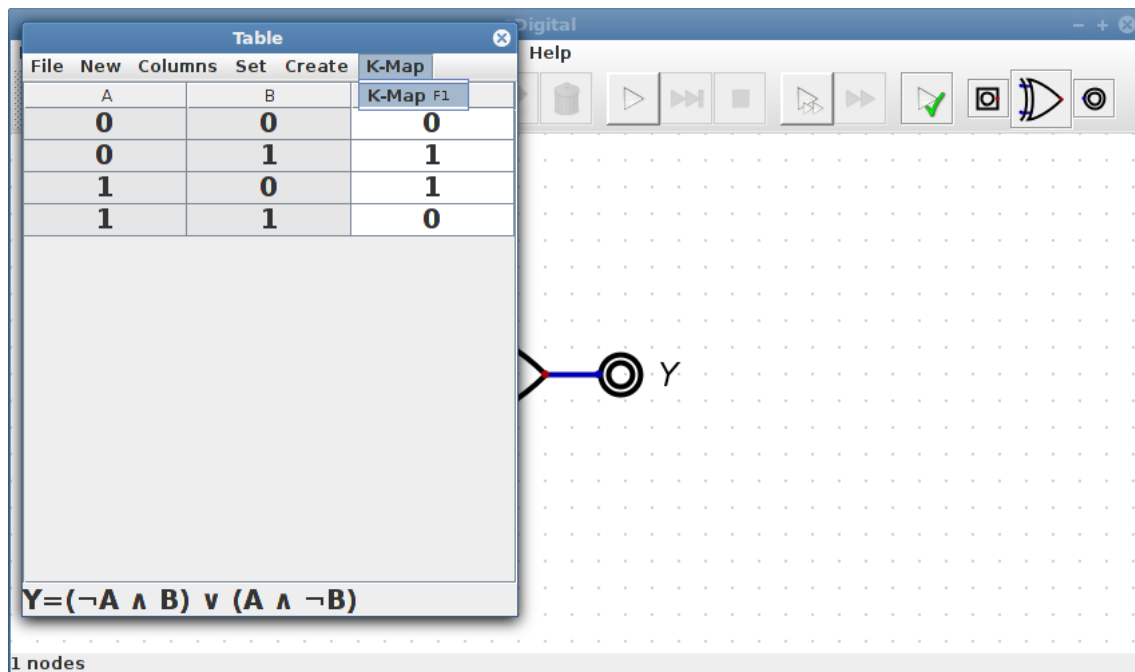
The table dialog has the menu entry *K-Map* in its main menu. This allows to display the truth table in the form of a K-map.
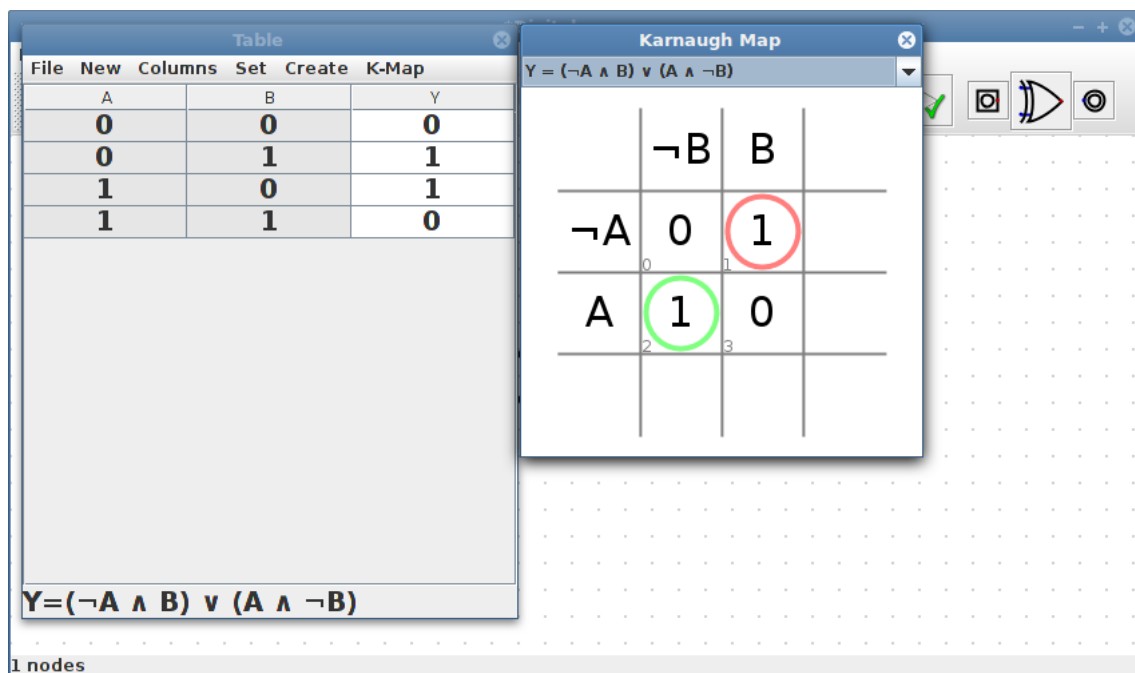
At the top of this dialog there is a drop-down list which allows the selection of the desired expression in the K-map. In this way you can, for example, illustrate how several equivalent algebraic expressions can result. However, in this example, there is only one minimal expression. The truth table can also be modified by clicking the K-map.

1.3. Wires

All components must be connected via wires. It is not possible to connect two components by placing them directly next to each other.

In addition, there are only connections between an endpoint of a wire and a component. If a pin of a component is placed in the middle of a wire, no connection is made between the component and the wire. Therefore, a wire must actually terminate at each pin which is to be connected. Even if the tunnel component is used, there must be a wire between the pin and the tunnel element.

The component needs to be selected using the rectangular selection tool in order to be moved, including the connected wires. For moving a component without the connected wires, select the component using a mouse click.

