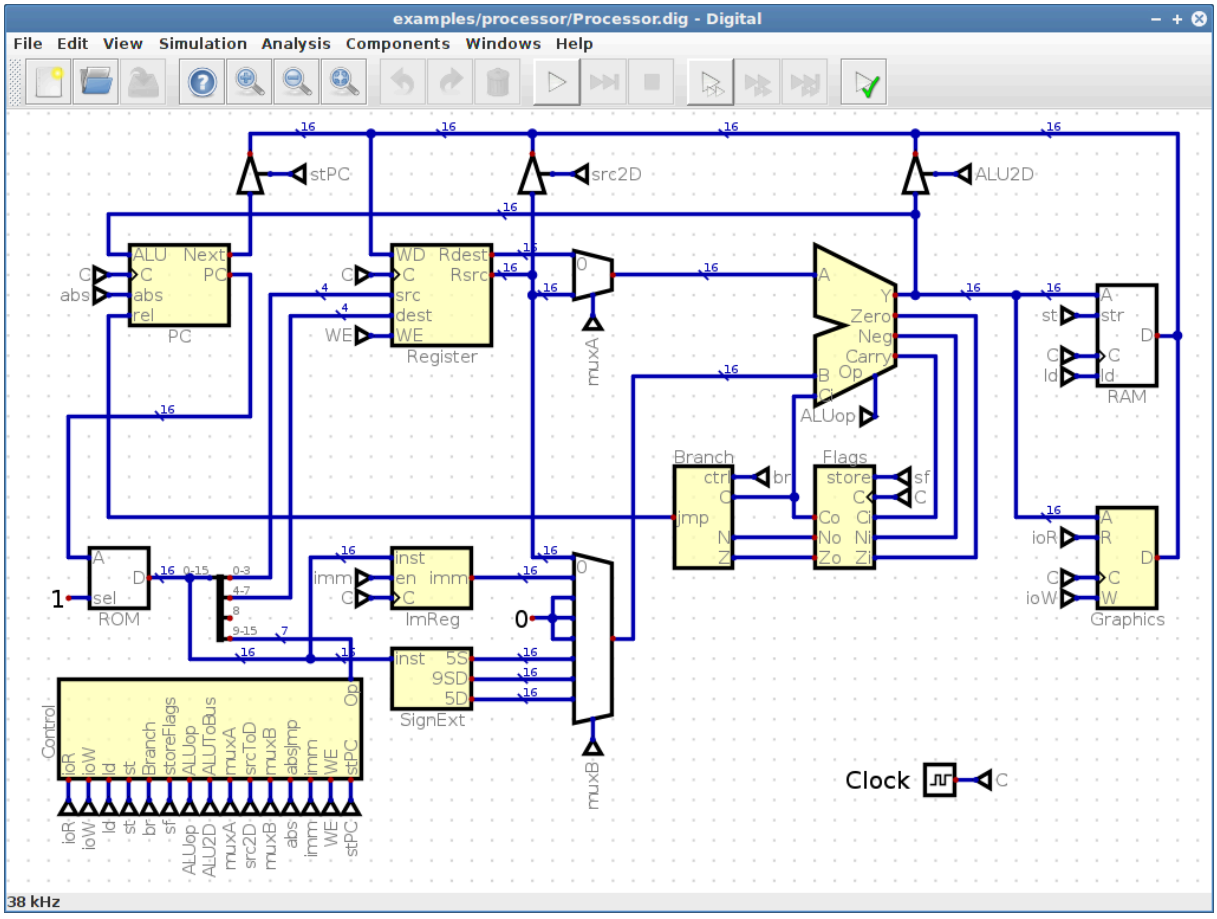
Digital



Revisão: v0.30
Data: 2023-02-03 10:55

Índice

E Biblioteca

A Geral

1. Digital

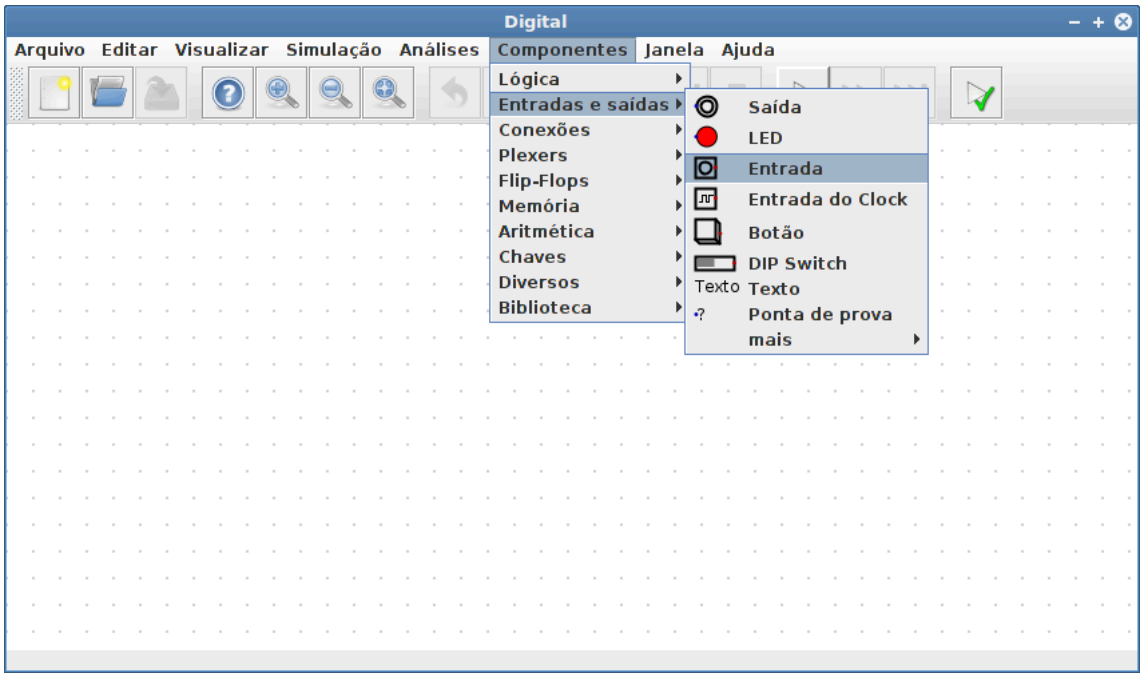
1.1. Introdução

Digital é um simulador simples usado para testar circuitos digitais. As portas lógicas são conectadas umas às outras por fios e o comportamento de todo o circuito pode ser simulado. O usuário pode interagir com a simulação ou pressionando botões, ou definindo valores para as entradas do circuito.

Dessa forma, a maioria dos circuitos básicos usados em eletrônica digital pode ser construída e simulada. Na pasta *examples*, os usuários encontrarão exemplos que incluem até um processador Harvard de 16-bits de ciclo único completamente funcional.

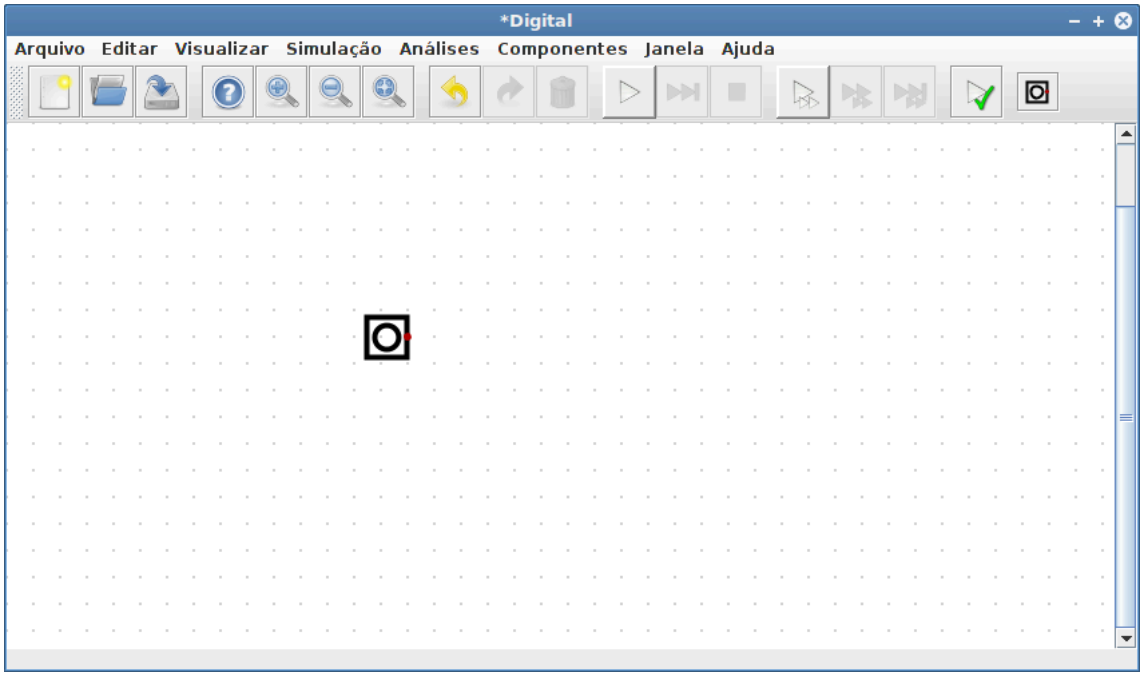
O simulador tem dois modos de operação: Edição e Simulação. No modo de edição, modificações no circuito poderão ser efetuadas. Os usuários poderão adicionar ou conectar componentes. Nesse modo, a simulação estará desabilitada. O modo de simulação será ativado ao se pressionar o botão *Iniciar* na barra de tarefas. Ao iniciar-se a simulação, o circuito será testado em relação à sua consistência. Se houver erros no circuito, uma mensagem apropriada será exibida e os componentes ou fios afetados ficarão em destaque. Se não houver erro, a simulação estará habilitada a prosseguir. Será possível, então, interagir com a simulação em curso. No modo de simulação não será possível modificar o circuito. Para fazer isso, para voltar ao modo de edição será necessário parar a simulação.

1.2. Primeiros passos

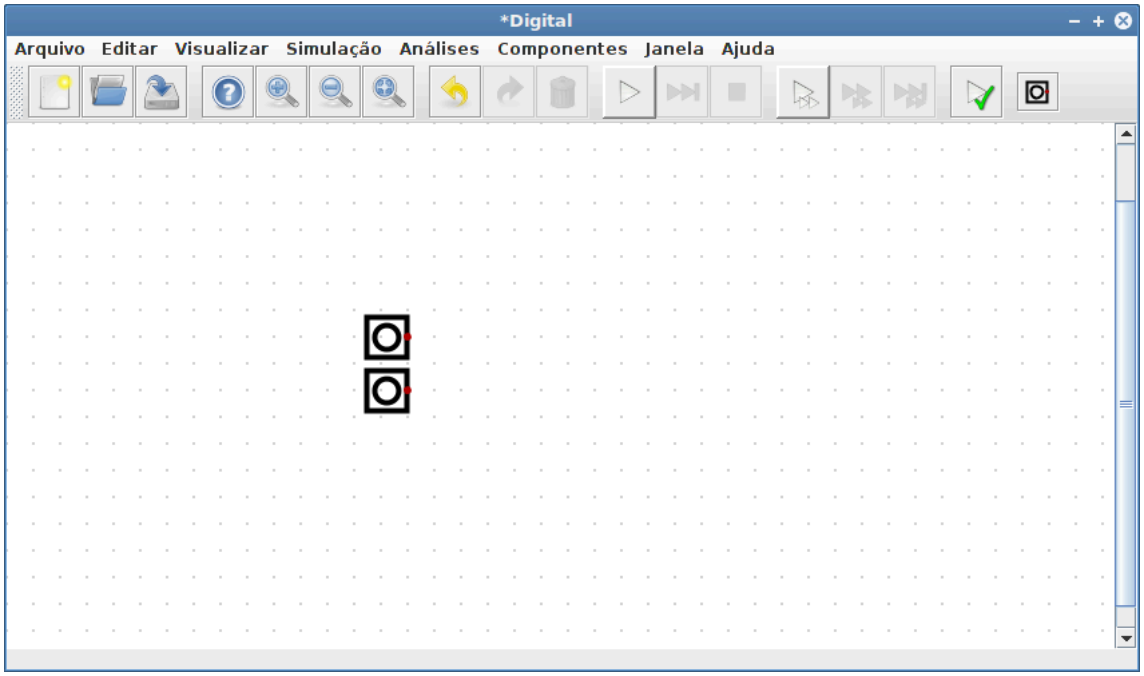


Como um primeiro exemplo, sugere-se construir um circuito com uma porta OU_Exclusivo (XOR). Na janela principal, o menu *Componentes* permitirá a seleção de vários tipos de componentes. Eles serão posicionados na área de trabalho. Esse processo poderá ser cancelado pressionando-se a tecla ESC a qualquer momento. Sugere-se iniciar pela seleção de um

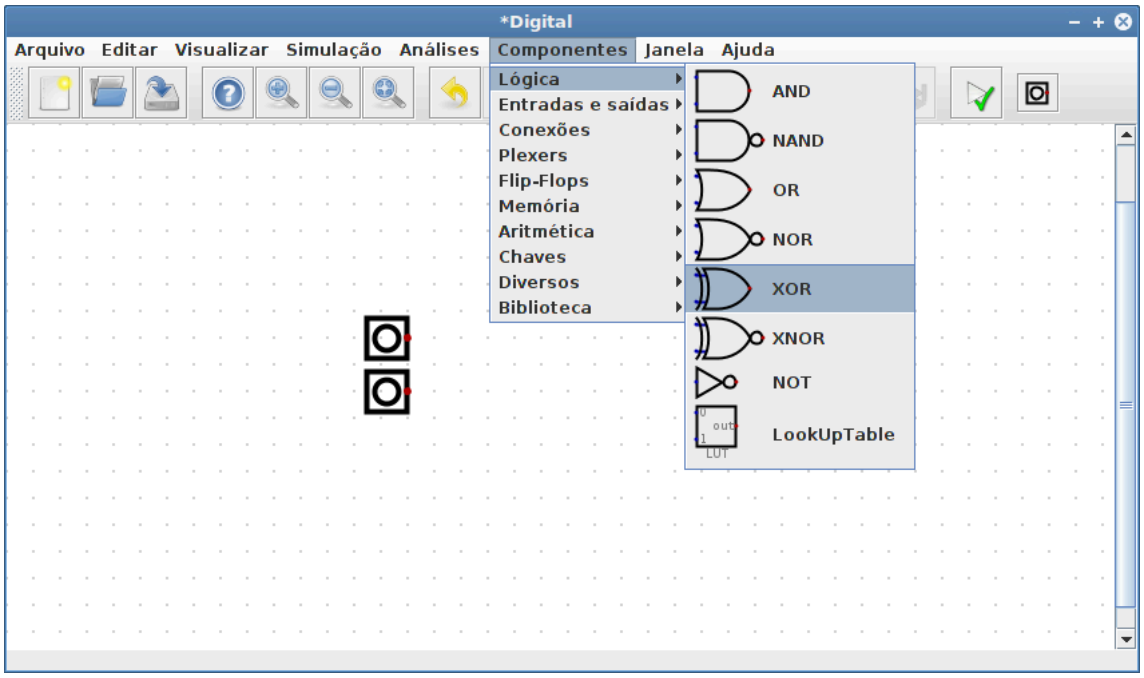
componente de entrada. Mais tarde esse componente poderá ser controlado interativamente mediante uso do mouse.



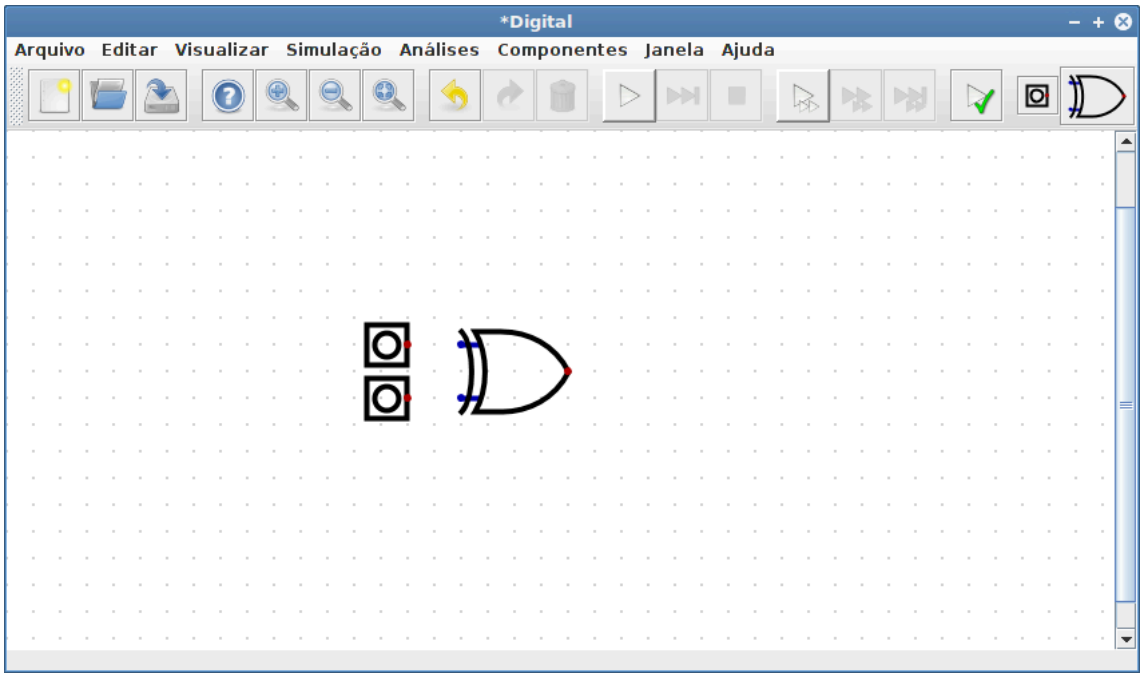
Após seleção, a primeira entrada poderá ser posicionada na área de trabalho. O ponto vermelho no símbolo do componente de entrada será seu ponto de conexão com um fio, ao qual se conectará mais tarde. A cor vermelha indicará uma saída. Isso significa que a porta definirá um sinal cujo valor será transmitido ao fio.



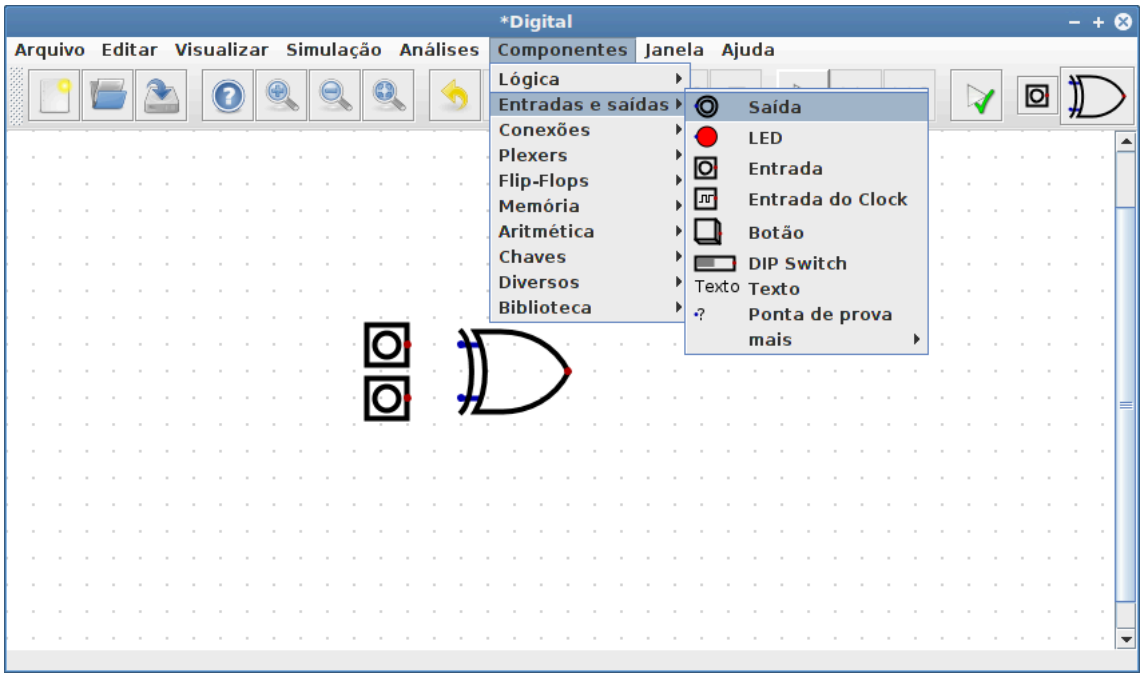
Da mesma forma, uma segunda entrada deverá ser adicionada. O melhor lugar para posicioná-la será diretamente abaixo da primeira.