1.4. Hierarchical Design

If a complex circuit is built up, this can quickly become very confusing. To keep track here, the different parts of a circuit can be stored in different files. This mechanism also makes it possible to use a subcircuit, which has been created once, several times in a further circuit. This approach also offers the advantage that the files can be stored independently of each other in a version control system and changes can be tracked.
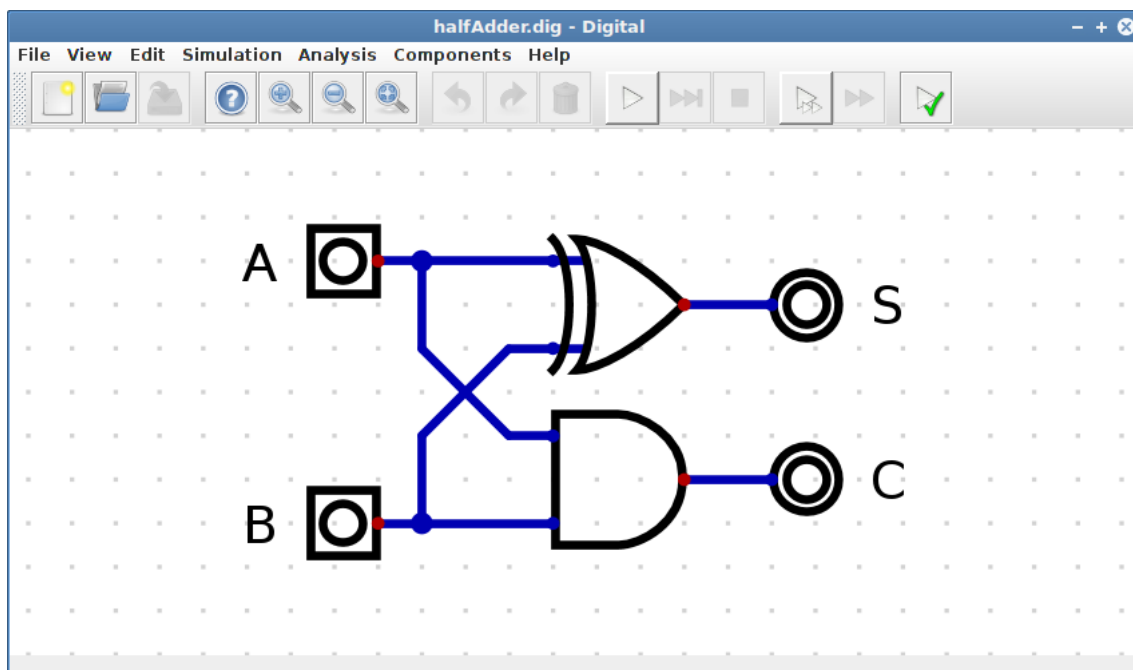
As an example, consider a 4-bit adder: First, we built a simple half-adder. This consists of an XOR gate and an AND gate. The sum of the two bits 'A' and 'B' is given to the outputs 'S' and 'C'. This circuit is stored in the file *halfAdder.dig*.
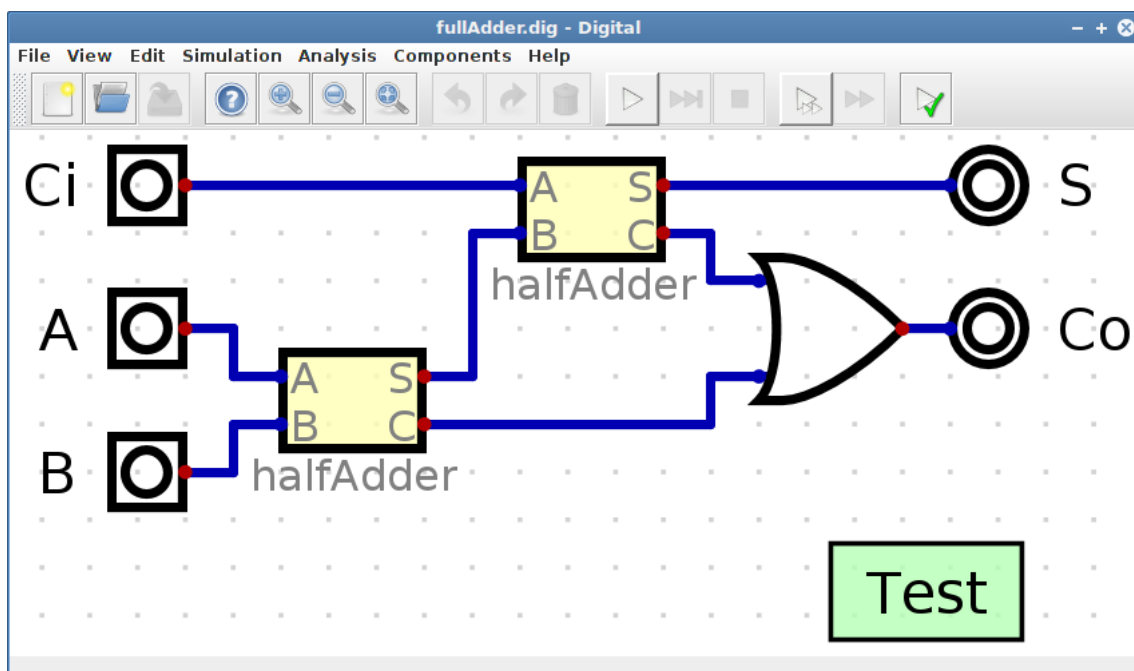
From two half adders a full adder can now be built. To do this, create a new empty file and save the empty file as *fullAdder.dig* in the same folder as the half adder. Then the half adder can be added to the new circuit via the *Components→Custom* menu. The order of the pins at the package of the half-adder can be rearranged in the half adder in the menu *Edit→Order inputs* or *Edit→Order outputs*. The full adder adds the three bits 'A', 'B' and 'Ci' and gives the sum to the outputs 'S' and 'Co'.
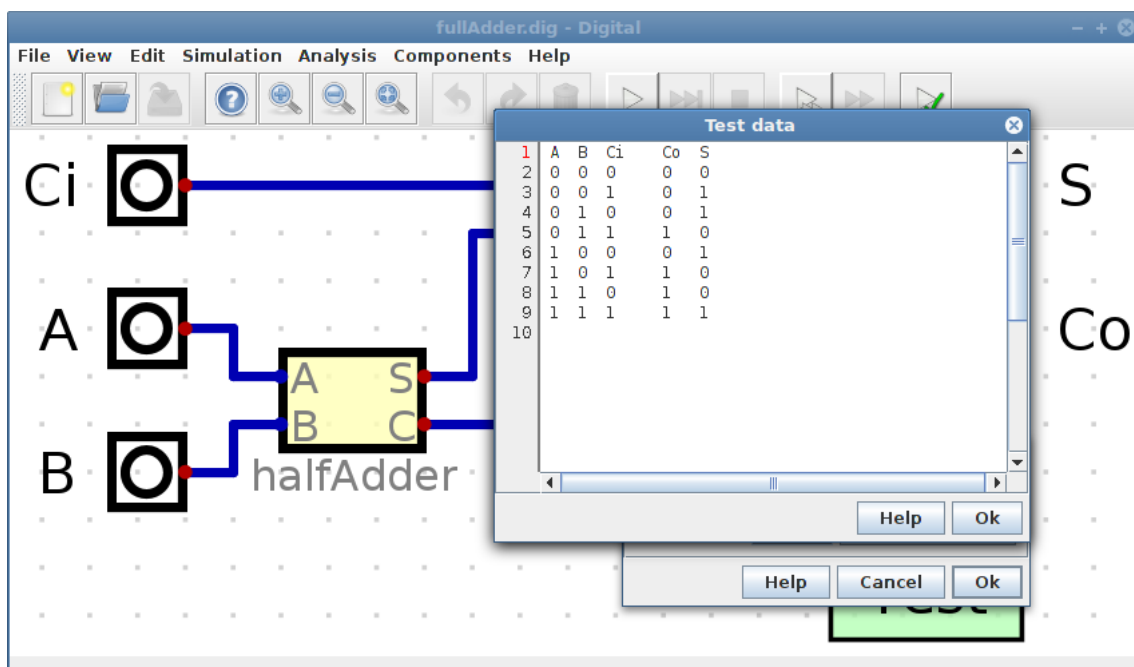
In order to check the correct function of the full adder, a test case should be added. In the test case, the truth table is stored, which should fulfill the circuit. In this way it can be automatically checked whether this is the case.
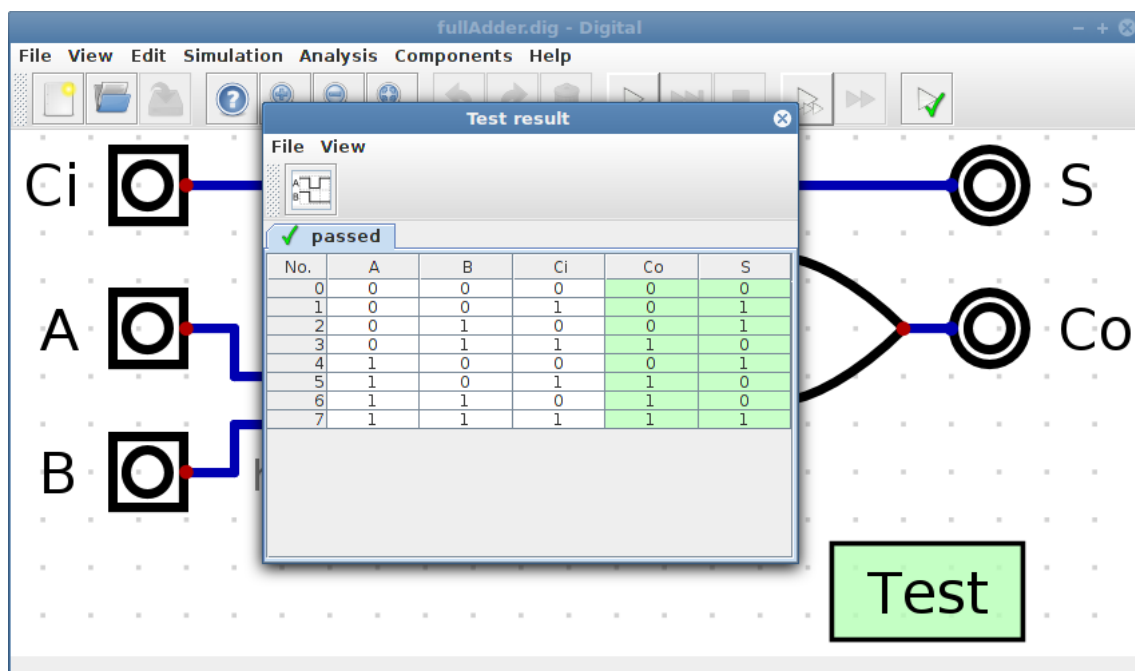
The tests can be executed via the test case editor or the test button in the toolbar. The table cells highlighted in green indicate that the output of the circuit matches the truth table given in the test case.
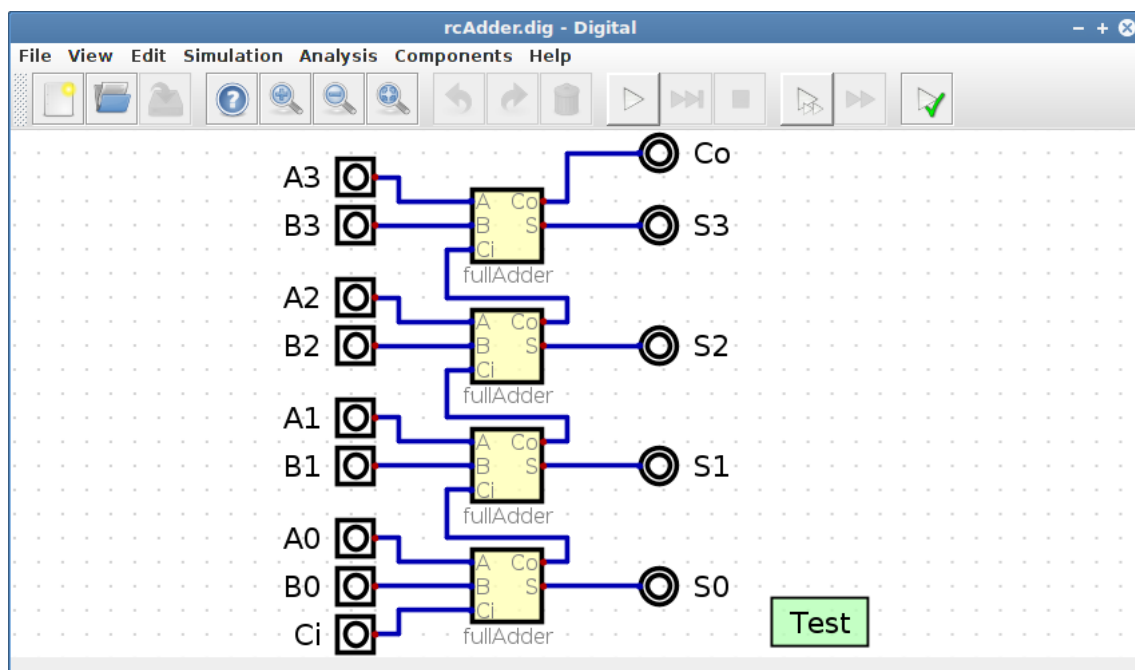
Now the full adders can be put together to form a so-called ripple-carry adder. In this case, the carry output of an addition is forwarded as a carry input to the addition of the next higher-order bit, just as is usual in pencil-and-paper addition. This 4-bit adder should be tested for correct function. For this purpose a test case was inserted.
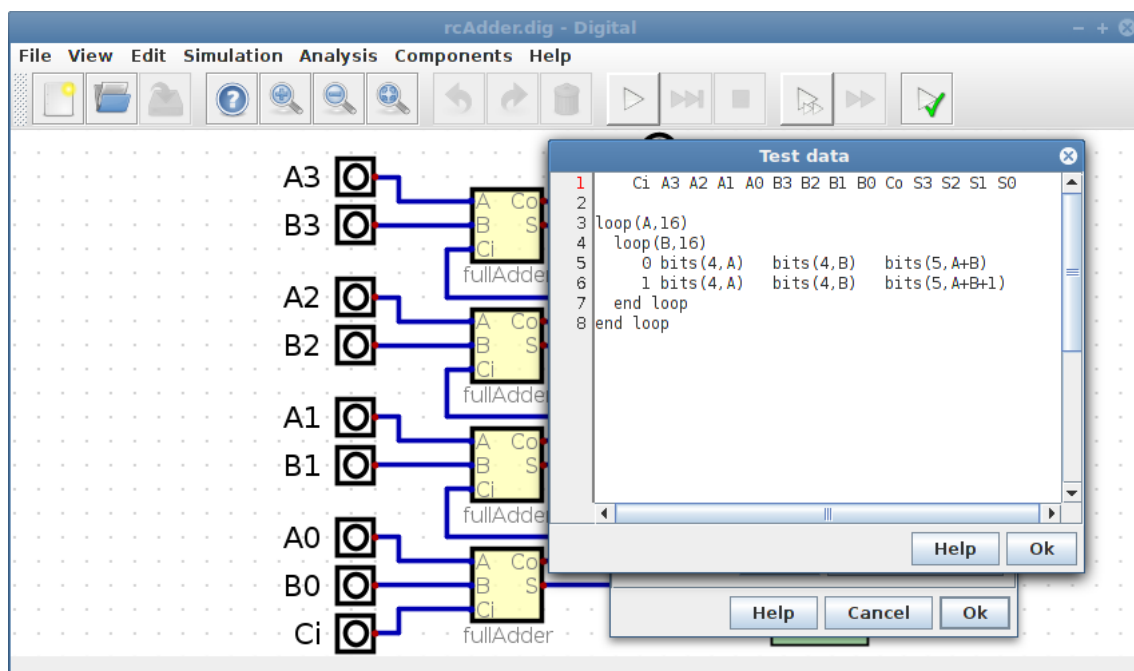
This test case performs a 100% test, which is possible only with relatively simple circuits: all possible 512 input combinations are applied to the circuit, and it is checked whether the output of the circuit is correct. The first line lists the input and output signals. Below this, the input values to be applied and the output values to be checked are specified in a row, as in a truth table. In this example, however, 512 lines are required. Entering this would be a tedious and error-prone task. It is easier and more reliable to automatically generate the required lines. For this purpose, the variables *A* and *B* are each traversed from 0 to 15. The respective values of *A* and *B* are then assigned to inputs 'A[n]' and 'B[n]'. Then it is checked whether the circuit outputs the value *A+B*. Then it is checked again with the carry bit set, in which case *A+B+1* must result. The details of the test syntax are provided by the help dialog.