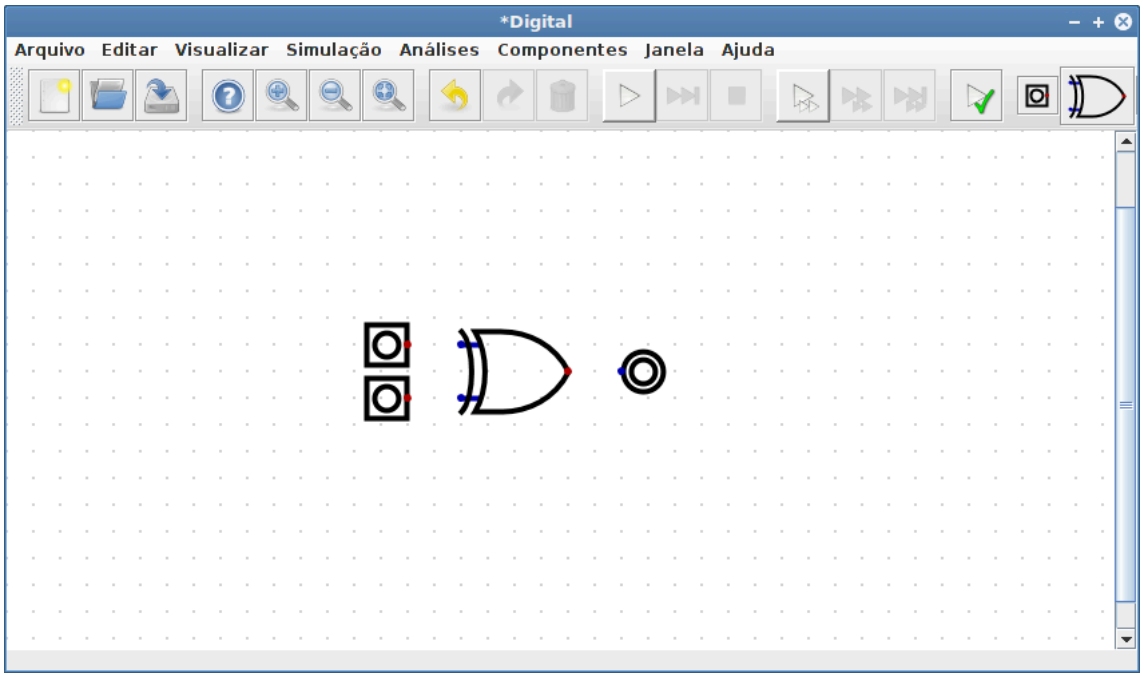
Após adicionar as entradas, a porta OU_Exclusivo (XOR) deverá ser selecionada. Essa porta representará o valor lógico da função.



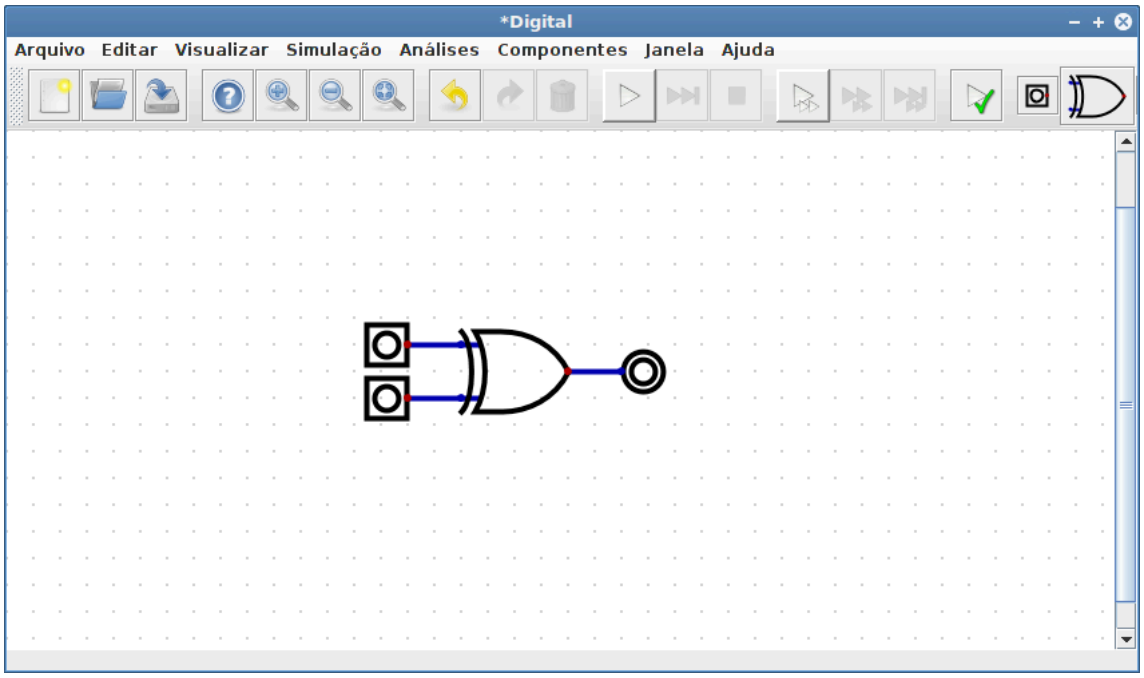
Essa porta poderá então ser adicionada ao circuito. É melhor posicioná-la de maneira que estabelecer as conexões subsequentes sejam tão simples de se fazer quanto o possível. Os pontos azuis indicarão os terminais de entrada da porta.



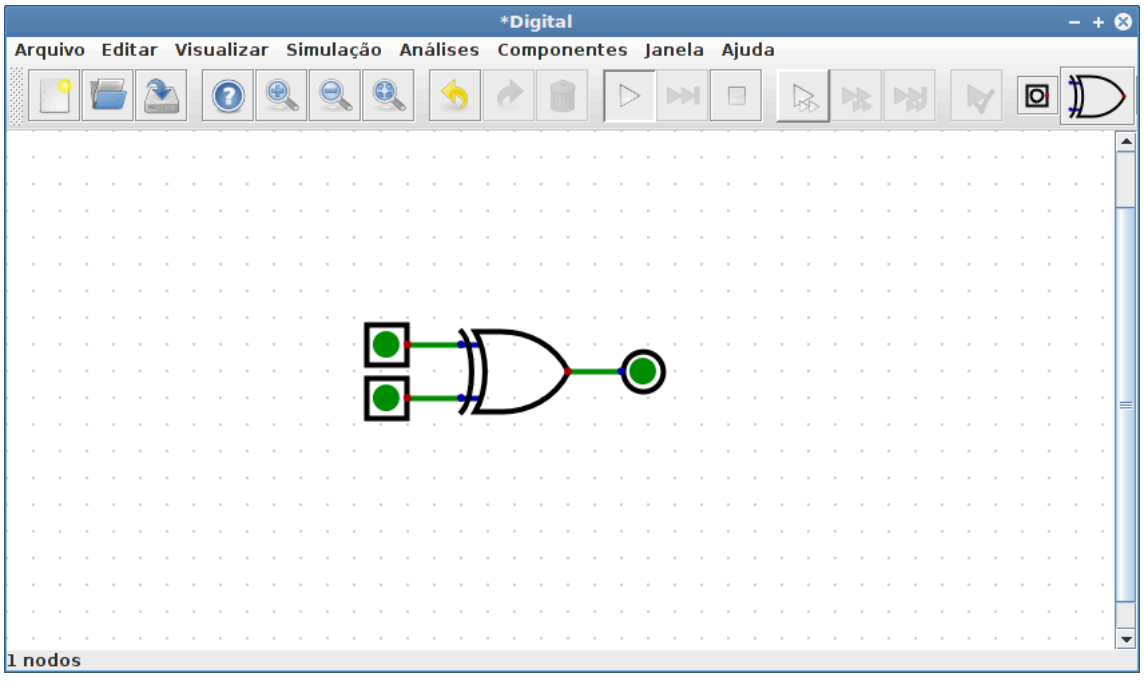
Em seguida, selecionar uma saída que será usada para mostrar o estado do sinal, ou que mais tarde transmitirá ao restante do circuito ao qual estiver incorporada.



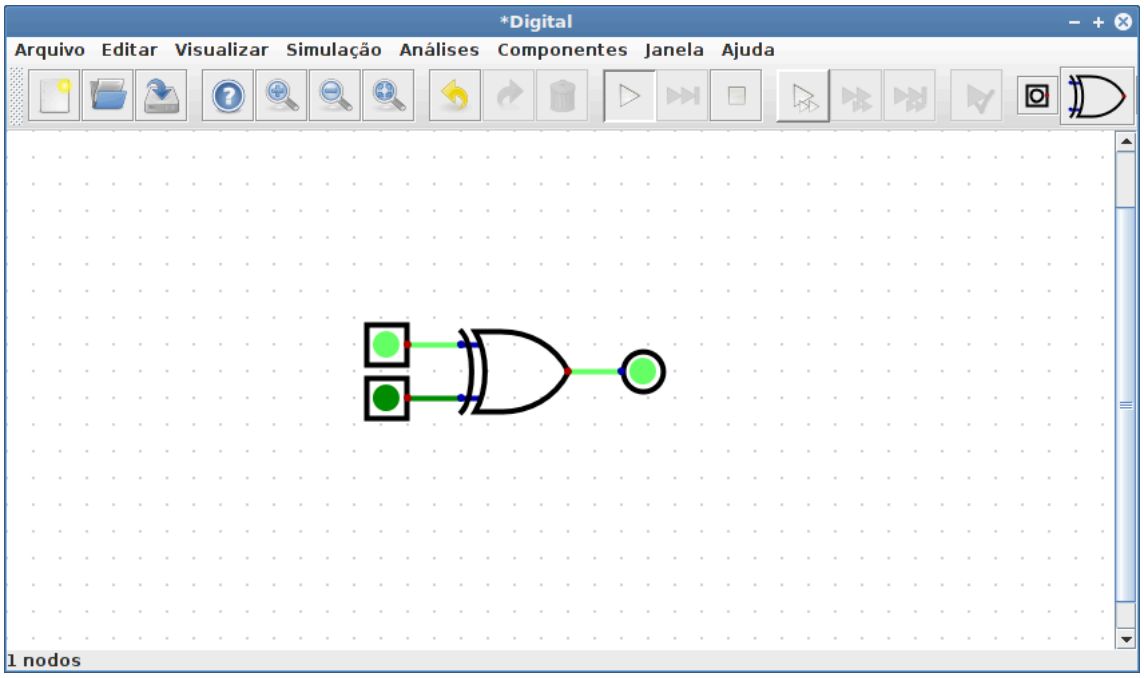
Ela também deverá ser posicionda de maneira que seja fácil executar conexões. A saída terá um ponto azul, que indicará um terminal de entrada. Por aí se poderá fornecer o valor que será exportado posteriormente.



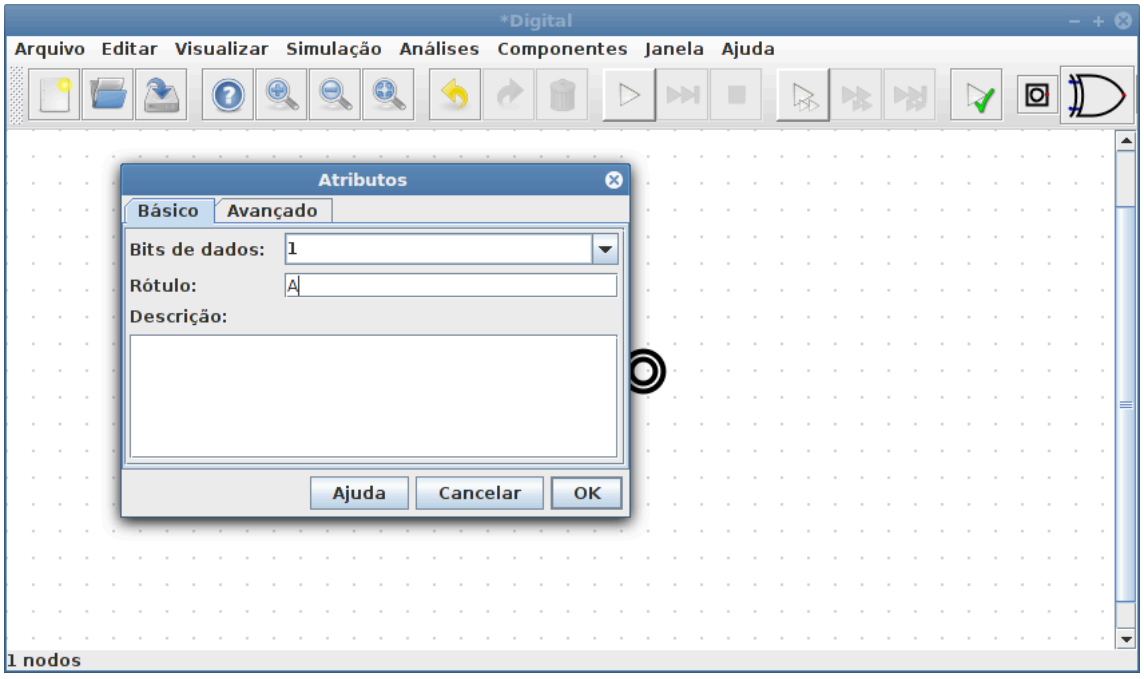
Depois que todos os componentes estiverem selecionados e devidamente posicionados, usar o mouse para criar as conexões entre os pontos azuis e vermelhos. Certificar-se que apenas um ponto vermelho esteja conectado a qualquer número de pontos azuis. Somente o uso de saídas tri-state poderão escapar dessa regra e interconectar outros pontos vermelhos. Quando todos os fios estiverem ligados, o circuito estará completo.



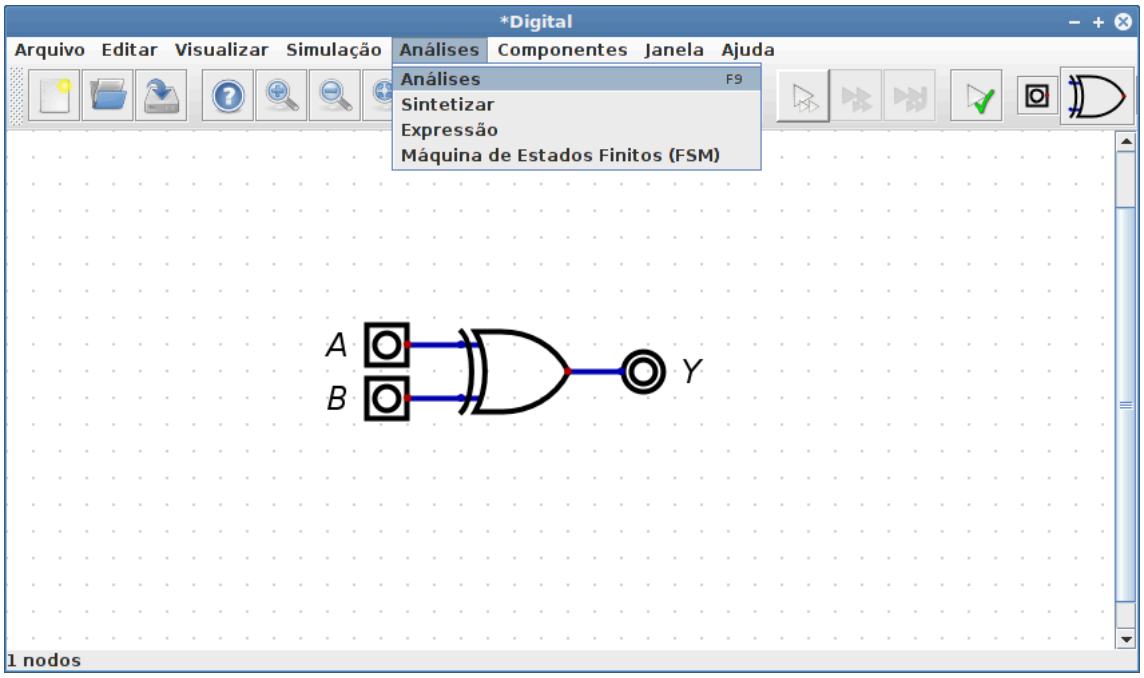
A interação com o circuito será possível quando a simulação for iniciada. Isso poderá ser feito clicando no botão *Iniciar* localizado na barra de ferramentas. Após a simulação iniciar-se, as cores dos fios irão se alterar e as entradas e as saídas tomarão seus respectivos valores. A cor verde clara indicará o nível lógico '1', e a verde escura, o nível lógico '0'. Na figura acima, todos os fios terão o valor '0'.



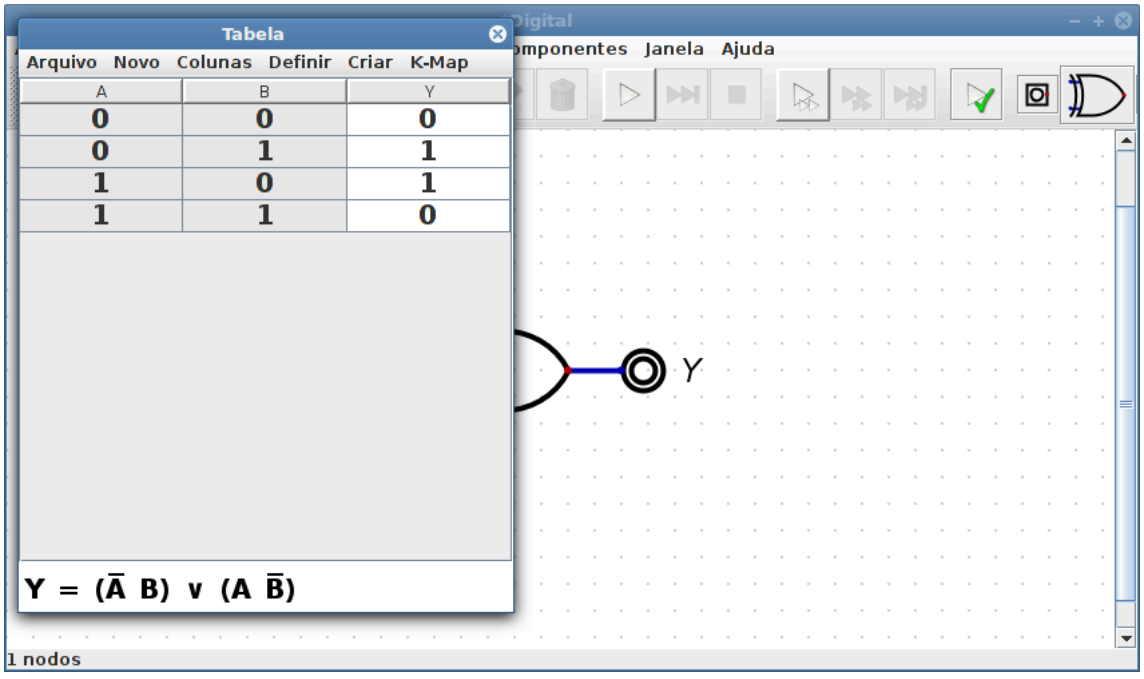
Por meio de clicks do mouse, as entradas poderão ser comutadas. Uma vez que a simulação estiver ativa, as saídas serão alteradas de acordo com os estados das entradas atuais. O circuito se comportará conforme a porta OU_Exclusivo esperada.



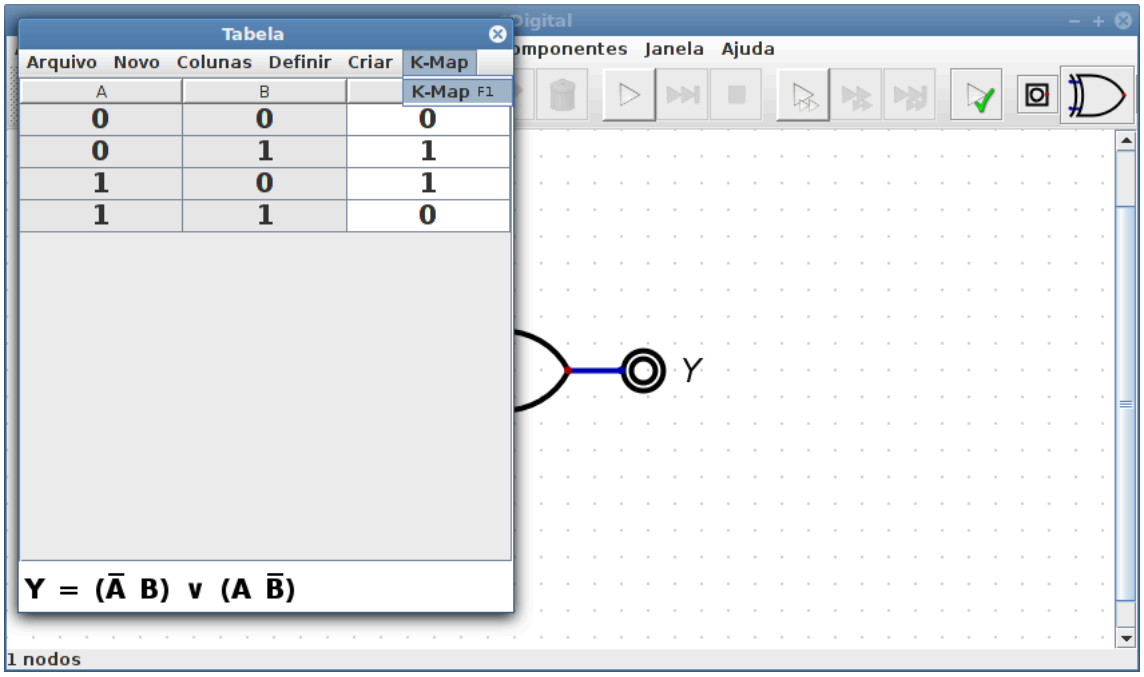
Para outras manipulações do circuito, primeiro a simulação deverá ser interrompida. A maneira mais fácil de fazer isso é por meio do botão *Parar* na barra de ferramentas. Ao clicar o botão direito sobre um componente, uma caixa de diálogo se abrirá para exibir suas propriedades. O rótulo 'A' poderá ser definido para a primeira entrada mediante esse atributo.



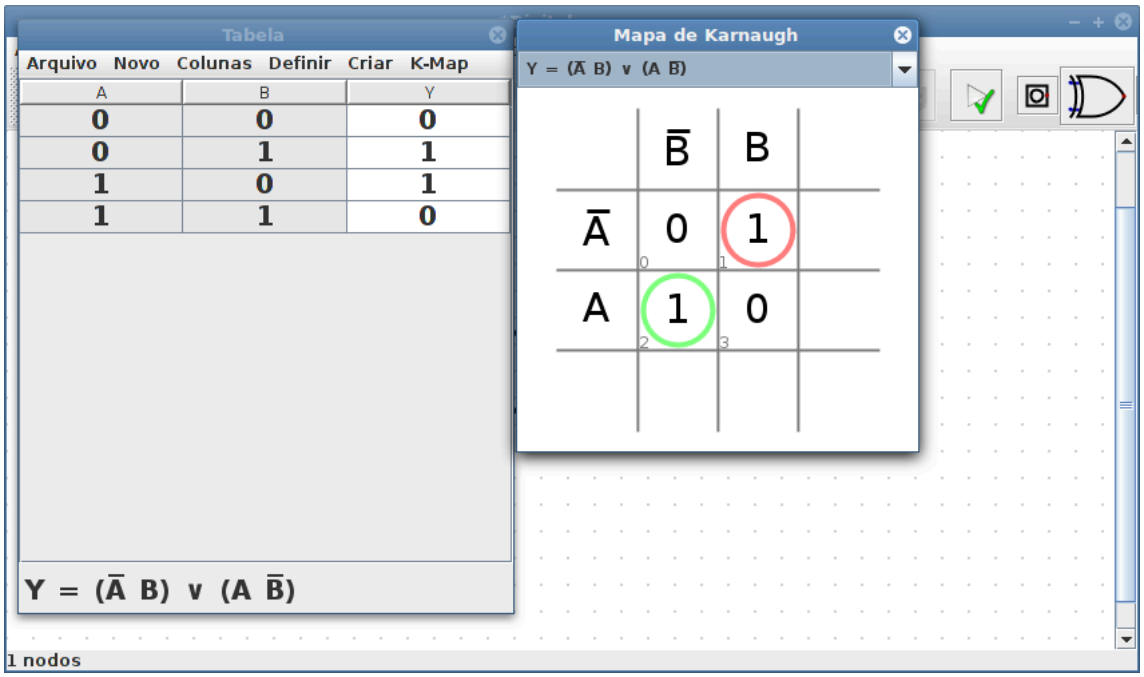
Dessa forma, os rótulos das outras entradas e saídas remanescentes poderão ser configurados. No menu *Análise* há um item correspondente. Essa função executará a análise do circuito atual. Contudo, isso somente será possível se todas as entradas e saídas estiverem devidamente rotuladas.



A tabela-verdade do circuito simulado será apresentada em uma nova janela. Abaixo da tabela poderá ser encontrada a expressão algébrica associada ao circuito. Se houver várias expressões algébricas equivalentes, uma janela em separado se abrirá, e mostrará todas as outras possíveis.



A caixa de diálogo da tabela tem uma entrada *K-Map* no menu principal. Isso permitirá exibir a tabela-verdade na forma de um K-Map.



Na parte de cima da caixa de diálogo há uma lista do tipo drop-down que permitirá selecionar a expressão desejada no K-Map. Dessa forma se poderá, por exemplo, ilustrar como várias expressões algébricas equivalentes poderão daí resultar. Entretanto, para esse exemplo, há apenas uma expressão mínima. A tabela-verdade também poderá ser modificada por meio de clicks no K-Map.

1.3. Fios

Todos os componentes deverão estar conectados por fios. Não será possível conectar dois componentes simplesmente colocando-os diretamente justapostos.

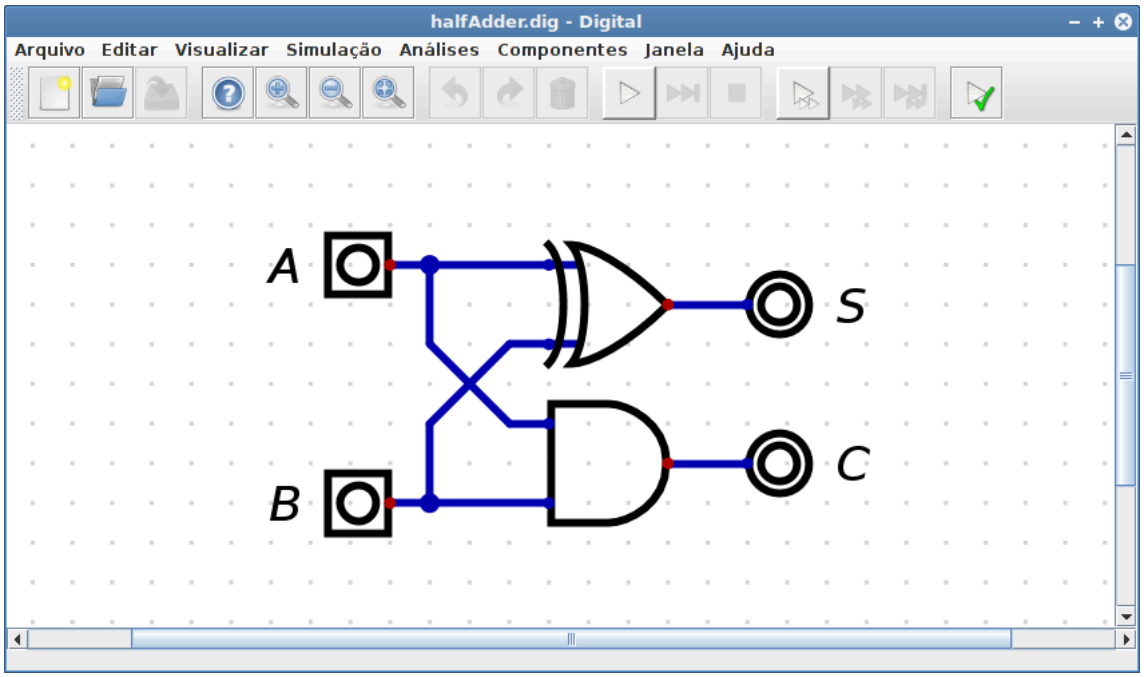
Além disso, só haverá conexões entre o ponto final de um fio e um componente. Se um pino de um componente for colocado no meio de um fio, nenhuma conexão se estabelecerá entre o componente e o fio. Portanto, um fio deverá realmente terminar em cada pino ao qual estiver conectado. Mesmo se um componente do tipo túnel for utilizado, deverá haver um fio entre o pino e esse túnel.

É necessário que o componente seja selecionado mediante a ferramenta de seleção retangular para que possa ser movido, incluindo os fios conectados. Para mover um componente sem os fios conectados, selecioná-lo apenas pelo click do mouse.

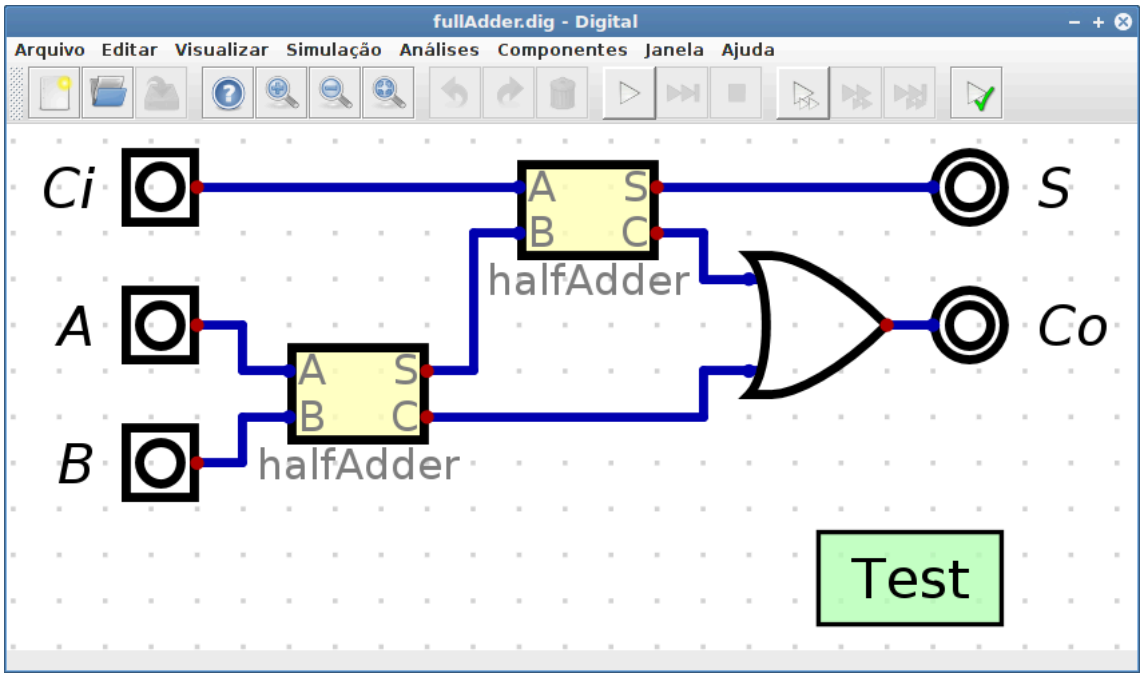
Com CTRL-click se poderá selecionar um único fio e movê-lo ou apagá-lo. Se a tecla 'D' for pressionada enquanto o fio for traçado, esse traçado poderá ser feito na diagonal. A tecla 'S' permitirá dividir um segmento de linha em dois.

1.4. Projeto hierárquico

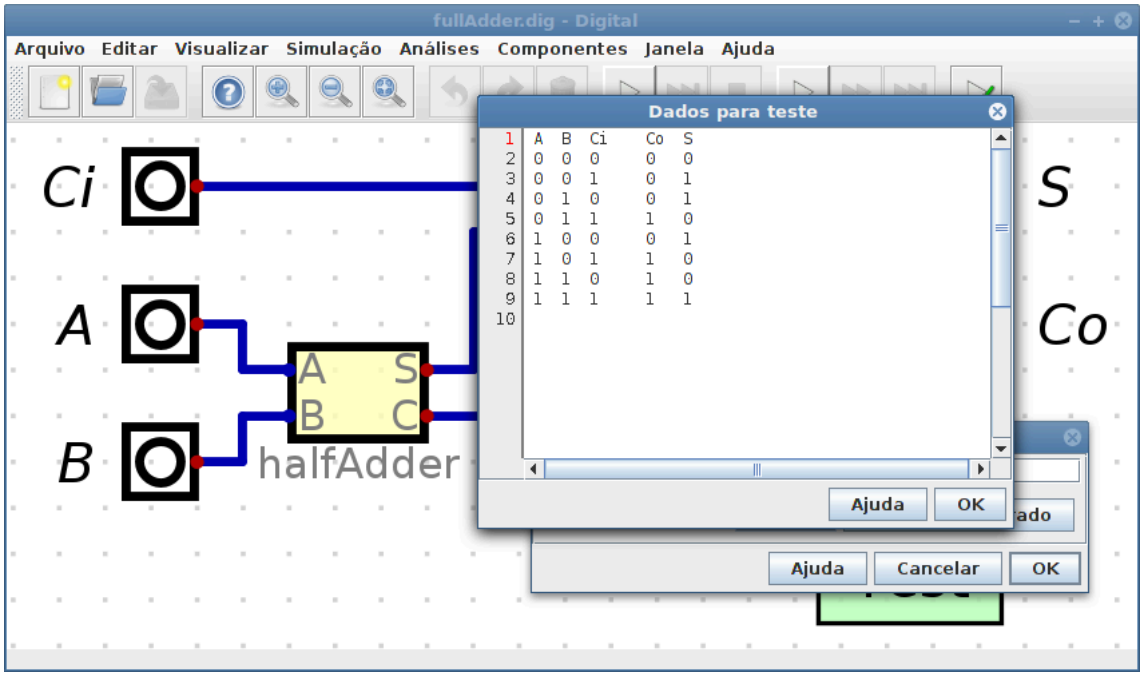
Ao se construir um circuito complexo, esse poderá se tornar rapidamente muito confuso. Para ajudar, diferentes partes de um circuito poderão ser armazenadas em arquivos distintos. Esse mecanismo também poderá ser usado em um subcircuito que, uma vez criado, venha ocorrer em vários outros circuitos. Essa abordagem é vantajosa porque os arquivos poderão ser armazenados independentemente empregando-se um sistema para o controle de versões, e as eventuais mudanças poderão ser melhor acompanhadas.



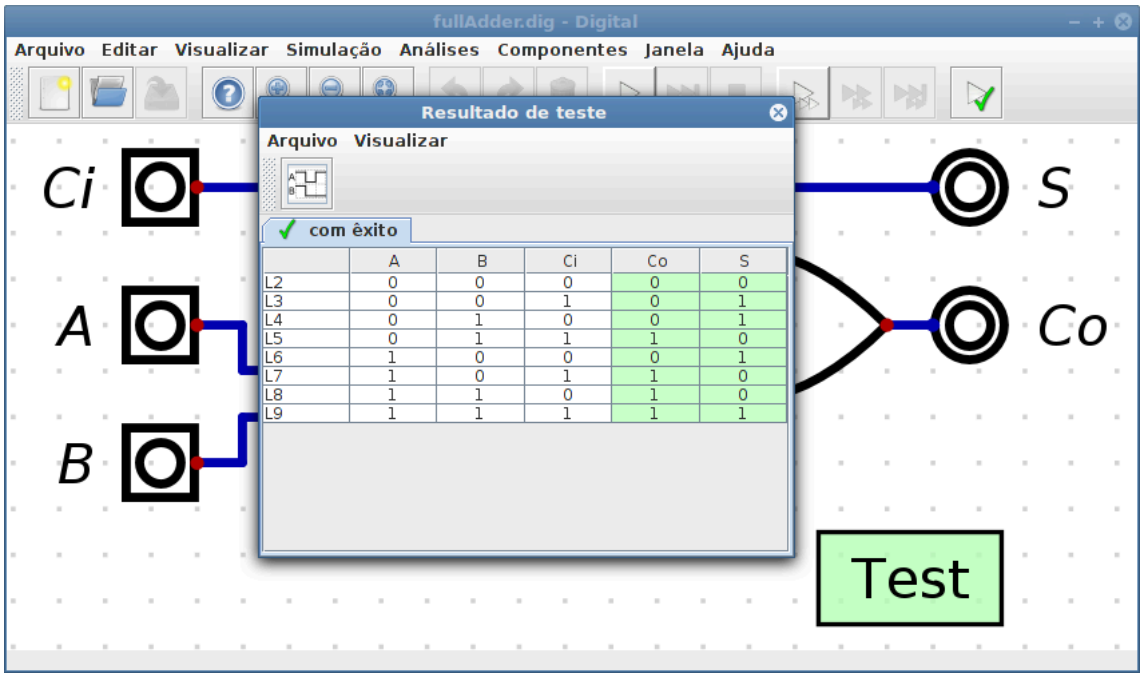
Por exemplo, considerar um somador de 4-bits: primeiramente, se construirá um somador simples para 2-bits (meia-soma). Esse será constituído por uma porta XOR e uma porta AND. A soma de dois bits 'A' e 'B' produzirá as saídas 'S' (soma) e 'C' (vai-um). Esse circuito será armazenado no arquivo *halfAdder.dig*.



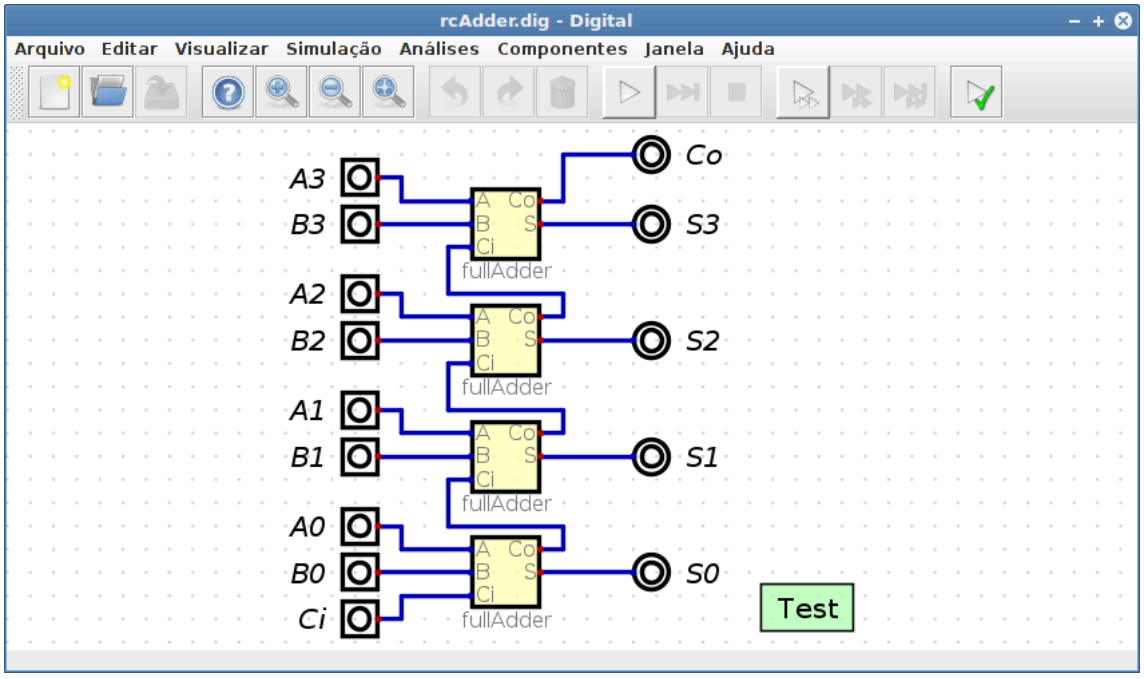
A partir da combinação de dois circuitos de meia-soma, um somador completo (full adder) poderá ser construído. Para isso, basta criar um arquivo vazio e salvá-lo como *fullAdder.dig* na mesma pasta onde estiver o *halfAdder.dig*. Em seguida, o somador simples poderá ser adicionado ao novo circuito via menu *Componentes→Personalizado*. A ordem dos pinos no pacote do somador simples poderá ser rearranjada pelo menu *Editar→Ordenar entradas* ou *Editar→Ordenar entradas*. O somador completo operará os três bits 'A', 'B' e 'Ci' e produzirá as saídas 'S' (soma) e 'Co' (vai-um).