If a circuit is embedded in an other circuit, only the file name of the subcircuit is stored in a circuit, not the embedded circuit itself. The corresponding files of the embedded subcircuits must therefore be found in the file system at runtime of the simulation. In order to support the various work methods of the users as best as possible and still to avoid a complex administration of import paths, etc., a somewhat unusual import strategy is implemented.

Only the file names of the embedded circuits are stored in a circuits file, not the full path. If a file needs to be opened, all subfolders are searched for a file of the corresponding name. If a suitable file is found, it is imported. This process only depends on the file name of the file to be read, not on its path. Correspondingly, an error message is generated if there are several files of the same name in different subfolders, since ambiguities then arise.

A suitable project structure therefore looks as follows: The root circuit is located in a separate folder. All imported circuits must be in the same folder or subfolders. All circuits must have different names, so it must not happen that there are circuits of the same name in different folders.

2. Simulation

2.1. Propagation Delay

During the simulation every logic gate has a propagation delay. Every component found in the library has the same propagation delay regardless of its complexity. The AND gate thus has the same propagation delay as the multiplier. The only exceptions are diodes, switches and splitters which are used to create data buses. These components have no propagation delay at all.

If it's necessary to simulate a gate - e.g. the multiplier - with a longer propagation delay, a delay gate must be inserted in the circuit right behind the output of the multiplier.

If a circuit is included in another parent circuit, the included circuit keeps its timing behaviour. So if you include a complex circuit which has a large propagation delay because the input signals has to pass three gates until it reaches the output, this behaviour is conserved while including this circuit. There are no additional delays introduced as a result of including a circuit. If not all outputs of a circuit have the same propagation delay, then this is also the case if it is included in a parent circuit. In general, including a circuit into an other circuit does not modify its timing behaviour at all. An included circuit behaves exactly the same way as if all components had been inserted at the same circuit level.

3. Circuit Analysis and Synthesis

A circuit can be analyzed via the menu entry *Analysis*. A truth table is generated for purely combinatorial circuits. This truth table can be edited as desired. A new circuit can be generated from this truth table after editing.

In addition to purely combinatorial circuits, it is also possible to analyze or generate sequential circuits. Instead of a simple truth table a so-called state transition table is created. Each flip-flop thereby appears on the input side and the output side of the state transition table. In this table, on the right-hand side, you can find the next state, which will occur after the next clock signal. This next state depends on the current state of the flip-flops as found at the left-hand side of the table. For an analysis to be possible, the flip-flops must be named.

The following naming convention applies: The following next state of a bit on the right side of the table is indicated by a lowercase 'n+1'. The corresponding current state is indicated by an appended 'n'. If there is a state variable 'A', 'An' indicates the current state and 'An+1' indicates the next state. If, in the truth table on the left and right side, signals are present, which correspond to this pattern it is assumed that the table is a state transition table, and a sequential circuit is generated instead of a combinatorial circuit.

It should be noted that the circuit to be analyzed may contain only purely combinatorial elements in addition to the built-in D and JK flip-flops. If a flip-flop is e.g. made from Nor gates, this circuit is not recognized as a flip-flop and therefore it is not possible to analyse such a circuit.

4. Hardware

4.1. GAL16v8 and GAL22v10

In the circuit generation menu in the truth table there are also functions to generate so-called JEDEC files. This is a special file format that describes the fuse map of a PLD. This JEDEC file

can be written into a corresponding PLD using a special programmer. At the moment, circuits of the type *GAL16v8* and *GAL22v10* or fuse map compatible devices are supported.

4.2. ATF150xAS

The chips in the *ATF150x* family are simple CPLDs with up to 128 macrocells. They are available in a PLCC package, which makes them suitable for laboratory exercises: If an IC is destroyed during exercises, it can simply be replaced. In addition, with the *ATDH1150USB* an easy to use, low-cost programmer is available. This programmer is able to program the *ATF150x* chips in system using a JTAG interface. A suitable evaluation board (*ATF15XX-DK3-U*) is also available. The software *ATMISP*, which is available on the ATMEL/Microchip website, is required for programming the chips.

Unfortunately, the fuse map details are not publicly available so that no suitable fitter for this chip can be integrated in Digital, as is possible with the *GAL16v8* and *GAL22v10* chips.

Therefore, the fitters *fit150[x].exe* provided by ATMEL must be used. These programs create a *JEDEC* file from a suitable *TT2* file which can then be programmed on the chip. Digital starts the fitter automatically every time a *TT2* file is created. For this purpose, the path to the *fit150[n].exe* fitters must be specified in the settings. The created *JEDEC* file can then be opened and programmed directly with *ATMISP*.

For legal reasons the fitter *fit1502.exe* can not be distributed with Digital. However, it can be found in the folder *WinCupl\Fitters* after installing *WinCupl*. *WinCupl* is available on the ATMEL/Microchip website. On Linux systems, the fitters can also be executed by Digital if *wine* is installed.

4.3. Export to VHDL or Verilog

A circuit can be exported to VHDL or Verilog. A file is generated which contains the complete description of the circuit. The generated VHDL code was tested with Xilinx Vivado and the open source VHDL simulator ghdl. The Verilog code is tested with the Verilog simulator Icarus Verilog.

If a circuit contains test cases, the test data is used to generate a HDL test bench. This can be used to check the correct function of the circuit in a HDL simulation.

Additional files which are needed by special boards can be created. At present only the BASYS3 board and the Mimas boards Mimas and Mimas V2 are supported. A constraints file is created, which contains the assignment of the pins. The description of the pins can be found in the boards data sheet, and must be entered as a pin number for the inputs and outputs.

At a BASYS3 board, if the circuit clock frequency is low, a frequency divider is integrated into the HDL code to divide the board clock accordingly. If the clock frequency selected in the circuit exceeds 37kHz, the MMCM unit of the Artix-7 is used for clock generation. This ensures that the FPGA resources provided for the clock distribution are used. This allows the included example processor to run at 20MHz, and if you can do without the multiplier, 30HMz is also possible.

Also at the Mimas-Boards the Spartan 6 DCM is utilized for the clock generation.

If a circuit is to run on a BASYS3 board, a new project can be created in Vivado. The generated VHDL file and the constraints file must be added to the project. Once the project has been created, the bitstream can be generated and the Hardware-Manager can be used to program a BASYS3 board.

To create the required constraints file, the circuit must contain a text field with the text "Board: BASYS3", "Board: MimasV1" or "Board: MimasV2".

5. Frequently asked Questions

How to move a wire?

Select one of the end points with the rectangular selection. Then move this point using the mouse. You can also select a wire with CTRL + mouse button.

How to delete a wire?

Select one of the end points and press *DEL* or click on the trashcan. You can also select a wire with CTRL + mouse button.

How to move a component including all the connected wires?

Select the component with the rectangular selection. The selection must include the entire component. Then move the component including the wires using the mouse.

There is a component not connected to a wire, even though the pins are on the wire.

A pin is only connected to a wire if the wire has an endpoint at the pin.

If the names of the pins in a circuit are long, the names are no longer readable when the circuit is embedded. What can I do?

The width of the block can be increased using the menu item *Edit→Edit circuit attributes*.

The pins in an embedded circuit have an non-optimal order. How can this be changed?

The sequence can be changed using the menu entry *Edit→Order inputs* or *Edit→Order outputs*.

When the simulation is started, a wire becomes gray. What does that mean?

The colors light green and dark green are used to represent high and low state. Gray means the wire is in high Z state.

I have a truth table. How to calculate the minimized boolean equations?

In the menu *Analysis* select the entry *Synthesise*. Then enter the truth table. At the bottom of the window you can find the matching boolean equation. If you enter more than one dependent variable, a new window opens in which all boolean equations are shown.

I have entered a truth table, but there is more then one boolean equation shown. Which of them is the correct one?

Minimizing a boolean equation can result in many equations, describing the same function. Digital shows all of them and they all create the same truth table. There may be differences depending on the "don't cares" in the truth table.

I have a truth table. How to create a circuit representing the truth table?

In the menu *Analysis* select the entry *Synthesise*. Then enter the truth table. You can edit the table using the *New* or *Add Columns* menus. In the menu *Create* you can create a circuit using the *Circuit* item.

How to edit a signal's name in the truth table?

Right click on the name in the table header to edit the name.

I have a boolean equation. How to create a circuit?

In the menu *Analysis* select the entry *Expression*. Then enter the expression.

How to create a truth table from a boolean equation?

In the menu *Analysis* select the entry *Expression*. Then enter the expression. Then create a circuit and in the menu *Analysis* use the entry *Analysis* to create the truth table.

How to create a JEDEC file from a given circuit?

In the menu *Analysis* select the entry *Analysis*. Then in the menu *Create* in the new window choose the correct device in the sub menu *Device*.

When creating a JEDEC file: How to assign a pin number to a certain signal?

At the corresponding inputs and outputs you can enter a pin number in the settings dialog of the pin.

I have created a JEDEC file. How to program it to a *GAL16v8* or *GAL22v10*?

To program such a chip a special programmer hardware is necessary.