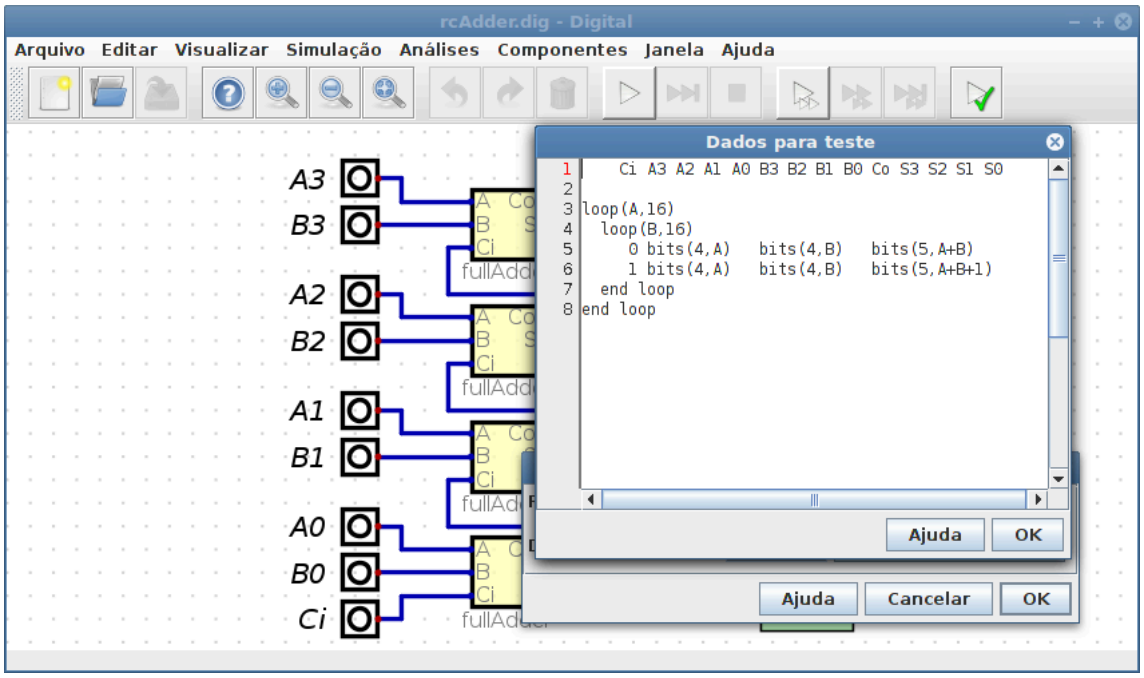
Para se verificar a correção da função do somador completo, um caso de testes poderá ser acrescentado. No caso de testes, a tabela-verdade será armazenada, a qual será submetida ao circuito. Dessa forma se poderá automaticamente verificar seu comportamento.



Os testes poderão ser executados via editor de caso de testes ou pelo botão de teste na barra de ferramentas. As células da tabela receberão destaque na cor verde para indicar quando a saída do circuito for equivalente à tabela-verdade nesse caso de testes.



Assim, somadores completos poderão ser combinados para formar um somador em cascata (ripple-carry). Nesse caso, o vai-um de uma adição serão repassado à próxima como entrada para operar o bit de ordem mais alta, da mesma maneira como se faz manualmente com papel-e-lápis. O somador de 4-bits deverá ser testado para confirmar sua função. Para isso, um caso de testes deverá ser acrescentado.



Um caso de testes executará 100% dos testes, o que somente será possível para circuitos relativamente simples: todas as possíveis combinações de 512 entradas serão aplicadas ao circuito, e se verificará se sua saída estará correta. A primeira linha relacionará a lista de sinais de entradas e saídas. Abaixo dessa, os valores das entradas a serem aplicados e os das saídas a serem verificadas na mesma linha, tal como em uma tabela-verdade. Nesse exemplo, contudo, 512 linhas serão requeridas. Isso poderá ser uma tarefa tediosa e não isenta da possibilidade de erros. Será mais fácil e mais confiável gerar automaticamente essas linhas necessárias. Para isso, as variáveis *A* e *B* deverão, cada uma, variar de 0 a 15. Os respectivos valores de *A* e *B* serão atribuídos às entradas 'A[n]' e 'B[n]'. Isso será verificado com as saídas do circuito para os valores de *A+B*. Será verificado novamente para os bits de vai-um definidos, para cada caso em que resultar em *A+B+1*. Os detalhes da sintaxe de teste serão fornecidos pela caixa de diálogo de ajuda.

Se um circuito estiver incorporado a outro, somente o nome do subcircuito será armazenado nesse, e não o próprio subcircuito. Os arquivos correspondentes aos subcircuitos incorporados deverão portanto serem encontrados na pasta de arquivos do sistema durante a simulação. De modo a prover suporte para os diversos métodos de trabalho dos usuários da melhor forma possível, e ainda evitar uma administração complexa dos caminhos para a importação e similares, uma estratégia pouco usual para fazê-lo foi implementada.

Somente os nomes dos circuitos incorporados serão armazenados no arquivo do circuito, não seus caminhos completos. Se um arquivo necessitar ser aberto, todas as subpastas serão vasculhadas para se encontrar o arquivo com nome correspondente. Se um arquivo aceitável for encontrado, será importado. Esse processo dependerá apenas do nome do arquivo a ser lido, e não do caminho até ele. De forma correspondente, uma mensagem de erro será gerada se houver diversos arquivos com nomes iguais em subpastas diferentes, uma vez que haverá ambiguidades.

Uma estrutura de projeto adequada portanto será a seguinte: o circuito principal (raiz) ficará localizado em uma pasta separada. Todos os circuitos importados deverão estar na mesma pasta ou em subpastas. Todos os circuitos deverão ter nomes diferentes, dessa forma não deverá acontecer que ocorram circuitos com o mesmo nome em pastas diferentes.

2. Simulação

2.1. Propagação de atraso

Durante a simulação, cada porta lógica terá um atraso de propagação. Cada componente encontrado na biblioteca terá o mesmo atraso de propagação independente de sua complexidade. Assim a porta AND terá o mesmo atraso de propagação de um multiplicador. As únicas exceções serão os diodos, as chaves e os distribuidores que serão usados para os barramentos de dados. Esses componentes não terão qualquer atraso.

Se for necessário simular uma porta, por exemplo, um multiplicador, com um atraso de propagação mais longo, um componente próprio para atraso deverá ser inserido no circuito logo antes da saída do multiplicador.

Se um circuito estiver incluído em outro maior, esse incluirá também o seu comportamento ao longo do tempo. Assim, se um circuito complexo for incluído e tiver um atraso de propagação bem grande, porque os sinais das entradas terão que passar por três portas, até que cheguem à saída, esse comportamento será conservado quando for incluído em outro circuito. Não haverá atrasos adicionais pela inclusão em outro circuito. Se nem todas as saídas de um circuito tiverem o mesmo atraso de propagação, isso também poderá ser o caso quando for incluído em um circuito maior. Em geral, ao se incluir um circuito em outro, isso não modificará o comportamento ao longo do tempo de ambos. Um circuito incorporado a outro se comportará exatamente da mesma forma como se todos os componentes também tivessem sido inseridos nesse mesmo nível do circuito.

3. Análise e síntese de circuito

Um circuito poderá ser analisado via menu*Análise*. A tabela-verdade será gerada apenas para circuitos combinacionais. A tabela-verdade poderá ser editada se assim o desejar. Um novo circuito será gerado a partir dessa tabela-verdade após sua edição.

Além de circuitos combinatórios, também será possível analisar ou gerar circuitos sequenciais. Em lugar de uma simples tabela-verdade, se criará uma tabela de transições de estado. Cada flip-flop ocorrerá tanto no lado das entradas e quanto no das saídas da tabela de transições de estado. Nessa tabela, do lado direito, poderão ser encontrados os próximos estados, que tomarão lugar após o próximo sinal do clock. O próximo estado dependerá do estado atual dos flip-flops encontrados do lado esquerdo da tabela. Para que a análise seja possível, todos os flip-flops deverão ter nomes.

A seguinte convenção se aplicará: os próximos estados de um bit do lado direito da tabela serão indicados por letras minúsculas 'n+1'. O estado atual correspondente será indicado pelo sufixo 'n'. Se houver uma variável de estado 'A', 'An' indicará o estado atual e 'An+1' indicará o próximo estado. Se, na tabela-verdade dos lados esquerdo e direito, os sinais estiverem presentes, e corresponderem a esse padrão, irá se supor uma transição de estados nessa tabela, e um circuito sequencial será gerado em lugar de um combinacional.

Deve-se notar que se um circuito ao ser analisado contiver elementos combinacionais, além dos flip-flops D e JK nativos, como por exemplo, portas NOR, isso não será reconhecido com um flip-flop e, portanto, não será possível analisar tal circuito.

4. Hardware

4.1. GAL16v8 e GAL22v10

No menu de geração de circuito na tabela-verdade também há funções para se gerar arquivos JEDEC. Esse formato especial de arquivo descreve o mapa de fusíveis de um PLD. O arquivo JEDEC poderá ser gravado em um PLD correspondente por meio de um programador especial. No momento, os circuitos do tipo *GAL16v8* e *GAL22v10* ou mapa de fusíveis para dispositivos compatíveis têm suporte.

4.2. ATF150xAS

Os chips na família *ATF150x* são CPLDs comuns com até 128 macrocélulas. São disponíveis em um encapsulamento PLCC, o que os torna convenientes para exercícios em laboratório: se um circuito integrado se queimar durante os exercícios, esse poderá ser simplesmente substituído. Além disso, como o *ATDH1150USB* é bem fácil de usar. Basta ter um programador de baixo custo disponível. Esse programador será capaz de lidar com os chips *ATF150x* em um sistema que use a interface JTAG. Uma placa acessível para avaliação (*ATF15XX-DK3-U*) também está disponível. O software *ATMISP* , à disposição no site da ATMEL/Microchip, será necessário para programar esses chips.

Infelizmente, detalhes do mapa de fusíveis não estão publicamente à disposição, de modo que não há filtros adequados para isso que possam ser incorporados à ferramenta, assim como foi possível para os chips *GAL16v8* e *GAL22v10*.

Portanto, os filtros *fit150[x].exe* fornecidos pela ATMEL deverão ser usados. Esses programas criarão um arquivo *JEDEC* adequado a partir de um arquivo *TT2* que poderá então ser usado para programar o chip. A ferramenta iniciará o filtro automaticamente cada vez que o arquivo *TT2* for criado. Para esse propósito, o caminho para os filtros *fit150[n].exe* deverão estar especificados nas configurações. O arquivo *JEDEC* criado poderá ser aberto e programado diretamente com *ATMISP* .

Por razões legais o filtro *fit1502.exe* não pode ser distribuído em conjunto com a ferramenta. Contudo, poderá ser encontrado na pasta *WinCupl\Fitters* após instalação do *WinCupl* . *WinCupl* está à disposição no site da ATMEL/Microchip. Em sistemas Linux, os filtros também poderão ser executados por essa ferramenta, se estiver instalado o *wine*.

4.3. Exportar para VHDL ou Verilog

Um circuito poderá ser exportado para VHDL ou Verilog. Um arquivo será gerado contendo a descrição completa do circuito. Códigos VHDL gerados foram testados com o Xilinx Vivado e o simulador de código aberto para VHDL - ghdl. O código Verilog foi testado com o simulador Icarus Verilog.

Se um circuito contiver casos de testes, seus dados serão usados para gerar o teste comportamental em HDL. Isso poderá ser usado para verificar a correção da função do circuito em uma simulação em HDL.

Arquivos adicionais necessários para placas especiais poderão ser criados. No momento, somente as placas BASYS3, Mimas e Mimas V2 têm suporte. Um arquivo de restrições será criado contendo as atribuições aos pinos. As descrições dos pinos poderão ser encontradas nos data sheet das placas, e deverão ser fornecidas, assim como o número do pino para suas entradas e saídas.

Na placa BASYS3, se a frequência de clock de um circuito for baixa, um divisor de frequências será incorporado ao código HDL a fim de dividir o clock da placa para o valor conveniente. Se a frequência de clock selecionada exceder 4.7MHz, a unidade MMCM do Artix-7 será usada para a geração de clock. Isso garantirá que os recursos disponíveis na FPGA para distribuição do clock serão usados. Isso permitirá que o exemplo de processador possa ser executado a 20MHz, e se quiser, sem o multiplicador, 30HMz também será possível.

Se um circuito for executado em uma placa BASYS3, um novo projeto poderá ser criado no Vivado. O arquivo VHDL gerado e o arquivo de restrições deverão ser acrescentados ao projeto. Uma vez criado o projeto, a sequência de bits (bitstream) poderá ser gerada e o Gerenciador de Hardware poderá ser usado para programar a placa BASYS3.

Para criar o arquivo de restrições necessário, além do arquivo HDL, a placa correspondente deverá ser configurada nas configurações. No campo "Toolchain Configuration" o arquivo XML correspondente poderá ser selecionado. As configurações disponíveis poderão ser encontradas na pasta *examples/hdl* e ter o arquivo extensão *.config*. Se a configuração foi integrada com sucesso, um outro menu será exibido, o que tornará acessíveis as funções específicas da placa.

5. Formatos personalizados

Embora a ferramenta tenha algumas opções para determinar a aparência de um circuito quando esse for incorporado a outro, em alguns casos poderá ser mais útil indicá-lo por meio de uma formato bem especial para o subcircuito. Uma demonstração disso poderá ser a forma de se representar uma ALU em um processador incluída em vários exemplos. Esse capítulo explicará como definir um formato especial para um circuito.

A ferramenta não dispõe de um editor para criar um formato especial. Para isso, um pequeno procedimento será necessário para dar outros formatos ao circuito: primeiro, o circuito será aberto, o que será representado por determinado formato. Depois, um gabarito SVG será criado para esse circuito. Nesse gabarito, o circuito será representado por um simples retângulo. Ele também conterá todos os pinos do circuito, representados em azul (entradas) e vermelho (saídas). Para se verificar qual círculo corresponderá a qual pino, conferir o ID do círculo nas propriedades do objeto. Esse ID estará na forma *pino:[nome]* ou *pino+:[nome]*. Na última forma variante, o pino terá um rótulo, caso tenha sido reimportado. Se tal rótulo não for desejado, o sinal + poderá ser removido.

Esse arquivo SVG poderá então ser editado. O mais adequado a se fazer é abrir o programa fonte Inkscape disponível gratuitamente. Os pinos poderão ser movidos livremente, mas aderidos ao ponto mais próximo na grade durante a reimportação.

Se arquivos SVG tiverem que ser usados, será mais fácil abrir e criar um gabarito e colar o gráfico existente nesse via Copiar&Colar.

Se o arquivo for salvo, ele poderá ser importado para a ferramenta. O arquivo será lido e todas as informações necessárias serão extraídas e armazenadas no circuito. Para usos futuros do circuito, o arquivo SVG não será mais necessário.

Uma observação final: SVG é um formato de arquivo muito útil e flexível. Poderá ser usado para descrever gráficos extremamente complexos. A ferramenta poderá não estar apta para importar todos os possíveis arquivos SVG sem erros. Se um arquivo não puder ser importado, ou não tiver o aspecto esperado, alguns experimentos poderão ser necessários antes que o resultado desejado possa ser alcançado.

6. Perquntas frequentes

Como mover um fio?

Selecionar uma das extremidades com a seleção retangular. Assim se poderá mover o fio usando o mouse. Também se poderá selecionar um fio usando CTRL + botão do mouse.

Como apagar um fio?

Selecionar uma das extremidades e apertar a tecla *DEL* ou clicar na lixeira. Também se poderá selecionar um fio usando CTRL + botão do mouse.

Como mover um componente incluido todos os fios conectados?

Selecionar o componente com a seleção retangular. A seleção deverá incluir todo o componente. Depois se poderá mover o componente incluindo seus fios usando o mouse.

Há um componente não conectado a um fio, mesmo havendo pinos sobre o fio.

Um pino somente estará conectado a um fio se o fio tiver seu ponto final nesse pino.

Se os nomes dos pinos em um circuito forem longos, esse não serão legíveis quando o circuito for incorporado, o que se poderá fazer?

A largura do bloco poderá ser aumentada usando o item de menu *Editar→Editar atribuitos de circuito*.

Os pinos de um circuito incorporado não estão uma ordem adequada, isso pode ser modificado?

A sequência poderá ser alterada usando-se a entrada de menu *Editar→Ordenar entradas* ou *Editar→Ordenar saídas*.

Quando a simulação se iniciar, e um fio ficar cinza, o que isso significa?

As cores verde claro e verde esculo são usadas para representar os estados em nível alto e baixo. A cor cinza significa que o fio está em estado de alta impedância (Z).

Fornecida a tabela-verdade, como calcular as equações booleanas minimizadas?

No menu *Análise* selecionar o item *Sintetizar*. Depois, fornecer a tabela-verdade. Na parte de baixo da janela se poderá encontrar a equação booleana correspondente. Se for indicada mais do que uma variável dependente, uma nova janela se abrirá para apresentar todas as equações booleanas geradas.

Fornecida a tabela-verdade, mas se houver mais do que uma equação booleana apresentada, qual delas é a correta?

A minimização de uma equação booleana poderá resultar em várias equivalentes que descreverão a mesma função. Todas as mostradas criarão a mesma tabela-verdade. Poderá haver diferenças dependentes de "don't cares" na tabela-verdade.

Se tiver a tabela-verdade, como criar um circuito que a represente?

No menu *Análise* selecionar o item *Sintetizar*. Depois, fornecer a tabela-verdade. A tabela poderá ser editada mediante uso dos menus *Nova* ou *Adicionar colunas*. No menu *Criar* se poderá construir o circuito pelo uso do item *Circuito*.

Como editar o nome de um sinal na tabela-verdade?

Clicar com o botão direito sobre o cabeçalho da tabela para editar o nome.

Se tiver a equação booleana, como criar um circuito?

No menu *Análise* selecionar o item *Expressão*. Depois, fornecer a expressão.

Como criar uma tabela-verdade para uma equação booleana?

No menu *Análise* selecionar o item *Expressão*. Depois, fornecer a expressão. Em seguida, criar um circuito e de novo no menu *Análise* no item *Análise* criar a tabela-verdade.

Como criar um arquivo JEDEC para um dado circuito?

No menu *Análise* selecionar o item *Análise*. Depois, no menu *Criar* na nova janela escolher o dispositivo correto no submenu *Dispositivo*.

Ao criar um arquivo JEDEC, como atribuir um número de pino a certo sinal?

Para as entradas e saídas correspondentes se poderá atribuir um número de pino na caixa de diáologo de configuração do mesmo.

Tenho um arquivo JEDEC criado. Como programá-lo para um *GAL16v8* ou *GAL22v10*?

Para programar tal chip um hardware programador será necessário.

7. Atalhos de teclado

Space	Iniciar ou parar a simulação.
F6	Abrir a caixa de diálogo para a tabela de medidas.
F7	Executar até haver interrupção
F8	Executar cados de testes
C	Uma mudança de clock (funcionará se, e somente se, houver um componente de clock simples).
F9	Análise do circuito
CTRL-A	Selecionar tudo.
CTRL-X	Recortar os elementos selecionados para a área de transferência.
CTRL-C	Copiar os elementos selecionados para a área de transferência.
CTRL-V	Inserir componentes a partir da área de transferência.
CTRL-D	Duplicar a seleção atual sem modificar a área de transferência.
R	Quando inserir isso, rotacionar os componentes.
L	Inserir o último componente de novo.
CTRL-N	Novo circuito.
CTRL-O	Abrir circuito.
CTRL-S	Salvar circuito.
CTRL-Z	Desfazer a última modificação.
CTRL-Y	Reaplicar a última modificação desfeita.
P	Programar um diodo ou um FG-FET.
D	Quando traçar um fio alterar para o modo em diagonal.
F	Quando traçar uma linha mudar a sua orientação.
S	Dividir um fio em dois.
ESC	Abortar a ação corrente.
Del	Remover os componentes selecionados.
Backspace	Remover os componentes selecionados.
+	Aumentar o número de entradas no componente apontado pelo mouse.
-	Diminuir o número de entradas no componente apontado pelo mouse.
CTRL +	Zoom in
CTRL -	Zoom out
F1	Ajustar ao tamanho
F5	Mostrar ou omitir a exibição hierárquica dos componentes

B Configurações

A seguir descrevem-se as configurações disponíveis no simulador.