6. Keyboard Shortcuts

Space	Starts or stops the simulation.
F6	Opens the measurement table dialog.
F7	Run to Break
F8	Execute test cases
C	A single clock step (Works only in a running simulation and only if there is a single clock component).
F9	Analysis of the circuit
CTRL-A	Select all.
CTRL-X	Cuts the selected components to the clipboard.
CTRL-C	Copys the selected components to the clipboard.
CTRL-V	Inserts the components from the clipboard.
CTRL-D	Duplicate the current selection without modifying the clipboard.
R	While inserting this rotates the components.
L	Inserts the last inserted component again.
CTRL-N	New circuit.
CTRL-O	Open circuit.
CTRL-S	Save the circuit.
CTRL-Z	Undo last modification.
CTRL-Y	Redo the last undone modification.
P	Programs a diode or a FG-FET.
D	While drawing a wire switches to the diagonal mode.
F	While drawing a line flips the orientation.
S	Splits a single wire into two wires.
ESC	Abort the current action.
Del	Removes the selected components.
Backspace	Removes the selected components.
+	Increases the number of inputs at the component the mouse points to.
-	Decreases the number of inputs at the component the mouse points to.
CTRL +	Zoom In
CTRL -	Zoom Out
F1	Fit to size
F5	Show or hide the components tree view

B Components

1. Logic
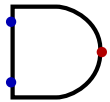
1.1. And

Binary AND gate. Returns high only if all inputs are also set high. It is also possible to use buses with several bits as inputs and output. In this case, a bitwise AND is executed. This means that the lowest bits of all inputs are connected with AND and is output as the lowest bit at the output. The same happens with bit 1, bit 2 and so on.

Inputs

In_1
    The 1. input value for the logic operation.
In_2
    The 2. input value for the logic operation.

Outputs

out
    Returns the result of the logic operation.

Attributes

Data Bits
    Number of data bits used.
Number of Inputs
    The Number of Inputs used. Every input needs to be connected.
inverted Inputs
    You can select the inputs that are to be inverted.
Rotation
    The orientation of the Element in the circuit.

1.2. NAnd

A combination of AND and NOT. Returns 0 only if all inputs are set to 1. If one of the inputs is set to 0 the output is set to 1. It is also possible to use buses with several bits per input. In this case, the operation is applied to each bit of the inputs.

Inputs

In_1
The 1. input value for the logic operation.
In_2
The 2. input value for the logic operation.

Outputs

out
Returns the result of the logic operation.

Attributes

Data Bits
Number of data bits used.
Number of Inputs
The Number of Inputs used. Every input needs to be connected.
inverted Inputs
You can select the inputs that are to be inverted.
Rotation
The orientation of the Element in the circuit.
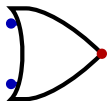
1.3. Or

Binary OR gate. Returns a 1 if one of the inputs is set to 1. If all inputs are set to 0 the output is also set to 0. It is also possible to use buses with several bits as inputs and output. In this case, a bitwise OR is executed. This means that the lowest bits of all inputs are connected with OR and is output as the lowest bit at the output. The same happens with bit 1, bit 2 and so on.

Inputs

In_1
The 1. input value for the logic operation.
In_2
The 2. input value for the logic operation.

Outputs

out
Returns the result of the logic operation.

Attributes

Data Bits
Number of data bits used.
Number of Inputs
The Number of Inputs used. Every input needs to be connected.
inverted Inputs
You can select the inputs that are to be inverted.

Rotation
    The orientation of the Element in the circuit.

1.4. NOr

A combination of OR and NOT. Returns a 0 if one of the inputs is set to 1. If all inputs are set to 0 the output is also set to 1. It is also possible to use buses with several bits per input. In this case, the operation is applied to each bit of the inputs.

Inputs

    In_1
        The 1. input value for the logic operation.
    In_2
        The 2. input value for the logic operation.

Outputs

    out
        Returns the result of the logic operation.

Attributes

    Data Bits
        Number of data bits used.
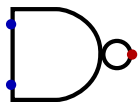   Number of Inputs
        The Number of Inputs used. Every input needs to be connected.
    inverted Inputs
        You can select the inputs that are to be inverted.
    Rotation
        The orientation of the Element in the circuit.
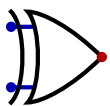
1.5. XOr

If two inputs are used, the output is 0 if both input bits are equal. Otherwise the output in set to 1. If more than two inputs are used, it behaves like cascaded XOR gates (A XOR B XOR C = (A XOR B) XOR C). It is also possible to use buses with several bits per input. In this case, the operation is applied to each bit of the inputs.

Inputs

    In_1
        The 1. input value for the logic operation.
    In_2
        The 2. input value for the logic operation.

Outputs

out
Returns the result of the logic operation.

Attributes

Data Bits
Number of data bits used.
Number of Inputs
The Number of Inputs used. Every input needs to be connected.
inverted Inputs
You can select the inputs that are to be inverted.
Rotation
The orientation of the Element in the circuit.
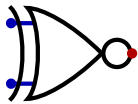
1.6. XNOr

A combination of XOR and NOT. The inputs are combined with the XOR operation. The result of this operation than is inverted. It is also possible to use buses with several bits per input. In this case, the operation is applied to each bit of the inputs.

Inputs

In_1
The 1. input value for the logic operation.
In_2
The 2. input value for the logic operation.

Outputs

out
Returns the result of the logic operation.

Attributes

Data Bits
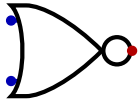Number of data bits used.
Number of Inputs
The Number of Inputs used. Every input needs to be connected.
inverted Inputs
You can select the inputs that are to be inverted.
Rotation
The orientation of the Element in the circuit.

1.7. Not

Inverts the input value. A 1 becomes a 0 and a 0 becomes 1. It is also possible to use a bus with several bits per input. In this case, the operation is applied to each bit of the inputs.

Inputs

in
   The input of the NOT gate.

Outputs

out
   The inverted input value.

Attributes

Data Bits
   Number of data bits used.
Rotation
   The orientation of the Element in the circuit.
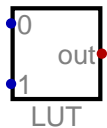
1.8. LookUpTable

Gets the output value from a stored table. So this gate can emulate every combinatorial gate.

Inputs

0
   Input 0. This input in combination with all other inputs defines the address of the stored value to be returned.
1
   Input 1. This input in combination with all other inputs defines the address of the stored value to be returned.

Outputs

out
   Returns the stored value at the address set via the inputs.

Attributes

Data Bits
   Number of data bits used.
Number of Inputs
   The Number of Inputs used. Every input needs to be connected.
Data
   The values stored in this element.

Rotation
>The orientation of the Element in the circuit.

2. IO

2.1. Out

Can be used to display an output signal in a circuit. This element is also used to connect a circuit to an embedding circuit. In this case the connection is bidirectional. Is also used to assign an pin number, if code for a CPLD or FPGA is generated.

Inputs

in
>This value is used for the output connection.

Attributes

Data Bits
>Number of data bits used.

Label
>The name of this element.

Description
>A short description of this element and its usage.

Rotation
>The orientation of the Element in the circuit.

Number Format
>The format used to show the numbers.

Pin number
>Number of this pin. Used for the representation of a circuit as a DIL package and the pin assignment when programming a CPLD. If there are several bits, all pin numbers can be specified as a comma-separated list.

2.2. LED

A LED can be used to visualize an output value. Accepts a single bit. Lights up if the input is set to 1.

Inputs

in
>LED Input. LED lights up if the input is set to 1.

Attributes

Label
>The name of this element.

Color
>The Color of the element.

Rotation
  The orientation of the Element in the circuit.
Size
  The size of the LED in the circuit.

2.3. Input

Can be used to interactively manipulate an input signal in a circuit with the mouse. This element is also used to connect a circuit to an embedding circuit. In this case the connection is bidirectional. Is also used to assign an pin number, if code for a CPLD or FPGA is generated.

Outputs

out
  Gives the value which is connected to this input.

Attributes

Data Bits
  Number of data bits used.
Label
  The name of this element.
Description
  A short description of this element and its usage.
Rotation
  The orientation of the Element in the circuit.
Default
  This value is set if the circuit is started. A "Z" means high-z state.
Is three-state input
  If set the input is allowed to be in high-z state. At the input component this is also allowed if high-z ("Z") is set as the default value.
Number Format
  The format used to show the numbers.
Pin number
  Number of this pin. Used for the representation of a circuit as a DIL package and the pin assignment when programming a CPLD. If there are several bits, all pin numbers can be specified as a comma-separated list.

2.4. Clock Input

A clock signal. Its possible to control it by a real-time clock. Depending on the complexity of the circuit, the clock frequency achieved may be less than the selected value. If the frequency is greater than 50Hz, the graphic representation of the circuit will no longer be updated at every clock cycle so that the wire colors will no longer be updated. If the real-time clock is not activated, the clock can be controlled by mouse clicks. Is also used to assign an pin number, if code for a CPLD or FPGA is generated.

Outputs

C

Switches between 0 and 1 with the selected clock frequency.

Attributes

Label

The name of this element.

Start real time clock

If enabled the runtime clock is started when the circuit is started

Frequency/Hz

The real time frequency used for the real time clock

Rotation

The orientation of the Element in the circuit.

Pin number

Number of this pin. Used for the representation of a circuit as a DIL package and the pin assignment when programming a CPLD. If there are several bits, all pin numbers can be specified as a comma-separated list.

2.5. Button

A simple push button which goes back to its original state when it is released.

Outputs

out

The output signal of the button.

Attributes

Label

The name of this element.

Active Low

If selected the output is low if the component is active.

Map to keyboard

Button is mapped to the keyboard. To use the cursor keys use UP, DOWN, LEFT or RIGHT as label.

Rotation

The orientation of the Element in the circuit.

2.6. DIP Switch

Simple DIP switch that can output either high or low.

Outputs

out

The output value of the switch.

Attributes

Label
The name of this element.
Description
A short description of this element and its usage.
Rotation
The orientation of the Element in the circuit.
Output is High
The default output value of the DIP switch when the simulation starts.

Text

2.7. Text

Shows a text in the circuit. Does not affect the simulation. The text can be changed in the attribute dialog.

Attributes

Description
A short description of this element and its usage.
Font Size
Sets the font size to use for this text.
Rotation
The orientation of the Element in the circuit.
Orientation
Position of the coordinate relative to the text.

•?

2.8. Probe

A measurement value which can be shown in the data graph or measurement table. This component can be used to easily observe values from embedded circuits. Does not affect the simulation.

Inputs

in
The measurement value.

Attributes

Label
The name of this element.
Rotation
The orientation of the Element in the circuit.
Number Format
The format used to show the numbers.

3. IO - more

3.1. Light Bulb

Light bulb with two connections. If a current flows, the bulb lights up! The direction of the current does not matter. The lamp lights when the inputs have different values. The bulb behaves similar to an XOr gate.

Inputs

A
Connection
B
Connection

Attributes

Label
The name of this element.
Color
The Color of the element.
Rotation
The orientation of the Element in the circuit.

3.2. LED with two connections.

LED with connections for the cathode and the anode. The LED lights up, if the anode is connected to high and the cathode is connected to low.

Inputs

A
The anode connection of the LED.
C
The cathode connection of the LED.

Attributes

Label
The name of this element.
Color
The Color of the element.
Rotation
The orientation of the Element in the circuit.
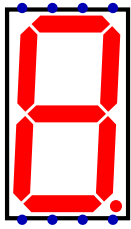
3.3. Seven-Segment Display

Seven Segment Display, every segment has its own control input.

Inputs

   a

   This input controls the upper, horizontal line.

   b

   This input controls the upper, right, vertical line.

   c

   This input controls the lower, right, vertical line.

   d

   This input controls the lower horizontal line.

   e

   This input controls the lower, left, vertical line.

   f

   This input controls the upper, left, vertical line.

   g

   This input controls the middle, horizontal line.

   dp

   This input controls the decimal point.

Attributes

   Color

   The Color of the element.

   Common Cathode

   If selected the common cathode input is also simulated.

   Avoid Flicker

   It is not possible to increase the frequency so much that the flickering disappears.
   With this option you can stabilize the display by keeping the LEDs on until the
   common cathode goes down again. This simulates a frequency above the critical
   flicker fusion frequency.
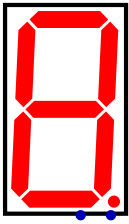
3.4. Seven-Segment-Hex Display

Seven Segment Display with a 4 bit hex input

Inputs

    d

        The value at this input is visualized at the display.

    dp

        This input controls the decimal point.

Attributes

    Color

        The Color of the element.
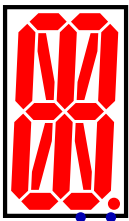
3.5. 16-Segement Display

The LED input has 16 bits which control the segments. The second input controls the decimal point.

Inputs

    led

        16-bit bus for driving the LEDs.

    dp

        This input controls the decimal point.

Attributes

    Color

        The Color of the element.

r-data
c-addr
LED-Matrix
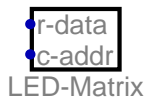
3.6. LED-Matrix

A matrix of LEDs. The LEDs are shown in a separate window. The LEDs of a column of the display are controlled by a data word. At another input, the current column is selected. So a multiplexed display is realized. The LEDs are able to light up indefinitely in the simulation to prevent the display from flickering.

Inputs

r-data
    The row state of the LEDs of a column. Each bit in this data word represents the state of a row of the current column.
c-addr
    The number of the current column whose state is currently visible at the other input.

Attributes

Label
    The name of this element.
Rows
    Specifies the number of rows by specifying the number of bits of the row word.
Address bits of columns
    Addresses the individual columns. Three bits means eight columns.
Color
    The Color of the element.
Avoid Flicker
    It is not possible to increase the frequency so much that the flickering disappears. With this option you can stabilize the display by keeping the LEDs on until the common cathode goes down again. This simulates a frequency above the critical flicker fusion frequency.
Rotation
    The orientation of the Element in the circuit.

A
B
C
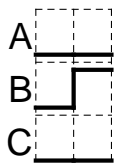
3.7. Data Graph

Shows a data plot inside of the circuit panel. You can plot complete clock cycles or single gate changes. Does not affect the simulation.

Attributes

Show single gate steps
    Shows all single step steps in the graphic.

Max number of steps to show
> The maximal number of values stored. If the maximum number is reached, the oldest values are discarded.
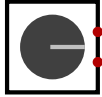
3.8. Rotary Encoder

Rotary knob with rotary encoder. Used to detect rotational movements.

Outputs

A
> encoder signal A

B
> encoder signal B

Attributes

Label
> The name of this element.

Rotation
> The orientation of the Element in the circuit.
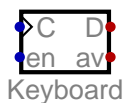
3.9. Keyboard

A keyboard that can be used to enter text. This component buffers the input, which can then be read out. A separate window is opened for the text input.

Inputs

C
> Clock. A rising edge removes the oldest character from the buffer.

en
> If high the output D is active and one character is output. It also enables the clock input.

Outputs

D
> The last typed character, or zero if no character is available.

av
> This output indicates that characters are available. It can be used to trigger an interrupt.

Attributes

Label
> The name of this element.

inverted Inputs
>   You can select the inputs that are to be inverted.
Rotation
>   The orientation of the Element in the circuit.
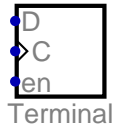
Terminal

3.10. Terminal

You can write ASCII characters to this terminal. The terminal opens its own window to visualize the output.

Inputs

D
>   The data to write to the terminal
C
>   Clock. A rising edge writes the value at the input to the terminal window.
en
>   A high at this input enables the clock input.

Attributes

Characters per line
>   The number of characters shown in a single line.
Lines
>   The number of lines to show.
Label
>   The name of this element.
Rotation
>   The orientation of the Element in the circuit.

4. Wires

4.1. Ground

A connection to ground. Output is always zero.

Outputs

out
>   Output always returns 0.

Attributes

Data Bits
>   Number of data bits used.
Rotation
>   The orientation of the Element in the circuit.

4.2. Supply voltage

A connection to the supply voltage. Output is always one.

Outputs

out
This output always returns 1.

Attributes

Data Bits
Number of data bits used.
Rotation
The orientation of the Element in the circuit.

1•

4.3. Constant value

A component which returns a given value as a simple constant value. The value can be set in the attribute dialog.

Outputs

out
Returns the given value as a constant.

Attributes

Data Bits
Number of data bits used.
Value
The value of the constant.
Rotation
The orientation of the Element in the circuit.
Number Format
The format used to show the numbers.

4.4. Tunnel

Connects components without a wire. All tunnel elements, which have the same net name, are connected together. Works only locally, so it is not possible to connect different circuits.

Inputs

in
The connection to the tunnel.

Attributes

Net name
    All nets with identical name are connected together.
Rotation
    The orientation of the Element in the circuit.

4.5. Splitter

Splits or creates a wire bundle or a data bus with more than one bit. With a bus it is e.g. possible to generate 16-bit connections without having to route 16 individual wires. All 16 connections can be merged into one wire.

Inputs

0-3
    The input bits 0-3.
4-7
    The input bits 4-7.

Outputs

0-7
    The output bits 0-7.

Attributes

Input Splitting
    If e.g. four bits, two bits and two further bits are to be used as inputs, this can be configured with "4,2,2". The number indicates the number of bits. For convenience, the asterisk can be used: 16 bits can be configured with "[Bits]*[Number]" as "1*16". It is also possible to specify the bits to be used directly and in any order. For example, "4-7,0-3" configures bits 4-7 and 0-3. This notation allows any bit arrangement. The input bits must be specified completely and unambiguously.
Output splitting
    If e.g. four bits, two bits and two further bits are to be used as outputs, this can be configured with "4,2,2". The number indicates the number of bits. For convenience, the asterisk can be used: 16 bits can be configured with "[Bits]*[Number]" as "1*16". It is also possible to specify the bits to be used directly and in any order. For example, "4-7,0-3" configures bits 4-7 and 0-3. This notation allows any bit arrangement. Output bits can also be output several times: "0-7,1-6,4-7"
Rotation
    The orientation of the Element in the circuit.
Spreading
    Configures the spread of the inputs and outputs in the circuit.

4.6. Driver

A driver can be used to connect a signal value to another wire. The driver is controlled by the sel input. If the sel input is low, the output is in high z state. If the sel input is high, the output is set to the input value.

Inputs

in
> The input value of the driver.

sel
> Pin to control the driver. If its value is 1 the input is set to the output. If the value is 0, the output is in high z state.

Outputs

out
> If the sel input is 1 the input is given to this output. If the sel input is 0, this output is in high z state.

Attributes

Data Bits
> Number of data bits used.

Flip selector position
> This option allows you to move te selector pin to the opposite side of the plexer.

Rotation
> The orientation of the Element in the circuit.

4.7. Driver, inverted select

A driver can be used to connect a data word to another line. The driver is controlled by the sel input. If the sel input is high, the output is in high z state. If the sel input is low, the output is set to the input value.

Inputs

in
> The input value of the driver.

sel
> Pin to control the driver. If its value is 0 the input is given to the output. If the value is 1, the output is in high z state.

Outputs

out
> If the sel input is 1 the input is given to this output. If the sel input is 0, this output is in high z state.

Attributes

Data Bits
Number of data bits used.
Flip selector position
This option allows you to move te selector pin to the opposite side of the plexer.
Rotation
The orientation of the Element in the circuit.

4.8. Delay

Delays the signal by on propagation delay time. Delays a signal for an adjustable number of gate delays. All other components in Digital have a gate delay of on propagation delay time. This component can be used to realize any necessary propagation delay.

Inputs

in
Input of the signal to be delayed.

Outputs

out
The input signal delayed by one gate delay time.

Attributes

Data Bits
Number of data bits used.
Duration
Delay time in units of the common gate propagation delay.
Rotation
The orientation of the Element in the circuit.

4.9. Pull-Up Resistor

If a net is in a HighZ state, this resistor pulls the net to high. In any other case this component has no effect.

Outputs

out
A "weak high".

Attributes

Data Bits
Number of data bits used.
Rotation
The orientation of the Element in the circuit.

4.10. Pull-Down Resistor

If the net is in a HighZ state, this resistor pulls the net to ground. In any other case this component has no effect.

Outputs

out
A "weak low".

Attributes

Data Bits
Number of data bits used.
Rotation
The orientation of the Element in the circuit.

5. Plexers

5.1. Multiplexer

A component which uses the value of the sel pin to decide which input value is set to the output.

Inputs

sel
This input is used to select the data input which is output.
in_0
The 0. data input of the multiplexer.
in_1
The 1. data input of the multiplexer.

Outputs

out
The value of the selected input.

Attributes

Data Bits
Number of data bits used.
Number of Selector Bits
Number of bits used for the selector input.
Flip selector position
This option allows you to move te selector pin to the opposite side of the plexer.

Rotation

> The orientation of the Element in the circuit.

5.2. Demultiplexer

A component that can output the input value to one of the outputs. The other outputs are set to the default value.

Inputs

sel
> This pin selects the output to use.

in
> The value of this input is given to the selected data output.

Outputs

out_0
> Data output 0.

out_1
> Data output 1.

Attributes

Data Bits
> Number of data bits used.

Number of Selector Bits
> Number of bits used for the selector input.

Flip selector position
> This option allows you to move te selector pin to the opposite side of the plexer.

Rotation
> The orientation of the Element in the circuit.

Default
> This value is set if the circuit is started. At the demultiplexer, this value is set for the non-selected outputs.

5.3. Decoder

One selectable output pin is 1, all other outputs are set to 0.

Inputs

sel
> This input selects the enabled output. The selected output is set to 1. All other outputs are set to 0.

Outputs

    out_0

        Output 0. This output is 1 if selected by the sel input.

    out_1

        Output 1. This output is 1 if selected by the sel input.

Attributes

    Number of Selector Bits

        Number of bits used for the selector input.