Configurações

As configurações globais do simulador especificação, dentre outras coisas, do idioma, a simbologia a ser usada ou os caminhos para as ferramentas externas.

Atributos

Usar formatos IEEE 91-1984
: Usar formatos IEEE 91-1984 em lugar dos retangulares

Idioma
: Idioma na interface gráfica (GUI). Será efetivo apenas após reinicialização.

Formato
: Formato da tela para expressões.

Esquema de cores
: Esquema de cores

Cores definidas pelo usuário
: Cores definidas pelo usuário

A hierarquia de componentes visível desde a inicialização.
: Se definido, fará com que a hierarquia de componentes seja visível desde a inicialização.

Mostrar grade.
: Mostrar uma grade na tela principal.

Mostrar o número de fios em um barramento.
: ATENÇÃO: O valor será atualizado somente quando a simulação for iniciada.

Dicas sobre conexões
: Se habilitada, linhas serão destacadas quando o mouse passar sobre elas.

Usar a tecla '='
: Usar a tecla '=' ao invés da tela '+'. Isso será sempre útil se o caractere '+' não for uma tecla primária, em seu lugar o caractere '=' será usado, como acontece em teclados com os formatos em inglês ou francês.

Mostrar caixa de diálogo para se renomear túneis.
: Se selecionado, uma caixa de diálogo será exibida para se renomear automaticamente todos os túneis de mesmo nome logo após um túnel tiver sido renomeado.

Biblioteca
: Pasta contendo a biblioteca com sub-circuitos predefinidos. Contém, por exemplo, os componentes da sére 74xx. Também será possível adicionar circuitos próprios que estejam guardados nesse mesmo lugar. Deve-se garantir que os nomes de todos os arquivos nessa pasta e subpastas sejam únicos.

Biblioteca Java
: Arquivo jar contendo componentes adicionais implementados em Java.

Filtro ATF15xx
: Caminho para o filtro ATF15xx. Fornecer a pasta que contenha os arquivos fit15xx.exe fornecidos pela Microchip (antes ATMEL).

ATMISP
: Caminho para o arquivo executável ATMISP.exe. Se definido, o software ATMISP poderá ser ativado automaticamente!

GHDL
: Caminho parao arquivo executável ghdl. Somente será necessário se for desejado usar ghdl para simular componentes definidos em vhdl.

IVerilog
: Caminho para a pasta de instalação do Icarus Verilog. Somente será necessário se for desejado usar iverilog para simular componentes definidos em verilog.

Configuração de ferramentas
: Usado para configurar uma integração a um conjunto de ferramentas. Permitirá o lançamento de ferramentas externas, por exemplo, para se programar FPGA ou similar.

Tamanho da fonte em menus [%]
: O tamanho das fontes usadas no menu é dada em uma porcentagem do tamanho padrão.

Usar clicks do mouse MacOS.
: Usar CTRL-click em lugar de botão direito.

Permitir conexão remota
: Se definido, uma porta TCP/IP será aberta e através da qual o controle do simulador será possível.

Número da porta
: Porta na qual o servidor remoto estará aberto.

Configurações específicas do circuito

Configurações específicas do circuito afetarão o comportamento do circuito corrente. Por exemplo, o formato que representará o circuito quando integrado em outro. Essas configurações serão armazenadas juntamente com o circuito.

Atributos

Rótulo
: Nome do elemento.

Largura
: Largura do símbolo a ser usada se o circuito for componente de outro.

Cor de fundo (background)
: Cor de fundo para o circuito quando estiver integrado a outro. Não é usada em encapsulamentos DIL.

Descrição
: Uma breve descrição do elemento e como usá-lo.

Detecção de oscilação
: Número de ciclos de propagação para se detectar uma oscilação se o circuito não estiver estabilizado até então.

Modificação bloqueada
: O circuito está bloqueado. É possível configurar diodos e FGF-FETs.

Formato
: Formato a ser usado na representação de circuito integrado a outro. No modo "Simples", as entradas estarão mostradas à esquerda, e as saídas, à direita de um retângulo. Em "Layout", as posições de entradas e saídas, bem como suas orientações no circuito, definirão as posições dos pinos. Assim será possível ter pinos nas partes de cima e de baixo. Se selecionado "DIL-Chip", esse encapsulamento será usado para apresentar o circuito. Os números dos pinos das entradas e das saídas determinarão as posições dos pinos nesse caso.

Formato customizado
: Importar um arquivo SVG

Altura
: Altura do símbolo a ser usada se o circuito for componente de outro.

Número de pinos DIL
: Número de pinos. Se igual a zero, significa que o número de pinos será determinado automaticamente.

Conteúdo das ROM's
: Conteúdo de todas as ROM's usadas

Mostrar o gráfico de medidas juntamente com o início da simulação
: Quando a simulação iniciar, uma tabela com os valores medidos será mostrada.

Mostrar o gráfico de medidas juntamente com o início da simulação
: Ao iniciar a simulação, o gráifco de valores medidos também será mostrado.

Mostrar o gráfico de medidas juntamente com o início da simulação passo-a-passo
: Ao iniciar a simulação, o gráifco de valores medidos também será mostrado no modo passo-a-passo. Todas as mudanças em portas estarão relacionadas no gráfico.

Número máximo de passos a serem mostrados
: Número máximo de valores armazenados. Se o quantidade máxima for alcançada, os valores mais antigos serão descartados.

Memória de programa com carregamento prévio na inicialização.
: Quando da simulação de um processador que use uma RAM como memória de programa, é difícil iniciá-lo pois o conteúdo da RAM geralmente começa com zeros ao iniciar-se a simulação. Essa configuração permitirá a carregar previamente dados na memória de programa. Para isso, a memória de programa na simulação deverá ser marcada para tal.

Arquivo de programa
: Arquivo que deverá ser carregado na memória de programa ao iniciar-se a simulação a simulação.

Usar Big-Endian ao importar.
: Usar byte em representação Big-Endian ao importar.

Dispensar a exportação em Verilog/VHDL
: Dispensar a geração interna exportada pelo circuito em Verilog/VHDL As referências ao circuito serão mantidas, tornando possível sobrepor sua implementação.

Circuito é genérico
: Permite criar um circuito genérico.

C Interface para a linha de comandos

java -cp Digital.jar CLI

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

O primeiro nome de arquivo especificará o circuito a ser testado. Se um segundo nome de arquivo for especificado, os casos de testes serão executados a partir desse arquivo. Se não houver um segundo nome de arquivo especificado, os testes serão executados a partir do primeiro arquivo.

Opções:

-circ [String(def:)]

Nome do arquivo a ser testado.

[-tests [String(def:)]]

Nome do arquivo com os casos para testes.

[-allowMissingInputs(def: false)]

Permite-se que entradas faltantes no circuito sejam definidas no caso de teste. Isso poderá ser útil se houver várias soluções que possam depender de diferentes entradas.

[-verbose(def: false)]

Se definido, a tabela de valores será exibida em caso de erro.

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

Poderá ser usado para criar um arquivo SVG a partir de um circuito.

Opções:

-dig [String(def:)]

O nome do arquivo do circuito.

[-svg [String(def:)]]

O nome do arquivo SVG a ser gravado.

[-ieee(def: false)]

Usar símbolos IEEE.

[-LaTeX(def: false)]

Texto a ser inserido em notação LaTeX. Inkscape será necessário para processamento futuro.

[-pinsInMathMode(def: false)]

Para os rótulos de pinos, usar o modo matemático mesmo se não contiver índices.

[-hideTest(def: false)]

Ocultar os casos de testes.

[-noShapeFilling(def: false)]

Polígonos não preenchidos.

[-smallIO(def: false)]

Entradas e saídas serão representadas por círculos pequenos.

[-noPinMarker(def: false)]

Os marcadores de pinos nas cores azul e vermelha serão omitidos.

[-thinnerLines(def: false)]

Se selecionado, as linhas serão traçadas ligeiramente mais estreitas.

[-highContrast(def: false)]

As conexões e o texto dos pinos serão exibidos em preto.

[-monochrome(def: false)]

Apenas tons de cinza serão usados.

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

Criar arquivo CSV contendo estatísticas do circuito. Todos os components usados serão listados no arquivo CSV.

Opções:

-dig [String(def:)]

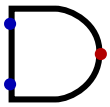
Nome do arquivo contendo circuito.

[-csv [String(def:)]]

Nome do arquivo CSV a ser criado. Se essa opção for omitida, a tabela será escrita na saída padrão.

D Componentes

1. Lógica



1.1. AND

Porta AND. Retornará 1 somente se todas as entradas também estiverem em 1. É possível usar barramentos com vários bits tanto para entradas quanto para saídas. Nesse caso, serão executados bit-a-bit. Isso significa que os bits de menor ordem de todas as entradas estarão conectados à AND e sua saída também será a de menor ordem. O mesmo ocorrerá para o bit 1, bit 2 e assim por diante. Exportável para VHDL/Verilog.

Entradas

In_1
: Valor de entrada para a operação lógica em 1.

In_2
: Valor de entrada para a operação lógica em 2.

Saídas

out
: Retornar o resultado da operação lógica.

Atributos

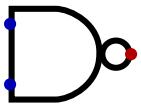
Bits de dados
: Número de bits de dados usados.

Número de entradas
: Número de entradas usado. Cada entrada necessitará estar contectada.

Entradas invertidas
: É possível selecionar entradas cujos valores serão invertidos.

Rotação
: Orientação do elemento no circuito.

Formato ampliado
: Usará um formato ampliado para visualizar a porta.



1.2. NAND

Combinação de AND e NOT. Retornará 0 somente se todas as entradas estiverem em 1. Se alguma das entradas estiver em 0 a saída irá para 1. Também é possível usar barramentos com vários bits para as entradas. Nesse caso, a operação se aplicará a cada bit das entradas. Exportável para VHDL/Verilog.

Entradas

In_1
: Valor de entrada para a operação lógica em 1.

In_2
: Valor de entrada para a operação lógica em 2.

Saídas

out
: Retornar o resultado da operação lógica.

Atributos

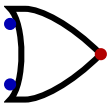
Bits de dados
: Número de bits de dados usados.

Número de entradas
: Número de entradas usado. Cada entrada necessitará estar contectada.

Entradas invertidas
: É possível selecionar entradas cujos valores serão invertidos.

Rotação
: Orientação do elemento no circuito.

Formato ampliado
: Usará um formato ampliado para visualizar a porta.



1.3. OR

Porta OR. Retornará 1 se alguma das entradas estiver em 1. Se todas as entradas estiverem em 0 a saída também irá para 0. Também é possível usar barramentos com vários bits tanto para entradas quanto para saídas. Nesse caso, serão executados bit-a-bit. Isso significa que os bits de menor ordem de todas as entradas estarão conectados à OR e sua saída também será a de menor ordem. O mesmo ocorrerá para o bit 1, bit 2 e assim por diante. Exportável para VHDL/Verilog.

Entradas

In_1
: Valor de entrada para a operação lógica em 1.

In_2
: Valor de entrada para a operação lógica em 2.

Saídas

out
: Retornar o resultado da operação lógica.

Atributos

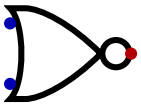
Bits de dados
: Número de bits de dados usados.

Número de entradas
: Número de entradas usado. Cada entrada necessitará estar contectada.

Entradas invertidas
: É possível selecionar entradas cujos valores serão invertidos.

Rotação
: Orientação do elemento no circuito.

Formato ampliado
: Usará um formato ampliado para visualizar a porta.



1.4. NOR

Combinação de OR e NOT. Retornará 0 se alguma das entradas estiver em 1. Se todas as entradas estiverem em 0 a saída irá para 1. Também é possível usar barramentos vários bits como entradas. Nesse caso, a operação se aplicará a cada bit das entradas. Exportável para VHDL/Verilog.

Entradas

In_1
: Valor de entrada para a operação lógica em 1.

In_2
: Valor de entrada para a operação lógica em 2.

Saídas

out
: Retornar o resultado da operação lógica.

Atributos

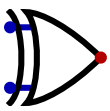
Bits de dados
: Número de bits de dados usados.

Número de entradas
: Número de entradas usado. Cada entrada necessitará estar contectada.

Entradas invertidas
: É possível selecionar entradas cujos valores serão invertidos.

Rotação
: Orientação do elemento no circuito.

Formato ampliado
: Usará um formato ampliado para visualizar a porta.



1.5. XOR

Se todas as entradas forem usadas, a saída será 0 para cada par de bits iguais. Caso contrário a saída será 1. Se mais do que duas entradas forem usadas, se comportará como portas XOR em cascata (A XOR B XOR C = (A XOR B) XOR C). Também é possível usar

barramentos com vários bits como entradas. Nesse caso, a operação se aplicará a cada bit das entradas. Exportável para VHDL/Verilog.

Entradas

In_1
: Valor de entrada para a operação lógica em 1.

In_2
: Valor de entrada para a operação lógica em 2.

Saídas

out
: Retornar o resultado da operação lógica.

Atributos

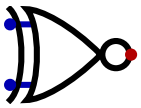
Bits de dados
: Número de bits de dados usados.

Número de entradas
: Número de entradas usado. Cada entrada necessitará estar contectada.

Entradas invertidas
: É possível selecionar entradas cujos valores serão invertidos.

Rotação
: Orientação do elemento no circuito.

Formato ampliado
: Usará um formato ampliado para visualizar a porta.



1.6. XNOR

Combinação de XOR e NOT. As entradas serão combinadas pela operação XOR. O resultado dessa operação será então invertido. Também é possível usar barramentos com vários bits como entradas. Nesse caso, a operação se aplicará a cada bit das entradas. Exportável para VHDL/Verilog.

Entradas

In_1
: Valor de entrada para a operação lógica em 1.

In_2
: Valor de entrada para a operação lógica em 2.

Saídas

out
: Retornar o resultado da operação lógica.

Atributos

Bits de dados
: Número de bits de dados usados.

Número de entradas
: Número de entradas usado. Cada entrada necessitará estar contectada.

Entradas invertidas
: É possível selecionar entradas cujos valores serão invertidos.

Rotação
: Orientação do elemento no circuito.

Formato ampliado
: Usará um formato ampliado para visualizar a porta.

1.7. NOT

Inverterá o valor à entrada. Um valor igual a 1 se tornará 0 e um 0 se tornará 1. Também é possível usar barramentos com vários bits como entradas. Nesse caso, a operação se aplicará a cada bit das entradas. Exportável para VHDL/Verilog.

Entradas

in
: A entrada da porta NOT.

Saídas

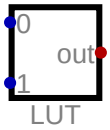
out
: O valor invertido da entrada.

Atributos

Bits de dados
: Número de bits de dados usados.

Rotação
: Orientação do elemento no circuito.

Formato ampliado
: Usará um formato ampliado para visualizar a porta.



1.8. LookUpTable

Obterá o valor da saída a partir de uma tabela armazenada. Dessa forma poderá emular qualquer porta combinatória. Exportável para VHDL/Verilog.

Entradas

0
: Entrada 0. Essa entrada em combinação com todas as outras definirá o endereço do valor armazenado que será retornado.

1
: Entrada 1. Essa entrada em combinação com todas as outras definirá o endereço do valor armazenado que será retornado.

Saídas

out
: Retornará o valor armazenado no endereço definido pelas entradas.

Atributos

Bits de dados
: Número de bits de dados usados.

Número de entradas
: Número de entradas usado. Cada entrada necessitará estar contectada.

Rótulo
: Nome do elemento.

Dados
: Os valores armazenados nesse elemento.

Rotação
: Orientação do elemento no circuito.

2. Entradas e saídas

2.1. Saída

Pode ser usada para mostrar um sinal de saída do circuito. Esse elemento também é usado para conectar um circuito a outro integrado. Nesse caso a conexão é bidirecional. Também é usado para atribuir um número PIN, se código para CPLD ou FPGA for gerado. Exportável para VHDL/Verilog.

Entradas

in
: Este valor é usado para a conexão de saída.

Atributos

Bits de dados
: Número de bits de dados usados.

Rótulo
: Nome do elemento.

Descrição
: Uma breve descrição do elemento e como usá-lo.

Rotação
: Orientação do elemento no circuito.

Formato para número
: Formato a ser usado para mostrar números.

dígitos de ponto fixo
: Número de dígitos binários na parte fracionária

Número do pino
: Número desse pino. Usado para a representação de um circuito com encapsulamento DIL e quando a a atribuição ao pino for usada na programação de um CPLD. Se houver vários bits, todos os números de pinos poderão ser especificados por uma lista, separados por vírgulas.

Mostrar medida graficamente
: Mostrar o valor de uma medida graficamente.

Forma pequena
: Se selecionada, uma forma pequena será usada.

2.2. LED

Um LED pode ser usado para visualizar um valor de saída. Aceita um único bit. Acenderá se a entrada estiver em 1.

Entradas

in
: Entrada do LED que acenderá se estiver em 1.

Atributos

Rótulo
: Nome do elemento.

Cor
: A cor do elemento.

Rotação
: Orientação do elemento no circuito.

Tamanho
: Tamanho do formato no circuito.

2.3. Entrada

Poderá ser usado para manipuar interativamente um sinal de entrada em um circuito com o mouse. Esse elemento também poderá ser usado para ligar um circuito a outro integrado. Nesse caso a conexão é bidirecional. Também é usado para atribuir um número PIN, se código para CPLD ou FPGA for gerado. Exportável para VHDL/Verilog.

Saídas

out
: Fornece o valor ao qual está ligada a essa entrada.

Atributos

Bits de dados
: Número de bits de dados usados.

Rótulo
: Nome do elemento.

Descrição
: Uma breve descrição do elemento e como usá-lo.

Rotação
: Orientação do elemento no circuito.

Valor padrão
: Valor definido para o circuito quando for ligado. "Z" significa alta impedância.

Entrada tri-state
: Se definido, permite-se que a entrada esteja em alta impedância. Em um componente de entrada, isso também será permitido, se ("Z") estiver definido como valor padrão.

Nenhuma saída em zero.
: Evitar saída em zero. Isso será útil especialmente quando forem definidos circuitos com relés . Somente poderá ser ativada se uma saída em alta impedância for permitida.

Formato para número
: Formato a ser usado para mostrar números.

dígitos de ponto fixo
: Número de dígitos binários na parte fracionária

Número do pino
: Número desse pino. Usado para a representação de um circuito com encapsulamento DIL e quando a a atribuição ao pino for usada na programação de um CPLD. Se houver vários bits, todos os números de pinos poderão ser especificados por uma lista, separados por vírgulas.

Mostrar medida graficamente
: Mostrar o valor de uma medida graficamente.

Forma pequena
: Se selecionada, uma forma pequena será usada.

2.4. Entrada do Clock

Um sinal de clock. É possível controlá-lo por um clock de tempo real. Dependendo da complexidade do circuito, a frequência de clock alcançada pode ser menor que a do valor selecionado. Se a frequência for maior que 50Hz, a representação gráfica do circuito não será mais atualizada a cada ciclo de clock de modo que as cores dos fios também não mais serão atualizadas. Se um clock de tempo real não estiver ativado, o elemento poderá ser controlado por clicks do mouse. Também é usado para atribuir um número PIN, se código para CPLD ou FPGA for gerado. Exportável para VHDL/Verilog.

Saídas

C
: Alternará entre 0 e 1, de acordo com a frequência selecionado do clock.

Atributos

Rótulo
: Nome do elemento.

Iniciar o clock de tempo real
: Se habilitado o clock da execução será iniciado juntamente com a ligação do circuito

Frequência/Hz
: Frequência em tempo real a ser usada no clock de tempo real

Rotação
: Orientação do elemento no circuito.