    Flip selector position

        This option allows you to move te selector pin to the opposite side of the plexer.

    Rotation

        The orientation of the Element in the circuit.

5.4. Bit Selector

Selects a single bit from a data bus.

Inputs

    in

        The input bus

    sel

        This input selects the bit

Outputs

    out

        The selected bit.

Attributes

    Number of Selector Bits
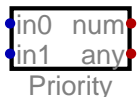
        Number of bits used for the selector input.

    Flip selector position

        This option allows you to move te selector pin to the opposite side of the plexer.

    Rotation

        The orientation of the Element in the circuit.

5.5. Priority Encoder

If one of the inputs is set, its number is output. If several inputs are set at the same time, the highest number is output.

Inputs

in0
>
> The 0. input of the priority encoder.

in1
>
> The 1. input of the priority encoder.

Outputs

num
>
> Number of the set input.

any
>
> If this output is set, at least one of the inputs is set.

Attributes

Label
>
> The name of this element.

Number of Selector Bits
>
> Number of bits used for the selector input.

Rotation
>
> The orientation of the Element in the circuit.

6. Flip-Flops
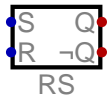
6.1. RS-Flip-flop

A component to store a single bit. Provides the functions "set" and "reset" to set or reset the stored bit. If both inputs are switched to one, both outputs also output a one. If both inputs switch back to zero at the same time, the final state is random.

Inputs

S
>
> The set input.

R
>
> The reset input.

Outputs

Q
>
> Returns the stored value.

¬Q
>
> Returns the inverted stored value.

Attributes

Label
>
> The name of this element.

inverted Inputs
>
> You can select the inputs that are to be inverted.

Rotation
> The orientation of the Element in the circuit.

Default
> This value is set if the circuit is started. At the demultiplexer, this value is set for the non-selected outputs.

Use as measurement value
> Is set the value is a measurement value and appears in the graph and data table. In addition, a label must be specified that can serve as identification of the value.

6.2. RS-Flip-flop, clocked

A component to store a single bit. Provides the functions "set" and "reset" to set or reset the stored bit. If both inputs (S, R) are set at the rising edge of the clock, the final state is random.

Inputs

S
> The set input.

C
> The clock input. A rising edge initiates a state transition.

R
> The reset input.

Outputs

Q
> Returns the stored value.

¬Q
> Returns the inverted stored value.

Attributes

Label
> The name of this element.

inverted Inputs
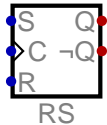> You can select the inputs that are to be inverted.

Rotation
> The orientation of the Element in the circuit.

Default
> This value is set if the circuit is started. At the demultiplexer, this value is set for the non-selected outputs.

Use as measurement value
> Is set the value is a measurement value and appears in the graph and data table. In addition, a label must be specified that can serve as identification of the value.

JK

6.3. JK-Flip-flop

Has the possibility to store (J=K=0), set (J=1, K=0), reset (J=0, K=1) or toggle (J=K=1) the stored value. A change of state takes place only at a rising edge at the clock input C.

Inputs

J
> The set input of the flip-flop.

C
> The clock input. A rising edge initiates a state change.

K
> The reset input of the flip-flop.

Outputs

Q
> Returns the stored value.

¬Q
> Returns the inverted stored value.

Attributes

Label
> The name of this element.

inverted Inputs
> You can select the inputs that are to be inverted.
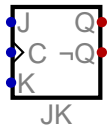
Rotation
> The orientation of the Element in the circuit.

Default
> This value is set if the circuit is started. At the demultiplexer, this value is set for the non-selected outputs.

Use as measurement value
> Is set the value is a measurement value and appears in the graph and data table. In addition, a label must be specified that can serve as identification of the value.

D

6.4. D-Flip-flop

A component used to store a value. The value on pin D is stored on a rising edge of the clock pin C. The bit width can be selected, which allows to store multiple bits.

Inputs

D

Input of the bit to be stored.

C

Clock pin to store a value. The value on input D is stored on a rising edge of this pin.

Outputs

Q

Returns the stored value.

¬Q

Returns the inverted stored value.

Attributes

Data Bits

Number of data bits used.

Label

The name of this element.

inverted Inputs

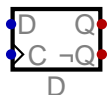You can select the inputs that are to be inverted.

Rotation

The orientation of the Element in the circuit.

Default

This value is set if the circuit is started. At the demultiplexer, this value is set for the non-selected outputs.

Use as measurement value

Is set the value is a measurement value and appears in the graph and data table. In addition, a label must be specified that can serve as identification of the value.

6.5. T-Flip-Flop

Stores a single bit. Toggles the state on a rising edge at input C.

Inputs

T

Enables the toggle function.

C

Clock input. A rising edge toggles the output, if input T is set to 1.

Outputs

Q

Returns the stored value.

¬Q

Returns the inverted stored value.

Attributes

Label

   The name of this element.

Enable Input

   If set an enable input (T) is available.

inverted Inputs

   You can select the inputs that are to be inverted.
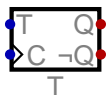
Rotation

   The orientation of the Element in the circuit.

Default

   This value is set if the circuit is started. At the demultiplexer, this value is set for the non-selected outputs.

Use as measurement value

   Is set the value is a measurement value and appears in the graph and data table. In addition, a label must be specified that can serve as identification of the value.
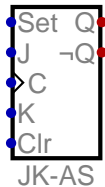
6.6. JK-Flip-flop, asynchronous

Has the possibility to store (J=K=0), set (J=1, K=0), reset (J=0, K=1) or toggle (J=K=1) the stored value. A change of state takes place only at a rising edge at the clock input C. There are two additional inputs which set or reset the state immediately without a clock signal.

Inputs

   Set

      asynchronous set. A high value at this input sets the flip-flop.

   J

      The set input of the flip-flop.

   C

      The Clock input. A rising edge initiates a state change.

   K

      The reset input of the flip-flop.

   Clr

      asynchronous clear. A high value at this input clears the flip-flop.

Outputs

   Q

      Returns the stored value.

   ¬Q

      Returns the inverted stored value.

Attributes

   Label

      The name of this element.

   inverted Inputs

   You can select the inputs that are to be inverted.

Rotation

    The orientation of the Element in the circuit.

Default

    This value is set if the circuit is started. At the demultiplexer, this value is set for the non-selected outputs.

Use as measurement value

    Is set the value is a measurement value and appears in the graph and data table. In addition, a label must be specified that can serve as identification of the value.

```
Set  Q
D   ¬Q
C
Clr
  D-AS
```
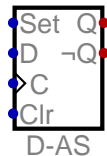
6.7. D-Flip-flop, asynchronous

A component used to store a value. The value on pin D is stored on a rising edge of the clock pin C. There are two additional inputs which set or reset the state immediately without a clock signal. The bit width can be selected, which allows to store multiple bits.

Inputs

Set

    asynchronous set. If set to one, all stored bits are set to one.

D

    Input of the bit to be stored.

C

    Control pin to store a bit. The bit on input D is stored on a rising edge of this pin.

Clr

    asynchronous clear. If set to one, all stored bits are set to zero.

Outputs

Q

    Returns the stored value.

¬Q

    Returns the inverted stored value.

Attributes

Data Bits

    Number of data bits used.

Label

    The name of this element.

inverted Inputs

    You can select the inputs that are to be inverted.

Rotation

    The orientation of the Element in the circuit.

Default

    This value is set if the circuit is started. At the demultiplexer, this value is set for the non-selected outputs.

Use as measurement value
> Is set the value is a measurement value and appears in the graph and data table. In addition, a label must be specified that can serve as identification of the value.
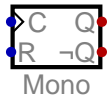
6.8. Monoflop

The monoflop is set at a rising edge at the clock input. After a configurable delay time, the monoflop will be cleared automatically. The monoflop is retriggerable. It can only be used if there is exactly one clock component present in the circuit. This clock component is used as time base to measure the time delay.

Inputs

C
> The Clock input. A rising edge sets the monoflop.

R
> Reset Input. A high value clears the monoflop.

Outputs

Q
> output

¬Q
> inverted output

Attributes

Label
> The name of this element.

Pulse Width
> The puls width is measured in clock cycles.

inverted Inputs
> You can select the inputs that are to be inverted.

Rotation
> The orientation of the Element in the circuit.

Default
> This value is set if the circuit is started. At the demultiplexer, this value is set for the non-selected outputs.

Use as measurement value
> Is set the value is a measurement value and appears in the graph and data table. In addition, a label must be specified that can serve as identification of the value.

7. Memory - RAM

RAM
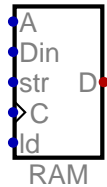
7.1. RAM, separated Ports

A RAM module with separate inputs for storing and output for reading the stored data.

Inputs

A
: The address to read from or write to.

Din
: The data to be stored in the RAM.

str
: If this input is high and when the clock becomes high, the the data is stored.

C
: Clock input

ld
: If this input is high the output is activated and the data is visible at the output.

Outputs

D
: The data output pin

Attributes

Data Bits
: Number of data bits used.

Address Bits
: Number of address bits used.

Label
: The name of this element.

Rotation
: The orientation of the Element in the circuit.

Program Memory
: Makes this ROM to program memory. So it can be accessed by an external IDE.

RAM

7.2. RAM, bidirectional Port

A RAM module with a bidirectional pin for reading and writing the data.

Inputs

A
  The address to read and write.
str
  If this input is high when the clock becomes high, the the data is stored.
C
  Clock
ld
  If this input is high the output is activated and the data is visible at the output.

Outputs

D
  The bidirectional data connection.

Attributes

Data Bits
  Number of data bits used.
Address Bits
  Number of address bits used.
Label
  The name of this element.
Rotation
  The orientation of the Element in the circuit.
Program Memory
  Makes this ROM to program memory. So it can be accessed by an external IDE.

RAM

7.3. RAM, Chip Select

A RAM module with a bidirectional connection for reading and writing the data. If the CS input is low, the component is disabled. This allows to build a larger RAM from some smaller RAMs and a address decoder.

Inputs

A

The address to read and write.

CS

If this input is high, this RAM is enabled. Otherwise the output is always in high Z state.

WE

If set to high the data is written to the RAM.

OE

If this input is high, the stored value is output.

Outputs

D

The bidirectional data connection.

Attributes

Data Bits

Number of data bits used.
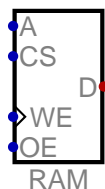
Address Bits

Number of address bits used.

Label

The name of this element.

inverted Inputs

You can select the inputs that are to be inverted.

Rotation

The orientation of the Element in the circuit.

Program Memory

Makes this ROM to program memory. So it can be accessed by an external IDE.
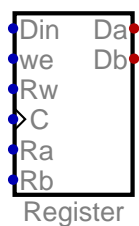
7.4. Register File

Memory with one port that allows to write and two ports that allow to read from the memory simultaneously. Can be used to implement processor registers. Two registers can be read simultaneously and a third can be written.

Inputs

Din
The data to be stored in the register Rw.
we
If this input is high and when the clock becomes high, the the data is stored.
Rw
The register into which the data is written.
C
Clock
Ra
The register which is visible at port a.
Rb
The register which is visible at port b.

Outputs

Da
Output Port a
Db
Output Port b

Attributes

Data Bits
Number of data bits used.
Address Bits
Number of address bits used.
Label
The name of this element.
Rotation
The orientation of the Element in the circuit.
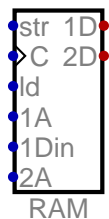
7.5. RAM, Dual Port

RAM with one port that allows to write to and read from the RAM, and a second read only port. This second port can be used to give some graphic logic access to the memory contents. In this way, a processor can write to the RAM, and a graphics logic can simultaneously read from the RAM.

Inputs

str
>   If this input is high and when the clock becomes high, the the data is stored.
C
>   Clock
ld
>   If this input is high the output is activated and the data is visible at the output 1D.
1A
>   The address at which port 1 is read or written.
1Din
>   The data to be stored in the RAM.
2A
>   The address used to read via port 2.

Outputs

1D
>   Output Port 1
2D
>   Output Port 2

Attributes

Data Bits
>   Number of data bits used.
Address Bits
>   Number of address bits used.
Label
>   The name of this element.
Rotation
>   The orientation of the Element in the circuit.
Program Memory
>   Makes this ROM to program memory. So it can be accessed by an external IDE.

7.6. Graphic RAM

Used to show a bitmap graphic. This element behaves like a RAM. In addition it shows its content on a graphic screen. Every pixel is represented by a memory address. The value stored defines the color of the pixel, using a fixed color palette. There are two screen buffers implemented to support page flipping. The input B selects which buffer is shown. Thus, the total memory size is dx * dy * 2 words.
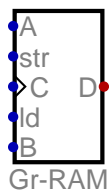
Inputs

A

The address to read and write.

str

If this input is high when the clock becomes high, the the data is stored.

C

Clock

ld

If this input is high the output is activated and the data is visible at the output.

B

Selects the screen buffer to show.

Outputs

D

The bidirectional data connection.

Attributes

Data Bits

Number of data bits used.

Label

The name of this element.

Width in pixels

The screen width in pixels.

Height in pixels

The screen height in pixels.

Rotation

The orientation of the Element in the circuit.

8. Memory - EEPROM
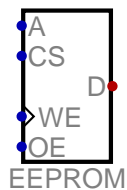
8.1. EEPROM

A EEPROM module with a bidirectional connection for reading and writing the data. If the CS input is low, the component is disabled. The data content is stored like in a ROM. It is thus preserved when the simulation is terminated and restarted.

Inputs

A
The address to read and write.
CS
If this input is high, this EEPROM is enabled. Otherwise the output is always in high Z state.
WE
If set to high the data is written to the EEPROM.
OE
If this input is high, the stored value is output.

Outputs

D
The bidirectional data connection.

Attributes

Data Bits
Number of data bits used.
Address Bits
Number of address bits used.
Label
The name of this element.
inverted Inputs
You can select the inputs that are to be inverted.
Data
The values stored in this element.
Rotation
The orientation of the Element in the circuit.
Program Memory
Makes this ROM to program memory. So it can be accessed by an external IDE.

8.2. EEPROM, separated Ports

A EEPROM module with separate inputs for storing and output for reading the stored data.

Inputs

A
: The address to read from or write to.

Din
: The data to be stored in the RAM.

str
: If this input is high and when the clock becomes high, the the data is stored.

C
: Clock input

ld
: If this input is high the output is activated and the data is visible at the output.

Outputs

D
: The data output pin

Attributes

Data Bits
: Number of data bits used.

Address Bits
: Number of address bits used.

Label
: The name of this element.

Data
: The values stored in this element.

Rotation
: The orientation of the Element in the circuit.

Program Memory
: Makes this ROM to program memory. So it can be accessed by an external IDE.

9. Memory

Reg
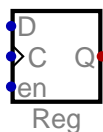
9.1. Register

A component to store values. The bit width of the data word can be selected. Unlike a D flip-flop, the register provides an input which enables the clock.

Inputs

D
: Input pin of the data word to be stored.

C
: Clock input. A rising edge stores the value at the D pin.

en
: Enable pin. Storing a value works only if this pin is set high.

Outputs

Q

Returns the stored value.

Attributes

Data Bits

Number of data bits used.

Label

The name of this element.

inverted Inputs

You can select the inputs that are to be inverted.

Rotation

The orientation of the Element in the circuit.

Program Counter

Makes this register a program counter. The value of this register is returned to the external assembler IDE to mark the current line of code during debugging.

Use as measurement value

Is set the value is a measurement value and appears in the graph and data table. In addition, a label must be specified that can serve as identification of the value.
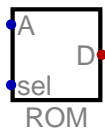
9.2. ROM

A non-volatile memory component. The stored data can be edited in the attributes dialog.

Inputs

A

This pin defines the address of data word to be output.

sel

If the input is high, the output is activated. If it is low, the data output is in high Z state.

Outputs

D

The selected data word if the sel input is high.

Attributes

Data Bits

Number of data bits used.

Address Bits

Number of address bits used.
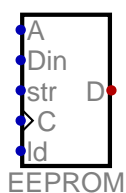
Label

The name of this element.

Data

The values stored in this element.

Rotation

The orientation of the Element in the circuit.

Program Memory
    Makes this ROM to program memory. So it can be accessed by an external IDE.
Reload at model start
    Reloads the hex file every time the model is started.
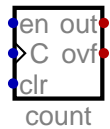
count

9.3. Counter

A simple counter component. The clock input increases the counter. Can be reset back to 0 with the clr input. The number of bits can be set in the attribute dialog.

Inputs

en
    If set to 1 the counter is enabled!
C
    The clock input. A rising edge increases the counter.
clr
    Synchronous reset of the counter if set to 1.

Outputs

out
    Returns the counted value.
ovf
    Overflow output. This pin is set to 1 if the counter is on its maximal value and the en input is set to 1.

Attributes

Data Bits
    Number of data bits used.
inverted Inputs
    You can select the inputs that are to be inverted.
Label
    The name of this element.
Rotation
    The orientation of the Element in the circuit.
Use as measurement value
    Is set the value is a measurement value and appears in the graph and data table. In addition, a label must be specified that can serve as identification of the value.

```
en  out
C  ovf
dir
in
ld
clr
Counter
```
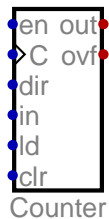
9.4. Counter with preset

A counter whose value can be set. In addition, a maximum value and a counting direction can be specified.

Inputs

en
   If set to 1 the counter is enabled!
C
   The clock input. A rising edge increases the counter.
dir
   Specifies the counting direction. A 0 means upwards.
in
   This data word is stored in the counter when ld is set.
ld
   If set, the value at input 'in' is stored in the counter at the next clock signal.
clr
   Synchronous reset of the counter if set to 1.

Outputs

out
   Returns the counted value.
ovf
   Overflow output. It is set to 1 if the 'en' input is set to 1 and if the counter reaches its maximum value when counting up, or has reached 0 when counting down.

Attributes

Data Bits
   Number of data bits used.
Maximum Value
   If a zero is entered, the maximum possible value is used (all bits are one).
inverted Inputs
   You can select the inputs that are to be inverted.
Label
   The name of this element.
Rotation
   The orientation of the Element in the circuit.
Use as measurement value
   Is set the value is a measurement value and appears in the graph and data table. In addition, a label must be specified that can serve as identification of the value.

10. Arithmetic

Add
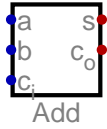
10.1. Adder

A component for simple add calculations. Adds the two integer values from input a and input b (a+b). The result will be incremented by one if the carry input is set.

Inputs

a
First input to add.
b
Second input to add.
c_i
Carry input, if set the result is incremented by one.

Outputs

s
The result of the addition
c_o
Carry output. If set there was an overflow.

Attributes

Label
The name of this element.
Data Bits
Number of data bits used.
Rotation
The orientation of the Element in the circuit.

Sub
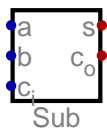
10.2. Subtract

A component for simple subtractions. Subtracts binary numbers on input a and input b (a-b). If the carry input is set to 1 the result is decremented by 1.

Inputs

a

Input a for subtraction.

b

Input b for subtraction.

c_i

Carry input, if set the result is decremented by one.

Outputs

s

Output returns the result of the subtraction.

c_o

Output returns 1 if an overflow occurred.

Attributes

Label

The name of this element.

Data Bits

Number of data bits used.

Rotation

The orientation of the Element in the circuit.
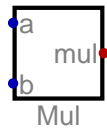
10.3. Multiply

A component for multiplication. Multiplies the integer numbers on input pin a and input pin b. An unsigned multiplication is performed.

Inputs

a

Input a for multiplication.

b

Input b for multiplication.

Outputs

mul

Output for the result of the multiplication.

Attributes

Label

The name of this element.

Data Bits

Number of data bits used.

Rotation

The orientation of the Element in the circuit.

```
in
        out
shift
    shift
```
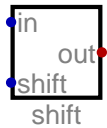
10.4. Barrel shifter

A component for bit shifting. Shifts the input value by the number of bits given by the shift input.

Inputs

    in
        Input with bits to be shifted.
    shift
        Input with shift width.

Outputs

    out
        Output with shifted value.

Attributes

    Label
        The name of this element.
    Data Bits
        Number of data bits used.
    shift input has sign
        shift input data has two complement format
    Direction
        Set direction.
    Mode
        Mode of barrel shifter
    Rotation
        The orientation of the Element in the circuit.

```
a    >
b    =
     <
```
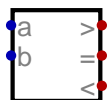
10.5. Comparator

A component for comparing bit values. Compares the binary numbers on input pin a and input pin b and sets the corresponding outputs.

Inputs

    a
        Input a to compare.
    b
        Input b to compare.

Outputs

>

Output is 1 if input a is greater then input b

=

Output is 1 if input a equals input b

<

Output is 1 if input a is less then input b

Attributes

Label

The name of this element.

Data Bits

Number of data bits used.

Signed

If selected the comparison is performed with signed (2th complement) values.

Rotation

The orientation of the Element in the circuit.

`in  out`
Neg
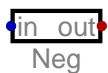
10.6. Negation

Negation in the 2th complement

Inputs

in

Input of the data word to be negated in 2th complement

Outputs

out

Returns the result of the negation in 2th complement.

Attributes

Data Bits

Number of data bits used.

Rotation

The orientation of the Element in the circuit.

`in  out`
SignEx
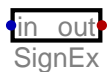
10.7. Sign extender

Increases the bit width of a signed value keeping the values sign. If the input is a single bit, this bit will be output on all output bits.