Número do pino
: Número desse pino. Usado para a representação de um circuito com encapsulamento DIL e quando a a atribuição ao pino for usada na programação de um CPLD. Se houver vários bits, todos os números de pinos poderão ser especificados por uma lista, separados por vírgulas.

Forma pequena
: Se selecionada, uma forma pequena será usada.

2.5. Botão

Botão simples que voltará ao seu estado original quando liberado.

Saídas

out
: O sinal de saída do botão.

Atributos

Rótulo
: Nome do elemento.

Ativo em nível baixo
: Se selecionada a saída estará em nível baixo, quando o circuito estiver ativo.

Mapear para o teclado
: Botão mapeado para o teclado. Para usar as teclas do cursor, utilizar as teclas de direção UP, DOWN, LEFT ou RIGHT.

Rotação
: Orientação do elemento no circuito.

Mostrar medida graficamente
: Mostrar o valor de uma medida graficamente.

2.6. DIP Switch

Chave DIP que pode ter a saída alta ou baixa.

Saídas

out
: O valor da chave de saída.

Atributos

Rótulo
: Nome do elemento.

Descrição
: Uma breve descrição do elemento e como usá-lo.

Rotação
: Orientação do elemento no circuito.

Saída em nível alto
: Valor padrão de saída de um DIP switch quando a simulação iniciar.

•?

2.7. Ponta de prova

Valor medido que poderá ser exibido na saída gráfica de dados ou na tabela de medições. Esse componente poderá ser usado para observar facilmente valores de circuitos integrados. Não afeta a simulação.

Entradas

in
: O valor da medição.

Atributos

Rótulo
: Nome do elemento.

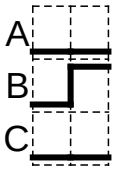
Modo de exibição
: Definir se o valor ou um contador deverá ser exibido.

Rotação
: Orientação do elemento no circuito.

Formato para número
: Formato a ser usado para mostrar números.

dígitos de ponto fixo
: Número de dígitos binários na parte fracionária

Mostrar medida graficamente
: Mostrar o valor de uma medida graficamente.



2.8. Gráfico de dados

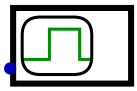
Mostrará um gráfico dentro do painel de circuito. Será possível exibir ciclos completos de clock ou mudanças de portas individuais. Não afeta a simulação.

Atributos

Mostrar passo-a-passo
: Mostrar todos os passos graficamente.

Número máximo de passos a serem mostrados
: Número máximo de valores armazenados. Se o quantidade máxima for alcançada, os valores mais antigos serão descartados.

Aderência à grade
: Se selecionado, o componente se alinhará à grade.



2.9. Gráficos com dados amostrados

Mostrará o gráfico de valores amostrados, os quais serão armazenados apenas se houver alteração no sinal de entrada. O armazenamento se dará quando o circuito estiver estabilizado. O gatilho não iniciará a medida como em um osciloscópio real, mas cada evento gatilhado armazenará um valor de medida apenas, para cada um dos sinais amostrados. Como entrada direta haverá apenas um gatilho. As entradas e saídas do circuito, flip-flops e registradores e o componente amostrador poderão ser usados como sinais. Isso poderá ser ativado nos respectivos componentes.

Entradas

T
: Uma alteração nessa entrada provocará o armazenamento de valores amostrados.

Atributos

Rótulo
: Nome do elemento.

Gatilho
: Condição de gatilho para o registro de dados.

Número máximo de passos a serem mostrados
: Número máximo de valores armazenados. Se o quantidade máxima for alcançada, os valores mais antigos serão descartados.

3. Entradas e saídas - Displays

3.1. LED RGB

Um LED RGB pode ter sua cor controlada através de três entradas. Em cada uma das três entradas, um canal de cor estará ligado.

Entradas

R
: O canal da cor vermelha.

G
: O canal da cor verde.

B
: O canal da cor azul.

Atributos

Bits de dados
: Número de bits de dados usados.

Rótulo
: Nome do elemento.

Rotação
: Orientação do elemento no circuito.

Tamanho
: Tamanho do formato no circuito.

3.2. LED com duas conexões.

LED com conexões para o cátodo e o ânodo. O LED acenderá, se o ânodo estiver ligado em nível alto e o cátodo estiver ligado em nível baixo. Esse LED não poderá ser usado como resistor pull-down. Só servirá como elemento para exibição. O resistor representado servirá apenas para simbolizar o equivalente aos necessários resistores em série usados para limitar a corrente.

Entradas

A
: A conexão do ânodo do LED.

C
: A conexão do cátodo do LED.

Atributos

Rótulo
: Nome do elemento.

Cor
: A cor do elemento.

Rotação
: Orientação do elemento no circuito.

3.3. Botão com LED

Um botão simples que voltará ao seu estado original quando não estiver mais acionado. O botão possui um LED que poderá ser comutado via um sinal de entrada.

Entradas

in
: Entrada para controlar o LED.

Saídas

out
: O sinal de saída do botão.

Atributos

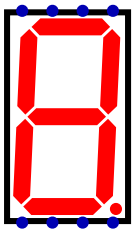
Rótulo
: Nome do elemento.

Ativo em nível baixo
: Se selecionada a saída estará em nível baixo, quando o circuito estiver ativo.

Mapear para o teclado
: Botão mapeado para o teclado. Para usar as teclas do cursor, utilizar as teclas de direção UP, DOWN, LEFT ou RIGHT.

Cor
: A cor do elemento.

Rotação
: Orientação do elemento no circuito.



3.4. Display de 7-Segmentos

Display de 7-Segmentos, cada segmento possui sua própria entrada de controle.

Entradas

a
: Essa entrada controlará a linha horizontal superior.

b
: Essa entrada controlará a linha vertical superior direita.

c
: Essa entrada controlará a linha vertical inferior direita.

d
: Essa entrada controlará a linha horizontal inferior.

e
: Essa entrada controlará a linha vertical inferior esquerda.

f
: Essa entrada controlará a linha vertical superior esquerda.

g
: Essa entrada controlará a linha horizontal ao meio.

dp
: Essa entrada controlará o ponto decimal.

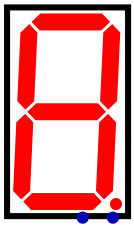
Atributos

Cor
: A cor do elemento.

Conexão comum
: Se a entrada for selecionada como ânodo ou cátodo comum, também será simulada.

Comum
: Tipo da conexão comum.

Persistência da visualização
: Especificar a duração de brilho. Quanto maior o valor, maior a duração do brilho.



3.5. Display de 7-Segmentos Hexadecimal

Display de 7-Segmentos com entrada hexadecimal de 4 bits

Entradas

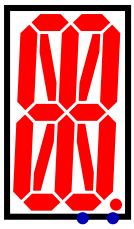
d
: O valor dessa entrada será exibido no display.

dp
: Essa entrada controlará o ponto decimal.

Atributos

Cor
: A cor do elemento.

Tamanho
: Tamanho do formato no circuito.



3.6. Display de 16 Segmentos

A entrada do LED input tem 16 bits que controlarão os segmentos. A segunda entrada controlará o ponto decimal.

Entradas

led
: Barramento de 16-bits para controlar os LEDs.

dp
: Essa entrada controlará o ponto decimal.

Atributos

Cor
: A cor do elemento.

Tamanho
: Tamanho do formato no circuito.

3.7. Lâmpada

Lâmpada com duas conexões. Se houver fluxo de corrente, a lâmpada acenderá! A direção da corrente não importa. A lâmpada acenderá quando as entradas tiverem valores diferentes. A lâmpada tem funcionamento similar ao da porta XOR.

Entradas

A
: Conexão

B
: Conexão

Atributos

Rótulo
: Nome do elemento.

Cor
: A cor do elemento.

Rotação
: Orientação do elemento no circuito.

Matriz de LEDs

3.8. Matriz de LEDs

A matriz de LEDs. Os LEDs serão exibidos em uma janela em separado. Os LEDs de uma coluna do display serão controlados por uma palavra de controle. Em outra entrada, a coluna corrente será selecionada. Dessa forma um display multiplexado será obtido. Os LEDs são capazes de se manter acesos indefinidamente durante a simulação para evitar que o display fique piscando.

Entradas

r-data
: A fileira de LEDs de uma coluna. Cada bit nessa palavra de dados representa o estado da coluna corrente.

c-addr
: O número da coluna corrente cujo estado estará visível segundo a outra entrada.

Atributos

Rótulo
: Nome do elemento.

Linhas
: Especificação do número de linhas mediante a largura em bits da palavra correspondente.

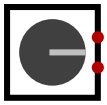
Bits de endereço das colunas
: Endereços das colunas individuais. Três bits corresponderão a oito colunas.

Cor
: A cor do elemento.

Prevenir o piscamento (flicker)
: Não é possível aumentar a frequência o bastante para que o efeito de piscamento possa desaparecer. Para suprimir o piscamento, um "afterglow" poderá ser acionado para manter os LEDs nessa condição. Se selecionado, os LEDs se manterão ligados, mesmo que um dos pinos for alterado para nível de alta impedância (z). Isso simulará uma frequência acima da crítica para a fusão.

Rotação
: Orientação do elemento no circuito.

4. Entradas e saídas - Mecânica



4.1. Codificador rotativo

Disco giratório com codificador rotativo. Usado para detectar movimentos de rotação.

Saídas

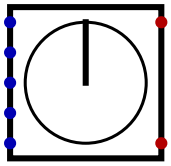
A
: sinal A do codificador

B
: sinal A do codificador

Atributos

Rótulo
: Nome do elemento.

Rotação
: Orientação do elemento no circuito.



4.2. Motor de passos, unipolar

Motor de passos unipolar com duas chaves de fim de curso. Há suporte para controle por passo completo, meio-passo e .

Entradas

P0
: Fase 0

P1
: Fase 1

P2
: Fase 2

P3
: Fase 3

com
: Conexão da bobina com derivação central

Saídas

S0
: Chave de fim de curso 0, será 1 quando o ângulo do motor for 0°.

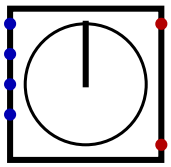
S1
: Chave de fim de curso 1, será 1 quando o ângulo do motor for 180°.

Atributos

Rótulo
: Nome do elemento.

Saída invertida
: Se definido, inverterá a saída.

Rotação
: Orientação do elemento no circuito.



4.3. Motor de passos, bipolar

Motor de passos bipolar com duas chaves de fim de curso. Há suporte para controle por passo completo, meio passo e .

Entradas

A+
: Bobina A, positivo

A-
: Bobina A, negativo

B+
: Bobina B, positivo

B-
: Bobina B, negativo

Saídas

S0
: Chave de fim de curso 0, será 1 quando o ângulo do motor for 0°.

S1
: Chave de fim de curso 1, será 1 quando o ângulo do motor for 180°.

Atributos

Rótulo
: Nome do elemento.

Saída invertida
: Se definido, inverterá a saída.

Rotação
: Orientação do elemento no circuito.

5. Entradas e saídas - Periféricos-

5.1. Teclado

Um teclado poderá ser usado para realizar entrada de texto. Esse componente receberá a entrada, a qual poderá ser lida posteriormente. Uma janela em separado será aberta para a entrada do texto.

Entradas

C
: Clock. Quando da borda de subida, o caractere mais antigo será removido do buffer.

en
: Se em nível alto, a saída D estará ativa e um caractere estará à saída. Isso também habilitará a entrada de clock.

Saídas

D
: O último caractere digitado, ou zero se nenhum caractere estiver disponível. A saída será um caractere Java com 16 bits.

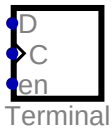
av
: Essa saída indicará que caracteres estarão disponíveis. Isso poderá ser usado para chavear uma interrupção.

Atributos

Rótulo
: Nome do elemento.

Entradas invertidas
: É possível selecionar entradas cujos valores serão invertidos.

Rotação
: Orientação do elemento no circuito.



5.2. Terminal

Será possível escrever caracteres ASCII nesse terminal. O terminal abrirá sua própria janela para exibir a saída.

Entradas

D
: Os dados a serem escritos no terminal

C
: Clock. Quando da borda de subida, enviará o valor da entrada para a janela do terminal.

en
: Um nível alto nessa entrada habilitará o recebimento do clock.

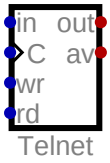
Atributos

Caracteress por linha
: Número de caracteres a serem mostrados em cada linha.

Linhas
: Número de linhas a serem mostradas.

Rótulo
: Nome do elemento.

Rotação
: Orientação do elemento no circuito.



5.3. Telnet

Permitir uma conexão Telnet ao circuito É possível receber e enviar caracteres via Telnet.

Entradas

in
: Dado a ser enviado.

C
: Entrada de clock

wr
: Se definido, o byte de dado à entrada será enviado.

rd
: Se definido, um byte recebido será emitido.

Saídas

out
: Saída de dados

av
: Emitir um dado, se existir.

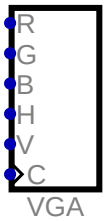
Atributos

Rótulo
: Nome do elemento.

Modo Telnet
: Se definido, os comandos de controle Telnet serão avaliados. Além disso, o servidor enviará os comandos SGA e ECHO. Se essa opção for desabilitada, irá tornar-se um simples servidor TCP.

Porta
: Porta a ser aberta pelo servidor.

Rotação
: Orientação do elemento no circuito.



5.4. Monitor VGA

Analisar sinais de entrada de vídeo e exibir o correspondente gráfico. Como a simulação não ocorre em tempo real, além dos sinais de vídeo também será requerido o sinal de clock para os pixels.

Entradas

R
: O componente para cor vermelha.

G
: O componente para cor verde

B
: O componente para a cor azul

H
: O sinal de sincronismo horizontal

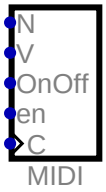
V
: O sinal de sincronismo vertical

C
: O clock para os pixels

Atributos

Rótulo
: Nome do elemento.

Rotação
: Orientação do elemento no circuito.



5.5. MIDI

Usar o sistema MIDI para tocar notas.

Entradas

N
: Nota

V
: Volume

OnOff
: Se definido, isso se traduzirá no pressionar de uma tecla (key down event), caso contrário, isso se traduzirá no liberar de uma tecla (key up event).

en
: Habilitar o componente

C
: Clock

Atributos

Rótulo
: Nome do elemento.

Canal MIDI
: Selecionar o canal MIDI a ser usado.

Instrumento MIDI
: O instrumento MIDI a ser usado.

Permitir alteração de programa
: Adicionar uma nova entrada PC. Se essa entrada estiver em nível alto, o valor da entrada N será usado para alterar o programa (instrumento).

Rotação
: Orientação do elemento no circuito.

6. Conexões

6.1. Terra

Conexão para terra. A saída será sempre igual a zero. Exportável para VHDL/Verilog.

Saídas

out
Saída sempre retornará 0.

Atributos

Bits de dados
Número de bits de dados usados.
Rótulo
Nome do elemento.
Rotação
Orientação do elemento no circuito.

6.2. Fonte

Conexão à fonte. A saída será sempre igual a um. Exportável para VHDL/Verilog.

Saídas

out
Saída sempre retornará 1.

Atributos

Bits de dados
Número de bits de dados usados.
Rótulo
Nome do elemento.
Rotação
Orientação do elemento no circuito.

6.3. Valor constante

Componente que retornará um dado valor constante. O valor poderá ser atribuído por caixa de diálogo. Exportável para VHDL/Verilog.

Saídas

out
Retornará o valor constante dado.

Atributos

Bits de dados
Número de bits de dados usados.
Valor
Valor de uma constante.
Rotação
Orientação do elemento no circuito.

Formato para número
: Formato a ser usado para mostrar números.

dígitos de ponto fixo
: Número de dígitos binários na parte fracionária

6.4. Túnel

Conectará componentes sem fios. Todos os elementos desse tipo, cujos nomes serão os mesmos, estarão conectados em conjunto. Funcionarão apenas localmente, não sendo possível usá-los em circuitos diferentes. Túneis sem identificação serão ignorados. Exportável para VHDL/Verilog.

Entradas

in
: A conexão a um túnel.

Atributos

Nome da rede
: ATodas as redes com nomes idênticas estarão ligadas entre si.

Rotação
: Orientação do elemento no circuito.

6.5. Distribuidor

Distribuir ou criar um conjunto de fios ou um barramento de dados com mais de um bit. Com um barramento será possível, por exemplo, gerar conexões de 16-bits sem ter quer rotear 16 fios individuais. Todas as 16 conexões serão reunidas em um único cabo. O distribuidor terá uma direção, isso significa que os sinais serão transmitidos em apenas um sentido. Exportável para VHDL/Verilog.

Entradas

0-3
: Os bits de entrada 0-3.

4-7
: Os bits de entrada 4-7.

Saídas

0-7
: Os bits de saída 0-7.

Atributos

Entrada para o distribuidor
: Se, por exemplo, quatro bits, dois bits mais outros dois serão usados como entradas, isso poderá ser configurado como "4,2,2". A quantidade indicará o número de bits. Por conveniência, o asterisco poderá ser usado: 16 bits poderão ser configurados como "[bits]*[quantidade]", ou seja, "1*16". Também é possível especificar os bits a serem usados diretamente e em qual ordem. Por exemplo, "4-7,0-3" configurará os bits 4-7 e 0-3. Essa notação permitirá qualquer arranjo de bits desejado. Os bits de entrada deverão ser especificados de forma completa e inequívoca.

Saída do distribuidor
: Se, por exemplo, quatro bits, dois bits mais outros dois serão usados como saídas, isso poderá ser configurado como "4,2,2". A quantidade indicará o número de bits. Por conveniência, o asterisco poderá ser usado: 16 bits poderão ser configurados como "[bits]*[quantidade]", ou seja, "1*16". Também é possível especificar os bits a serem usados diretamente e em qual ordem. Os bits de saída poderão ser combinados várias vezes: "0-7,1-6,4-7"

Rotação
: Orientação do elemento no circuito.

Espelhamento
: Espelhar o componente em um circuito.

Propagação
: Configuração de como irão se propagar as entradas e saídas no circuito.

6.6. Driver

Um driver poderá ser usado para conectar um sinal a outro fio. O driver é controlado pela entrada definida. Se o sinal de entrada estiver em nível baixo, a saída estará em estado de alta impedância. Se o sinal de saída estiver em nível alto, a saída será definida pelo valor à entrada. Exportável para VHDL/Verilog.

Entradas

in
: O valor de entrada do driver.

sel
: Pino para controlar o driver. Se sua entrada for 1, a saída terá o valor dado à entrada. Se sua entrada for 0, a saída estará em estado de alta impedância.

Saídas

out
: Se a entrada for 1, a saída terá o valor da entrada. Se seu valor for 0, a saída estará em estado de alta impedância.

Atributos

Bits de dados
: Número de bits de dados usados.

Alternar posição do seletor
: Essa opção permitirá deslocar o pino do seletor para o lado oposto.

Rotação
: Orientação do elemento no circuito.

6.7. Driver invertido

Um driver poderá ser usado para conectar um sinal a outro fio O driver é controlado pela entrada definida. Se sua entrada estiver em nível alto, a saída estará em estado de alta impedância. Se o sinal de saída estiver em nível baixo, a saída será definida pelo valor à entrada. Exportável para VHDL/Verilog.

Entradas

in
: Valor de entrada do driver.

sel
: Pino para controlar o driver. Se sua entrada for 0, a saída terá o valor dado à entrada. Se sua entrada for 1, a saída estará em estado de alta impedância.

Saídas

out
: Se a entrada for 1, a saída terá o valor da entrada. Se seu valor for 0, a saída estará em estado de alta impedância.

Atributos

Bits de dados
: Número de bits de dados usados.

Alternar posição do seletor
: Essa opção permitirá deslocar o pino do seletor para o lado oposto.

Rotação
: Orientação do elemento no circuito.

6.8. Atraso

Atrasará o sinal no tempo de propagação. Atrasará um sinal por um número de atrasos de porta regulável. Todos os outros componentes no simulador têm um atraso de porta no tempo de propagação. Esse componente pode ser usado para implementar qualquer atraso de propagação necessário.

Entradas

in
: Entrada do sinal a ser atrasado.

Saídas

out
: O sinal de entrada atrasado pelo tempo correspondente a um atraso de porta.

Atributos

Bits de dados
: Número de bits de dados usados.

Duração
: Unidades de tempo do atraso de propagação das portas comuns.