Inputs

in
Input value. The input bit width must be smaller than the output bit width!

Outputs

out
Extended input value. The input bit width must be smaller than the output bit width!

Attributes

Label
The name of this element.
Input Bit Width
The number of output bits must be greater than the number of input bits.
Output Bit Width
The number of output bits must be greater than the number of input bits.
Rotation
The orientation of the Element in the circuit.

in out
Bit count
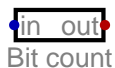
10.8. Bit counter

Returns the number of 1-bits in the input value.

Inputs

in
The input which 1-bits are counted.

Outputs

out
Outputs the number of 1-bits.

Attributes

Data Bits
Number of data bits used.
Rotation
The orientation of the Element in the circuit.

11. Switches

11.1. Diode to VDD

A simplified unidirectional diode, used to pull a wire to VDD. It is used to implement a wired OR. So it is necessary to connect a pull down resistor to the diodes output. In the simulation the diode behaves like an active gate with a trivalent truth table: Is the input high, also the output is high. In all other cases (input is low or high z) the output is in high z state. So two

anti parallel connected diodes can keep each other in high state, which is not possible with real diodes. This is an ideal diode: There is no voltage drop across a forward-biased diode.

Inputs

in

If the input is high also the output is high. In all other cases the output is in high z state.

Outputs

out

If the input is high also the output is high. In all other cases the output is in high z state.

Attributes

Programmed

If set a diode is "blown" or "programmed". At a floating gate FET the floating gate is charged. You can change this setting with the key 'p'.

Rotation

The orientation of the Element in the circuit.

11.2. Diode to Ground

A simplified unidirectional diode, used to pull a wire to ground. It is used to implement a wired AND. So it is necessary to connect a pull up resistor to the diodes output. Is the input low, also the output is low. In the other cases (input is high or high z) the output is in high z state. So two anti parallel connected diodes can keep each other in low state, which is not possible with real diodes. So this is a ideal diode: There is no voltage drop across a forward-biased diode.

Inputs

in

If the input is low also the output is low. In all other cases the output is in high z state.

Outputs

out

If the input is low also the output is low. In all other cases the output is in high z state.

Attributes

Programmed

If set a diode is "blown" or "programmed". At a floating gate FET the floating gate is charged. You can change this setting with the key 'p'.

Rotation

The orientation of the Element in the circuit.

11.3. Switch

Simple switch. There is no gate delay: A signal change is propagated immediately.

Outputs

out1
   One of the connections of the switch.
out2
   One of the connections of the switch.

Attributes

Data Bits
   Number of data bits used.
Label
   The name of this element.
Closed
   Sets the initial state of the switch.
Rotation
   The orientation of the Element in the circuit.

11.4. Fuse

A fuse used to build a one time programmable memory.

Outputs

out1
   out1
out2
   out2

Attributes

Programmed
   If set a diode is "blown" or "programmed". At a floating gate FET the floating gate is charged. You can change this setting with the key 'p'.
Rotation
   The orientation of the Element in the circuit.
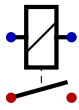
11.5. Relay

A relay is a switch which can be controlled by a coil. If a current flows through the coil, the switch is closed or opened. There is no flyback diode so the current direction does not matter. The switch is actuated if the inputs have different values. The relay behaves similar to an XOr gate.

Inputs

in1
    On of the inputs to control the relay.
in2
    On of the inputs to control the relay.

Outputs

A1
    A1
B1
    B1

Attributes

Data Bits
    Number of data bits used.
Label
    The name of this element.
Pole count
    Number of poles available.
Relay is normally closed.
    If set the relay is closed if the input is low.
Rotation
    The orientation of the Element in the circuit.
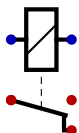
11.6. Double Throw Relay

A relay is a switch which can be controlled by a coil. If a current flows through the coil, the switch is closed or opened. There is no flyback diode so the current direction does not matter. The switch is actuated if the inputs have different values. The relay behaves similar to an XOr gate.

Inputs

in1

On of the inputs to control the relay.

in2

On of the inputs to control the relay.

Outputs

A1

A1

B1

B1

C1

C1

Attributes

Data Bits

Number of data bits used.

Label

The name of this element.

Pole count

Number of poles available.

Rotation

The orientation of the Element in the circuit.

11.7. P-Channel FET

P-Channel Field Effect Transistor. The bulk is connected to the pos. voltage rail and the transistor is simulated without a body diode.

Inputs

G

Gate

Outputs

S

Source

D

Drain

Attributes

Data Bits

Number of data bits used.

Unidirectional

Unidirectional transistors propagate a signal only from source to drain. They are much faster to simulate than bidirectional transistors. Since there is no feedback from drain to source, in this mode, the transistor can not short the connected wires when it is conducting. Thus, this mode is necessary to simulate certain CMOS circuits.

Label

The name of this element.

Rotation

The orientation of the Element in the circuit.

11.8. N-Channel FET

N-Channel Field Effect Transistor. The bulk is connected to ground and the transistor is simulated without a body diode.

Inputs

G

Gate

Outputs

D

Drain

S

Source

Attributes

Data Bits

Number of data bits used.

Unidirectional

Unidirectional transistors propagate a signal only from source to drain. They are much faster to simulate than bidirectional transistors. Since there is no feedback from drain to source, in this mode, the transistor can not short the connected wires when it is conducting. Thus, this mode is necessary to simulate certain CMOS circuits.

Label

The name of this element.

Rotation

The orientation of the Element in the circuit.

11.9. P-Channel floating gate FET

P-Channel Floating Gate Field Effect Transistor. The bulk is connected to ground and the transistor is simulated without a body diode. If there is a charge stored in the floating gate, the fet isn't conducting even if the gate is low.

Inputs

G

Gate

Outputs

S

Source

D

Drain

Attributes

Data Bits

Number of data bits used.

Label

The name of this element.

Programmed

If set a diode is "blown" or "programmed". At a floating gate FET the floating gate is charged. You can change this setting with the key 'p'.

Rotation

The orientation of the Element in the circuit.

11.10. N-Channel floating gate FET

N-Channel Floating Gate Field Effect Transistor. The bulk is connected to ground and the transistor is simulated without a body diode. If there is a charge stored in the floating gate, the fet isn't conducting even if the gate is high.

Inputs

G

Gate

Outputs

D

Drain

S

Source

Attributes

Data Bits

Number of data bits used.

Label

The name of this element.

Programmed

If set a diode is "blown" or "programmed". At a floating gate FET the floating gate is charged. You can change this setting with the key 'p'.

Rotation
  The orientation of the Element in the circuit.

11.11. Transmission-Gate

A real transmission-gate is build from only two transistors. Therefore, it is often used to save transistors during implementation on silicon.

Inputs

  S
    control input.
  ¬S
    inverted control input

Outputs

  A
    input A
  B
    input B

Attributes

  Data Bits
    Number of data bits used.
  Rotation
    The orientation of the Element in the circuit.

12. Misc.

Test

12.1. Test case

Describes a test case. In a test case you can describe how a circuit should behave. It can then be automatically checked whether the behavior of the circuit actually corresponds to this description. If this is not the case, an error message is shown.
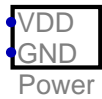
Attributes

  Label
    The name of this element.
  Test data
    The description of the test case. Details of the syntax can be found in the help dialog of the test data editor.

```
VDD
GND
```
Power

12.2. Power

Has no function. Makes sure that VDD and GND are connected. Can be used in 74xx circuits to generate the pins for the voltage supply, which are tested for correct wiring.

Inputs

VDD
Must be connected to VDD!
GND
Must be connected to GND!

Attributes

Label
The name of this element.
Rotation
The orientation of the Element in the circuit.

D   D0
OE

12.3. Bidirectional Splitter

Can be used for data buses and simplifies especially the construction of memory modules in a DIL package, as the implementation of the data bus is simplified.

Inputs

OE
When set, the value at the common data terminal D is output to the bit outputs D[i], if not, the bits D[i] are output to the common output D.

Outputs

D
The common data connection.
D0
The data bit 0 of the bus splitter.

Attributes

Data Bits
Number of data bits used.
Rotation
The orientation of the Element in the circuit.
Spreading
Configures the spread of the inputs and outputs in the circuit.

12.4. Reset

The output of this component is held high during the initialisation of the circuit. After the circuit has stabilized the output goes to low. If the output is inverted it behaves the opposite way.

Outputs

  Reset
      Reset Output.

Attributes

  Label
      The name of this element.
  inverted output
      If selected the output is inverted.
  Rotation
      The orientation of the Element in the circuit.

12.5. Break

If the input of this component becomes high, the fast forward clocking is stopped. You can use the component to implement a BRK assembly instruction. Then you can run a simulated processor until a BRK instruction is reached. Fast forward clocking can only be used if the real-time clock is disabled.

Inputs

  brk
      Stops fast simulation if set to 1.

Attributes

  Label
      The name of this element.
  Timeout cycles
      If this amount of cycles is reached without a break signal, an error is created.
  Rotation
      The orientation of the Element in the circuit.

Async

12.6. Asynchronous Timing

Allows configuration of the timing of an asynchronous sequential circuit such as a Muller-pipeline. The circuit must be started in single gate step mode and must be able to reach a stable state at startup. The sequential circuit can then be started interactively or with a reset gate. It is not allowed to use a regular clock component in this mode.
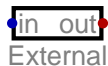
Attributes

Start real time clock
    If enabled the runtime clock is started when the circuit is started
Frequency/Hz
    The real time frequency used for the real time clock

External

12.7. External

Component to execute an external process to calculate the logic function. Is used to specify the behaviour of a component by VHDL or Verilog. The actual simulation of the behavior must be done with an external simulator. At present only the VHDL simulator ghdl and the verilog simulator Icarus Verilog are supported.

Inputs

    in

Outputs

    out

Attributes

Label
    The name of this element.
Width
    With of symbol if this circuit is used as an component in an other circuit.
Inputs
    The inputs of the external process. It is a comma-separated list of signal names. For each signal name, with a colon separated, a number of bits can be specified. The inputs of an 8-bit adder could thus be described as "a:8,b:8,c_in".
Outputs
    The outputs of the external process. It is a comma-separated list of signal names. For each signal name, with a colon separated, a number of bits can be specified. The outputs of an 8-bit adder could thus be described as "s:8,c_out".
Programcode
    The programm code to be executed by the external application.
Application
    Defines which application to use.

C Library

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 high

7448: BCD to 7-segment decoder, active low

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

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

74191: Presettable synchronous 4-bit binary up/down counter

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

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

74540: octal buffer/line driver, inverted

74541: octal buffer/line driver

74590: 8-bit binary counter with tri-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

744075: triple 3-input OR gate