Rotação
: Orientação do elemento no circuito.

6.9. Resistor Pull-Up

Se o circuito estiver em estado de alta impedância, o resistor levará o valor para nível alto. Em qualquer outro caso, esse componente não terá efeito.

Saídas

out
: Um nível alto "fraco".

Atributos

Bits de dados
: Número de bits de dados usados.

Rotação
: Orientação do elemento no circuito.

6.10. Resistor Pull-Down

Se o circuito estiver em estado de alta impedância, o resistor levará o valor para nível baixo. Em qualquer outro caso, esse componente não terá efeito.

Saídas

out
: Um nível baixo "fraco".

Atributos

Bits de dados
: Número de bits de dados usados.

Rotação
: Orientação do elemento no circuito.

6.11. Desconectado

Esse componente poderá ser usado para colocar uma conexão em alta impedância (Z). Se um entrada de uma porta lógica estiver em alta impedância, o valor lido será indefinido. Observa-se que, na realidade, em muitos casos, o consumo excessivo de corrente e até mesmo danos poderão ocorrer se uma entrada digital não estiver definida como alta ou baixa, mas permanecer desconectada.

Saídas

out
: Esta saída estará sempre em alta impedância.

Atributos

Bits de dados
: Número de bits de dados usados.

7. Plexers

7.1. Multiplexador

Componente que usará o valor de entrada de seleção para direcionar um dos valor escolhido dentre as entradas para a saída. Exportável para VHDL/Verilog.

Entradas

sel
: Essa entrada será usada para selecionar os dados à entrada que serão direcionados para a saída.

in_0
: O valor da primeira entrada do multiplexador 0.

in_1
: O valor da primeira entrada do multiplexador 1.

Saídas

out
: O valor da entrada selecionada.

Atributos

Bits de dados
: Número de bits de dados usados.

Número de bits do seletor
: Número de bits usados na entrada do seletor.

Alternar posição do seletor
: Essa opção permitirá deslocar o pino do seletor para o lado oposto.

Rotação
: Orientação do elemento no circuito.

7.2. Demultiplexador

Componente que poderá direcionar um valor de entrada para uma saída escolhida. As outras saídas terão o valor padrão definido. Exportável para VHDL/Verilog.

Entradas

sel
: O pino que seleciona a saída a ser usada.

in
: O valor dessa entrada será dado à saída selecionada.

Saídas

out_0
: Primeira saída de dados 0.

out_1
: Primeira saída de dados 1.

Atributos

Bits de dados
: Número de bits de dados usados.

Número de bits do seletor
: Número de bits usados na entrada do seletor.

Alternar posição do seletor
: Essa opção permitirá deslocar o pino do seletor para o lado oposto.

Rotação
: Orientação do elemento no circuito.

Valor padrão
: Valor definido para o circuito quando for ligado. No demultiplexador, esse valor será definido para as saídas não selecionadas.

7.3. Decodificador

Um dos pinos de saída estará em 1, todos as outras saídas estarão em 0. Exportável para VHDL/Verilog.

Entradas

sel
: Essa entrada selecionará a saída. A saída selecionda terá o valor igual a 1. Todas as outras saídas serão iguais a 0.

Saídas

out_0
: Saída 0. essa saída estará em 1 de acordo com a entrada de seleção.

out_1
: Saída 1. essa saída estará em 1 de acordo com a entrada de seleção.

Atributos

Número de bits do seletor
: Número de bits usados na entrada do seletor.

Alternar posição do seletor
: Essa opção permitirá deslocar o pino do seletor para o lado oposto.

Rotação
: Orientação do elemento no circuito.

7.4. Seletor de Bit

Seleciona um único bit do barramento de dados. Exportável para VHDL/Verilog.

Entradas

in
: O barramento de entrada

sel
: Essa entrada selecionará o bit

Saídas

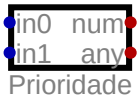
out
: O bit selecionado.

Atributos

Número de bits do seletor
: Número de bits usados na entrada do seletor.

Alternar posição do seletor
: Essa opção permitirá deslocar o pino do seletor para o lado oposto.

Rotação
: Orientação do elemento no circuito.



7.5. Codificador de prioridade

Se um das entradas estiver definida, a saída será o seu número. Se várias entradas estiverem definidas ao mesmo tempo, a saída será igual ao número mais alto. Exportável para VHDL/Verilog.

Entradas

in0
: O valor da primeira entrada do codificador de prioridade 0.

in1
: O valor da primeira entrada do codificador de prioridade 1.

Saídas

num
: Número da entrada definida.

any
: Se essa saída estiver definida, pelo menos uma das entradas também estará.

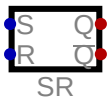
Atributos

Rótulo
: Nome do elemento.

Número de bits do seletor
: Número de bits usados na entrada do seletor.

Rotação
: Orientação do elemento no circuito.

8. Flip-Flops



8.1. Flip-flop SR

Componente que armazenará um único bit. Oferece funções para armazenar e limpar determinado bit. Se ambas as entradas forem para 1, ambas as saídas também irão para 1. Se ambas as entradas forem para 0, ao mesmo tempo, o estado final será aleatório.

Entradas

S
: A entrada para armazenar.

R
: A entrada para limpar.

Saídas

Q
: Retornará o valor armazenado.

¬Q
: Retornará o inverso do valor armazenado.

Atributos

Rótulo
: Nome do elemento.

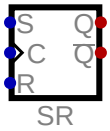
Entradas invertidas
: É possível selecionar entradas cujos valores serão invertidos.

Rotação
: Orientação do elemento no circuito.

Espelhamento
: Espelhar o componente em um circuito.

Valor padrão
: Valor definido para o circuito quando for ligado. No demultiplexador, esse valor será definido para as saídas não selecionadas.

Usar como valor medido
: Se definido, o valor será usado como o de medição, e será mostrado no gráfico e também na tabela de dados. Além disso, um rótulo deverá ser especificado para servir como identificação do valor.



8.2. Flip-flop SR submetido ao clock

Componente que armazenará um único bit. Oferece funções para armazenar e limpar determinado bit. Se ambas as entradas (S, R) estiverem em nível alto quando da subida do clock, o estado final será aleatório.

Entradas

S
: A entrada para armazenar.

C
: A entrada do clock. A subida do clock inicia o processo de transição.

R
: A entrada para limpar.

Saídas

Q
: Retornará o valor armazenado.

¬Q
: Retornará o inverso do valor armazenado.

Atributos

Rótulo
: Nome do elemento.

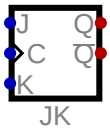
Entradas invertidas
: É possível selecionar entradas cujos valores serão invertidos.

Rotação
: Orientação do elemento no circuito.

Espelhamento
: Espelhar o componente em um circuito.

Valor padrão
: Valor definido para o circuito quando for ligado. No demultiplexador, esse valor será definido para as saídas não selecionadas.

Usar como valor medido
: Se definido, o valor será usado como o de medição, e será mostrado no gráfico e também na tabela de dados. Além disso, um rótulo deverá ser especificado para servir como identificação do valor.



8.3. Flip-flop JK

Tem a possibilidade para manter (J=K=0), levar para nível alto (J=1, K=0), levar para níve baixo (J=0, K=1) ou trocar (J=K=1) o valor armazenado. A mudança de estado ocorrerá somente quando houver uma borda de subida na entrada do clock C. Exportável para VHDL/Verilog.

Entradas

J
: A entrada para armazenar no flip-flop.

C
: A entrada do clock. Uma borda de subida iniciará a transição de estado.

K
: A entrada para limpar o flip-flop.

Saídas

Q
: Retornará o valor armazenado.

¬Q
: Retornará o inverso do valor armazenado.

Atributos

Rótulo
: Nome do elemento.

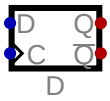
Entradas invertidas
: É possível selecionar entradas cujos valores serão invertidos.

Rotação
: Orientação do elemento no circuito.

Espelhamento
: Espelhar o componente em um circuito.

Valor padrão
: Valor definido para o circuito quando for ligado. No demultiplexador, esse valor será definido para as saídas não selecionadas.

Usar como valor medido
: Se definido, o valor será usado como o de medição, e será mostrado no gráfico e também na tabela de dados. Além disso, um rótulo deverá ser especificado para servir como identificação do valor.



8.4. Flip-flop D

Componente que armazenará um único bit. O valor presente no pino D será armazenado na subida do clock no pino C. A largura de bits poderá ser selecionada, o que permitirá armazenar múltiplos bits. Exportável para VHDL/Verilog.

Entradas

D
: Entrada para o bit a ser armazenado.

C
: Entrada do clock para armazenar um valor. O valor presente no pino D será armazenado na subida do clock nesse pino.

Saídas

Q
: Retornará o valor armazenado.

¬Q
: Retornará o inverso do valor armazenado.

Atributos

Bits de dados
: Número de bits de dados usados.

Rótulo
: Nome do elemento.

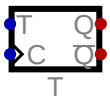
Entradas invertidas
: É possível selecionar entradas cujos valores serão invertidos.

Rotação
: Orientação do elemento no circuito.

Espelhamento
: Espelhar o componente em um circuito.

Valor padrão
: Valor definido para o circuito quando for ligado. No demultiplexador, esse valor será definido para as saídas não selecionadas.

Usar como valor medido
: Se definido, o valor será usado como o de medição, e será mostrado no gráfico e também na tabela de dados. Além disso, um rótulo deverá ser especificado para servir como identificação do valor.



8.5. Flip-Flop T

Armazenará um único bit. Alternará o valor a cada borda de subida na entrada C.

Entradas

T
: Habilitará a função de troca.

C
: Entrada de clock. A cada borda de subida, alternará a saída, se a entrada T estiver em 1.

Saídas

Q
: Retornará o valor armazenado.

¬Q
: Retornará o inverso do valor armazenado.

Atributos

Rótulo
: Nome do elemento.

Entrada para habilitação
: Se definido, tornará a entrada de habilitação (T) disponível.

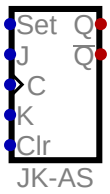
Entradas invertidas
: É possível selecionar entradas cujos valores serão invertidos.

Rotação
: Orientação do elemento no circuito.

Espelhamento
: Espelhar o componente em um circuito.

Valor padrão
: Valor definido para o circuito quando for ligado. No demultiplexador, esse valor será definido para as saídas não selecionadas.

Usar como valor medido
: Se definido, o valor será usado como o de medição, e será mostrado no gráfico e também na tabela de dados. Além disso, um rótulo deverá ser especificado para servir como identificação do valor.



8.6. Flip-flop JK, assíncrono

Tem a possibilidade para manter (J=K=0), levar para nível alto (J=1, K=0), levar para níve baixo (J=0, K=1) ou trocar (J=K=1) o valor armazenado. A mudança de estado ocorrerá somente quando houver uma borda de subida na entrada do clock C. Há duas entradas adicionais para armazenar ou limpar o estado sem a presença de um sinal de clock. Exportável para VHDL/Verilog.

Entradas

Set
: Habilitação assíncrona. Um valor em nível alto nessa entrada levará a saída para nível alto.

J
: A entrada para armazenar do flip-flop.

C
: A entrada do clock. Uma borda de subida iniciará a transição de estado.

K
: A entrada para limpar o flip-flop.

Clr
: Limpeza assíncrona. Um valor em nível alto nessa entraa levará a saída para nível baixo.

Saídas

Q
: Retornará o valor armazenado.

¬Q
: Retornará o inverso do valor armazenado.

Atributos

Rótulo
: Nome do elemento.

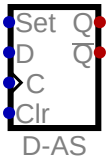
Entradas invertidas
: É possível selecionar entradas cujos valores serão invertidos.

Rotação
: Orientação do elemento no circuito.

Espelhamento
: Espelhar o componente em um circuito.

Valor padrão
: Valor definido para o circuito quando for ligado. No demultiplexador, esse valor será definido para as saídas não selecionadas.

Usar como valor medido
: Se definido, o valor será usado como o de medição, e será mostrado no gráfico e também na tabela de dados. Além disso, um rótulo deverá ser especificado para servir como identificação do valor.



8.7. Flip-flop D, assíncrono

Componente que armazenará um único bit. O valor presente no pino D será armazenado na subida do clock no pino C. Há duas entradas adicionais para armazenar ou limpar o estado imediatamente sem a presença de um sinal de clock. A largura de bits poderá ser selecionada, o que permitirá armazenar múltiplos bits. Exportável para VHDL/Verilog.

Entradas

Set
: Habilitação assíncrona. Um valor em nível alto nessa entrada levará a saída para nível alto.

D
: Entrada do bit a ser armazenado.

C
: Pino de controle para armazenar um bit. O valor presente no pino D será armazenado na subida do sinal nesse pino.

Clr
: Limpeza assíncrona. Um valor em nível alto nessa entrada levará a saída para nível baixo.

Saídas

Q
: Retornará o valor armazenado.

¬Q
: Retornará o inverso do valor armazenado.

Atributos

Bits de dados
: Número de bits de dados usados.

Rótulo
: Nome do elemento.

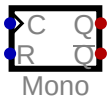
Entradas invertidas
: É possível selecionar entradas cujos valores serão invertidos.

Rotação
: Orientação do elemento no circuito.

Espelhamento
: Espelhar o componente em um circuito.

Valor padrão
: Valor definido para o circuito quando for ligado. No demultiplexador, esse valor será definido para as saídas não selecionadas.

Usar como valor medido
: Se definido, o valor será usado como o de medição, e será mostrado no gráfico e também na tabela de dados. Além disso, um rótulo deverá ser especificado para servir como identificação do valor.



8.8. Monoflop

O monoflop será determinado na subida da entrada de clock. Após um atrado de tempo configurável, o monoflop será limpo automaticamente. O monoflop poderá ser gatilhável. Isso somente será utilizado se existir apenas um componente de clock presente no circuito. Esse componente de clock será usado como base de tempo para medir os atrasos.

Entradas

C
: A entrada de clock. Uma borda de subida irá engatilhar o monoflop.

R
: Entrada para limpar. Um valor em nível alto irá limpar o monoflop.

Saídas

Q
: Saída

¬Q
: Saída invertida

Atributos

Rótulo
: Nome do elemento.

Largura de pulso
: A largura de pulso é medida em ciclos de clock.

Entradas invertidas
: É possível selecionar entradas cujos valores serão invertidos.

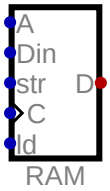
Rotação
: Orientação do elemento no circuito.

Espelhamento
: Espelhar o componente em um circuito.

Valor padrão
: Valor definido para o circuito quando for ligado. No demultiplexador, esse valor será definido para as saídas não selecionadas.

Usar como valor medido
: Se definido, o valor será usado como o de medição, e será mostrado no gráfico e também na tabela de dados. Além disso, um rótulo deverá ser especificado para servir como identificação do valor.

9. Memória - RAM



9.1. RAM, portas separadas

Módulo de RAM com portas separadas para armazenar e prover a saída para de dados armazenados. Exportável para VHDL/Verilog.

Entradas

A
: Endereço onde ler ou gravar.

Din
: Dados a serem armazenados.

str
: Se essa entrada estiver em nível alto, quando da borda de subida do clock, os dados serão armazenados.

C
: Entrada de clock

ld
: Se essa entrada estiver em nível alto, a saída estará ativada e os dados visíveis.

Saídas

D
: Pino para a saída de dados

Atributos

Bits de dados
: Número de bits de dados usados.

Bits de endereço
: Número de bits de endereço usado.

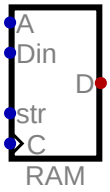
Rótulo
: Nome do elemento.

Rotação
: Orientação do elemento no circuito.

Formato para número
: Formato a ser usado para mostrar números.

dígitos de ponto fixo
: Número de dígitos binários na parte fracionária

Memória de programa
: Tomar o conteúdo da ROM como a memória de programa. Dessa forma, poderá ser acessado por uma IDE externa.



9.2. RAM-Bloco, portas separadas

Módulo de RAM com entradas separadas para armazenar dados, e saída para se ler os dados armazenados. Essa RAM somente atualizará sua saída na borda de subida da entrada de clock. Isso permitirá o uso dessa RAM em FPGA. Exportável para VHDL/Verilog.

Entradas

A
: Endereço onde ler ou gravar.

Din
: Dados a serem armazenados em RAM.

str
: Se essa entrada estiver em nível alto, quando houver borda de subida do clock, os dados serão armazenados.

C
: Entrada de clock

Saídas

D
: Pino de saída de dados

Atributos

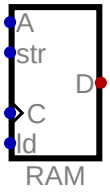
Bits de dados
: Número de bits de dados usados.

Bits de endereço
: Número de bits de endereço usado.

Rótulo
: Nome do elemento.

Rotação
: Orientação do elemento no circuito.

Memória de programa
: Tomar o conteúdo da ROM como a memória de programa. Dessa forma, poderá ser acessado por uma IDE externa.



9.3. RAM, porta bidirecional

Módulo de RAM com pino bidirecional para ler e gravar dados.

Entradas

A
: Endereço onde ler ou gravar.

str
: Se essa entrada estiver em nível alto, quando da borda de subida do clock, os dados serão armazenados.

C
: Entrada de clock

ld
: Se essa entrada estiver em nível alto, a saída estará ativada e os dados visíveis.

Saídas

D
: Conexão de dados bidirecional.

Atributos

Bits de dados
: Número de bits de dados usados.

Bits de endereço
: Número de bits de endereço usado.

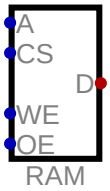
Rótulo
: Nome do elemento.

Rotação
: Orientação do elemento no circuito.

Formato para número
: Formato a ser usado para mostrar números.

dígitos de ponto fixo
: Número de dígitos binários na parte fracionária

Memória de programa
: Tomar o conteúdo da ROM como a memória de programa. Dessa forma, poderá ser acessado por uma IDE externa.



9.4. RAM, seleção de chip

Módulo de RAM com conexão bidirecional para ler e gravar dados. Se a entrada CS estiver em nível baixo, o componente estará desabilitado. Isso permitirá a construção de RAM's maiores a partir de módulos menores e um decodificador de endereços. O ciclo de escrita funcionará como se segue: ao colocar CS em nível alto, o componente será selecionado. Uma borda de subida em WE receberá o endereço, e na próxima borda de descida irá armazenar os dados.

Entradas

A
: Endereço onde ler ou gravar.

CS
: Se essa entrada estiver em nível alto, essa RAM estará habilitada. De outro modo, a saída estará sempre em estado de alta impedância.

WE
: Se essa entrada estiver em nível alto, os dados serão gravados na RAM.

OE
: Se essa entrada estiver em nível alto, o valor armazenado estará disponível na saída.

Saídas

D
: Conexão de dados bidirectional.

Atributos

Bits de dados
: Número de bits de dados usados.

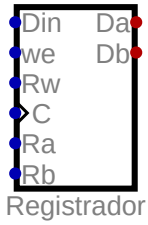
Bits de endereço
: Número de bits de endereço usado.

Rótulo
: Nome do elemento.

Entradas invertidas
: É possível selecionar entradas cujos valores serão invertidos.

Rotação
: Orientação do elemento no circuito.

Memória de programa
: Tomar o conteúdo da ROM como a memória de programa. Dessa forma, poderá ser acessado por uma IDE externa.



9.5. Bloco de registradores

Memória com uma porta que permitirá a gravação e duas portas que permitirão leituras simultâneas Poderá ser usada para implementar registradores em um processador. Dois registradores poderão ser lidos simultaneamente, enquanto um terceiro poderá ser escrito. Exportável para VHDL/Verilog.

Entradas

Din
: Dados a serem armazenados no registrador Rw.

we
: Se essa entrada estiver em nível alto, quando da borda de subida do clock, os dados serão armazenados.

Rw
: Registrador no qual os dados serão escritos.

C
: Clock

Ra
: Registrador visível à porta (a).

Rb
: Registrador visível à porta (b).

Saídas

Da
: Porta de saída (a)

Db
: Porta de saída (b)

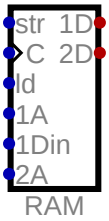
Atributos

Bits de dados
: Número de bits de dados usados.

Bits de endereço
: Número de bits de endereço usado.

Rótulo
: Nome do elemento.

Rotação
: Orientação do elemento no circuito.



9.6. RAM, duas portas

RAM com uma porta que permitirá leitura e gravação, e uma outra porta onde apenas a leitura é permitida. A segunda porta pdoerá ser usada para dar a alguma lógica envolvendo parte gráfica acesso aos conteúdos de memória. Dessa maneira, o processador poderá escrever na RAM, enquanto a parte gráfica poderá ler da mesma simultaneamente. Exportável para VHDL/Verilog.

Entradas

str
: Se essa entrada estiver em nível alto, quando da borda de subida do clock, os dados serão armazenados.

C
: Clock

ld
: Se essa entrada estiver em nível alto, a saída estará ativada e os dados estarão visíveis em 1D.

1A
: Endereço onde a porta 1 irá ler ou gravar.

1Din
: Dados a serem armazenados na RAM.

2A
: Endereço onde a porta 2 irá acessar para ler .

Saídas

1D
: Porta de saída 1

2D
: Porta de saída 2

Atributos

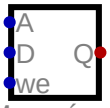
Bits de dados
: Número de bits de dados usados.

Bits de endereço
: Número de bits de endereço usado.

Rótulo
: Nome do elemento.

Rotação
: Orientação do elemento no circuito.

Memória de programa
: Tomar o conteúdo da ROM como a memória de programa. Dessa forma, poderá ser acessado por uma IDE externa.



RAM, assíncrona

9.7. RAM, assíncrona

Uma vez definido, será armazenado. Corresponde a uma RAM simples, onde as linhas de endereços e de dados estão diretamente conectadas aos decodificadores das células de memória. Exportável para VHDL/Verilog.

Entradas

A
: O endereço onde se fará a leitura ou a escrita.

D
: Os dados a serem armazenados

we
: Habilitar a escrita. Se essa entrada for igual a 1, o valor aplicado em D será armazenado no endereço aplicado em A, quando A ou D forem modificados.

Saídas

Q
: Saída dos dados armazenados.

Atributos

Bits de dados
: Número de bits de dados usados.

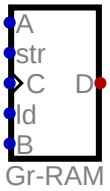
Bits de endereço
: Número de bits de endereço usado.

Entradas invertidas
: É possível selecionar entradas cujos valores serão invertidos.

Rótulo
: Nome do elemento.

Rotação
: Orientação do elemento no circuito.

Memória de programa
: Tomar o conteúdo da ROM como a memória de programa. Dessa forma, poderá ser acessado por uma IDE externa.



9.8. RAM gráfica

Usada para exibir gráficos bitmap. Tem comportamento semelhante a de uma. Além disso, exibirá seu conteúdo em ua janela gráfica. Cada pixel será representado por um endereço de memória. O valor armazenado definirá a cor do pixel, a partir de uma paleta de cores fixa. Haverá dois buffers de tela implementados para dar suporte à mudança de páginas. A entrada B selecionará o buffer a ser exibido. Assim, o tamanho total da memória será dx * dy * 2 palavras.

Entradas

A
: Endereço onde ler ou gravar.

str
: Se essa entrada estiver em nível alto, quando da borda de subida do clock, os dados serão armazenados.

C
: Clock

ld
: Se essa entrada estiver em nível alto, os dados armazenados estarão disponíveis na saída.

B
: Selecionará o buffer de tela a ser exibido.

Saídas

D
: Conexão de dados bidirectional.

Atributos

Bits de dados
: Número de bits de dados usados.

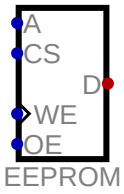
Rótulo
: Nome do elemento.

Largura em pixels
: Largura da tela em pixels.

Altura em pixels
: Altura da tela em pixels.

Rotação
: Orientação do elemento no circuito.

10. Memória - EEPROM



10.1. EEPROM

Módulo de EEPROM com conexão bidirecional para ler e gravar dados. Se a entrada CS estiver em nível baixo, o componente estará desabilitado. Os dados estarão armazenados tal como em uma ROM. Estarão assim preservados durante, e após a simulação terminar, e quando for reiniciada. O ciclo de escrita funcionará como se segue: ao colocar CS em nível alto, o componente será selecionado. Uma borda de subida em WE receberá o endereço, e na próxima borda de descida irá armazenar os dados.

Entradas

A
: Endereço onde ler ou gravar.

CS
: Se essa entrada estiver em nível alto, essa EEPROM estará habilitada. De outro modo, a saída estará sempre em estado de alta impedância.

WE
: Se essa entrada estiver em nível alto, os dados serão escritos na EEPROM.

OE
: Se essa entrada estiver em nível alto, o valor armazenado estará disponível na saída.

Saídas

D
: Conexão de dados bidirectional.

Atributos

Bits de dados
: Número de bits de dados usados.

Bits de endereço
: Número de bits de endereço usado.

Rótulo
: Nome do elemento.

Entradas invertidas
: É possível selecionar entradas cujos valores serão invertidos.

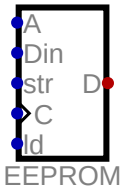
Dados
: Os valores armazenados nesse elemento.

Rotação
: Orientação do elemento no circuito.

Formato para número
: Formato a ser usado para mostrar números.

dígitos de ponto fixo
: Número de dígitos binários na parte fracionária

Memória de programa
: Tomar o conteúdo da ROM como a memória de programa. Dessa forma, poderá ser acessado por uma IDE externa.



10.2. EEPROM, portas separadas

Módulo de EEPROM com portas separadas para armazenar e prover a saída para de dados armazenados.

Entradas

A
: Endereço onde ler ou gravar.

Din
: Dados a serem armazenados

str
: Se essa entrada estiver em nível alto, quando da borda de subida do clock, os dados serão armazenados.

C
: Entrada de clock

ld
: Se essa entrada estiver em nível alto, a saída estará ativada e os dados visíveis.

Saídas

D
: Pino para a saída de dados

Atributos

Bits de dados
: Número de bits de dados usados.

Bits de endereço
: Número de bits de endereço usado.

Rótulo
: Nome do elemento.

Dados
: Os valores armazenados nesse elemento.

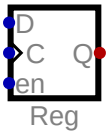
Rotação
: Orientação do elemento no circuito.

Formato para número
: Formato a ser usado para mostrar números.

dígitos de ponto fixo
: Número de dígitos binários na parte fracionária

Memória de programa
: Tomar o conteúdo da ROM como a memória de programa. Dessa forma, poderá ser acessado por uma IDE externa.

11. Memória



11.1. Registrador

Componente para armazenar valores. A largura em bits da palavra de dados poderá ser selecionada. Diferente do flip-flop D, o registrador possui uma entrada que o habilitará dependendo do clock. Exportável para VHDL/Verilog.

Entradas

D
: Entrada da palavra de dados a ser armazenada.

C
: Entrada de clock. Na borda de subida armazenará o valor presente no pino D.

en
: Pino para habilitação. O armazenamento do valor funcionará somente se esse pino estiver em nível alto.

Saídas

Q
: Retornará o valor armazenado.

Atributos

Bits de dados
: Número de bits de dados usados.

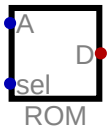
Rótulo
: Nome do elemento.

Entradas invertidas
: É possível selecionar entradas cujos valores serão invertidos.

Rotação
: Orientação do elemento no circuito.

Contador de programa
: Tomar o registrador como contador de programa. O valor do registrador será tomado a partir da IDE do montador externo para marcar a linha de código corrente durante a depuração.

Usar como valor medido
: Se definido, o valor será usado como o de medição, e será mostrado no gráfico e também na tabela de dados. Além disso, um rótulo deverá ser especificado para servir como identificação do valor.



11.2. ROM

Componente de memória não-volátil. Os dados armazenados poderão ser editados por meio da caixa de diálogo de atributos. Exportável para VHDL/Verilog.

Entradas

A
: Este pino definirá o endereço da palavra de dados a ser enviada para a saída.

sel
: Se a entrada estiver em nível alto, a saída será ativada. Se estiver em nível baixo, a saída de dados estará em estado de alta impedância.

Saídas

D
: A palavra de dados selecionada se a entrada estiver em nível alto.

Atributos

Bits de dados
: Número de bits de dados usados.

Bits de endereço
: Número de bits de endereço usado.

Rótulo
: Nome do elemento.

Dados
: Os valores armazenados nesse elemento.

Rotação
: Orientação do elemento no circuito.

Formato para número
: Formato a ser usado para mostrar números.

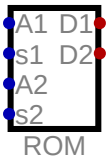
dígitos de ponto fixo
: Número de dígitos binários na parte fracionária

Memória de programa
: Tomar o conteúdo da ROM como a memória de programa. Dessa forma, poderá ser acessado por uma IDE externa.

Carga inicial do modelo
: Recarregar o arquivo hexadecimal toda vez que o modelo for ligado.

Arquivo
: Arquivo a ser carregado em ROM.

Usar Big-Endian ao importar.
: Usar byte em representação Big-Endian ao importar.



11.3. ROM com porta dupla

Componente de memória não volátil. Os dados armazenados poderão ser editados na caixa de diálogo para atributos.

Entradas

A1
: Esse pino definirá o endereço da palavra de dados à saída D1.

s1
: Se a entrada estiver em nível alto, a saída D1 estará ativada. Se estiver em nível baixo, a saída estará em alta impedância (Z).

A2
: Esse pino definirá o endereço da palavrra de dados à saída D2.

s2
: Se a entrada estiver em nível alto, a saída D2 estará ativada. Se estiver em nível baixo, a saída estará em alta impedância (Z).

Saídas

D1
: Palavra de dados selecionada se a entrada s1 estiver em nível alto.

D2
: Palavra de dados selecionada se a entrada s2 estiver em nível alto.

Atributos

Bits de dados
: Número de bits de dados usados.

Bits de endereço
: Número de bits de endereço usado.

Rótulo
: Nome do elemento.

Dados
: Os valores armazenados nesse elemento.

Rotação
: Orientação do elemento no circuito.

Formato para número
: Formato a ser usado para mostrar números.

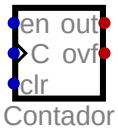
dígitos de ponto fixo
: Número de dígitos binários na parte fracionária

Memória de programa
: Tomar o conteúdo da ROM como a memória de programa. Dessa forma, poderá ser acessado por uma IDE externa.

Carga inicial do modelo
: Recarregar o arquivo hexadecimal toda vez que o modelo for ligado.

Arquivo
: Arquivo a ser carregado em ROM.

Usar Big-Endian ao importar.
: Usar byte em representação Big-Endian ao importar.



11.4. Contador

Componente que funcionará como contador simples. Dependendo da entrada de clock, será incrementado. Poderá retornar a zeor mediante udo da entrada clr. A largura de bits poderá ser determinada por uma caixa de diálogo de atributos. Exportável para VHDL/Verilog.

Entradas

en
: Se estiver em 1, o contador estará habilitado!

C
: Entrada de clock. Quando da borda de subida, incrementará o contador.

clr
: Limpeza síncrona do contador se igual a 1.

Saídas

out
: Retornará o valor contado.

ovf
: Transbordamento de saída (overflow). Esse pino será igual a 1 se o contador exceder seu valor máximo e a entrada en estiver em 1.

Atributos

Bits de dados
: Número de bits de dados usados.

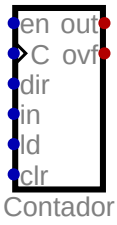
Entradas invertidas
: É possível selecionar entradas cujos valores serão invertidos.

Rótulo
: Nome do elemento.

Rotação
: Orientação do elemento no circuito.

Usar como valor medido
: Se definido, o valor será usado como o de medição, e será mostrado no gráfico e também na tabela de dados. Além disso, um rótulo deverá ser especificado para servir como identificação do valor.

Contador de programa
: Tomar o registrador como contador de programa. O valor do registrador será tomado a partir da IDE do montador externo para marcar a linha de código corrente durante a depuração.



11.5. Contador com preset

Contador com valor que poderá ser pré-estabelecido. Além disso, o valor máximo e a direção de contagem também poderão ser especificados. Exportável para VHDL/Verilog.

Entradas

en
: Se estiver em 1, o contador estará habilitado!

C
: A entrada de clock. Quando da borda de subida, incrementará o contador.

dir
: Especificará a direção de contagem. Se for 0 será para cima.

in
: Palavra de dados a ser armazenada no contador quando (ld) for habilitado.

ld
: Se habilitado, o valor na entrada in será armazenado no contador no próximo sinal do clock.

clr
: Limpeza síncrona do contador se igual a 1.

Saídas

out
: Retornará o valor contado.

ovf
: Saída de transbordamento (overflow output). Se estiver em 1, e a entrada en estiver em 1 e se o contador alcançar seu valor máximo quando estiver sendo incrementado, ou quando atingir 0 ao ser decrementado.

Atributos

Bits de dados
: Número de bits de dados usados.

Valor máximo
: Se o valor zero for fornecido, o maior valor possível será usado (todos os bits em 1).

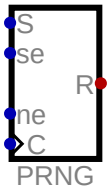
Entradas invertidas
: É possível selecionar entradas cujos valores serão invertidos.

Rótulo
: Nome do elemento.

Rotação
: Orientação do elemento no circuito.

Usar como valor medido
: Se definido, o valor será usado como o de medição, e será mostrado no gráfico e também na tabela de dados. Além disso, um rótulo deverá ser especificado para servir como identificação do valor.

Contador de programa
: Tomar o registrador como contador de programa. O valor do registrador será tomado a partir da IDE do montador externo para marcar a linha de código corrente durante a depuração.



11.6. Gerador de número aleatório

Poderá ser usado para gerar números aleatórios. Quando a simulação for iniciada, o gerador será reiniciado de modo que uma nova sequência de números pseudo-aleatórios seja gerada a cada vez. O gerador poderár ser iniciado durante simulação com um valor definido para semente para que se possa gerar uma sequência definida de valores pseudo-aleatórios.

Entradas

S
: Novo valor para a semente do gerador.

se
: Se definido, o gerador de número aleatório será reiniciado com uma nova semente na subida do clock.

ne
: Se definido, um novo número aleatório estará disponível na próxima borda de subida do clock.

C
: Entrada do clock.

Saídas

R
: Saída do número pseudo-aleatório.

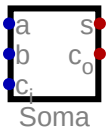
Atributos

Bits de dados
: Número de bits de dados usados.

Rótulo
: Nome do elemento.

Rotação
: Orientação do elemento no circuito.

12. Aritmética



12.1. Somar

Componente para realizar uma soma simples. Somará dois valores inteiros (a+b) presentes nas entradas (a) e (b). O resultado será incrementado em uma unidade se a entrada de carry estiver em nível alto. Exportável para VHDL/Verilog.

Entradas

a
: Primeira entrada para somar.

b
: Segunda entrada para somar.

c_i
: Entrada para carry-in, se estiver em nível alto, irá incrementar uma unidade.

Saídas

s
: O resultado da soma

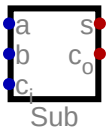
c_o
: Saída carry-out. Estará em nível ato se houver transbordamento (overflow).

Atributos

Rótulo
: Nome do elemento.

Bits de dados
: Número de bits de dados usados.

Rotação
: Orientação do elemento no circuito.



12.2. Subtrair

Componente para realizar uma subtração simples. Subtrairá dois valores inteiros (a-b) presentes nas entradas (a) e (b). O resultado será decrementado em uma unidade se a entrada de carry estiver em nível alto. Exportável para VHDL/Verilog.

Entradas

a
: Primeira entrada para subtrair.

b
: Segunda entrada para subtrair.

c_i
: Entrada para carry-in, se estiver em nível alto, irá decrementar de uma unidade.

Saídas

s
: O resultado da subtração.

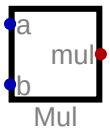
c_o
: Saída carry-out. Estará em nível alto se houver transbordamento (overflow).

Atributos

Rótulo
: Nome do elemento.

Bits de dados
: Número de bits de dados usados.

Rotação
: Orientação do elemento no circuito.



12.3. Multiplicar

Componente para realizar uma multiplicação. Multiplicará dois valores inteiros (a*b) presentes nas entradas (a) e (b). Exportável para VHDL/Verilog.

Entradas

a
: Primeira entrada para multiplicar.

b
: Segunda entrada para multiplicar.

Saídas

mul
: O resultado da multiplicação.

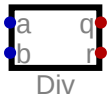
Atributos

Rótulo
: Nome do elemento.

Operação com sinal
: Se selecionado, a operação será executada usando valores com sinal (em complemento de 2).

Bits de dados
: Número de bits de dados usados.

Rotação
: Orientação do elemento no circuito.



12.4. Dividir

Componente para realizar uma divisão. Dividirá um valor inteiro por outro (a/b), ambos presentes nas entradas (a) e (b). Se o divisor for zero, o valor igual a 1 será usado no lugar. Em se tratando de divisão com sinal, o resto será sempre positivo.

Entradas

a
: dividendo

b
: divisor

Saídas

q
: quociente

r
: resto

Atributos

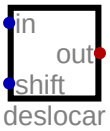
Rótulo
: Nome do elemento.

Bits de dados
: Número de bits de dados usados.

Operação com sinal
: Se selecionado, a operação será executada usando valores com sinal (em complemento de 2).

Resto de divisão sempre positivo
: Se definido, o resto de uma divisão com sinal será sempre positivo.

Rotação
: Orientação do elemento no circuito.



12.5. Registrador de deslocamento

Componente para deslocamento de bits. Deslocará o valor na entrada por certo número de bits presente à entrada.

Entradas

in
: Entrada com os bits a serem deslocados.

shift
: Entrada com a largura a ser deslocada.

Saídas

out
: Saída com o valor deslocado.

Atributos

Rótulo
: Nome do elemento.

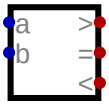
Bits de dados
: Número de bits de dados usados.

Entrada para o deslocamento com sinal
: Entrada para o deslocamento em complemento de 2

Direção
: Definir direção.

Modo
: Modo do registrador de deslocamento

Rotação
: Orientação do elemento no circuito.



12.6. Comparador

Componente para comparar valores bit a bit. Irá comparar dois valores (a) e (b) presentes às entradas e compor as saídas correspondentes. Exportável para VHDL/Verilog.

Entradas

a
: Entrada (a) para comparar.

b
: Entrada (b) para comparar.

Saídas

>
: Saída com o valor igual a 1, se a entrada (a) for maior que (b)

=
: Saída com o valor 1, se (a) igual a (b)

<
: Saída com o valor igual a 1, se a entrada (a) for menor que (b)

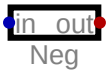
Atributos

Rótulo
: Nome do elemento.

Bits de dados
: Número de bits de dados usados.

Operação com sinal
: Se selecionado, a operação será executada usando valores com sinal (em complemento de 2).

Rotação
: Orientação do elemento no circuito.



Neg

12.7. Negação

Complemento de 2 Exportável para VHDL/Verilog.

Entradas

in
: Entrada da palavra de dados a qual será aplicada o complemento de 2

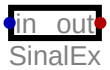
Saídas

out
: Retornará o resultado da negação em complemento de 2.

Atributos

Bits de dados
: Número de bits de dados usados.

Rotação
: Orientação do elemento no circuito.



SinalEx

12.8. Extensor de sinal

Incrementará a larguram em bits de um valor com sinal mantendo seu valor. Se a entrada for de um único bit, esse valor será disponível em todos os bits da saída. Exportável para VHDL/Verilog.

Entradas

in
: Valor da entrada. A largura em bits da entrada que deverá ser menor que a largura em bits da saída!

Saídas

out
: Valor da entrada estendida. A largura em bits da entrada que deverá ser menor que a largura em bits da saída!

Atributos

Rótulo
: Nome do elemento.

Largura em bits da entrada
: O número de bits da saída deverá ser maior do que o número de bits da entrada.

Largura em bits da saída
: O número de bits da saída deverá ser maior queo o número de bits da entrada.

Rotação
: Orientação do elemento no circuito.

12.9. Contador de bits

Retornará o número de bits iguais a 1 presentes no valor à entrada.

Entradas

in
: Entrada para se contar os valores iguais a 1.

Saídas

out
: Saída com a quantidade de valores iguais a 1.

Atributos

Bits de dados
: Número de bits de dados usados.

Rotação
: Orientação do elemento no circuito.

13. Chaves

13.1. Chave

Interruptor simples. Não haverá atraso. Uma mudança de sinal irá propagar imediatamente.

Saídas

A1
: Um das conexões da chave.

B1
: Um das conexões da chave.

Atributos

Bits de dados
: Número de bits de dados usados.

Rótulo
: Nome do elemento.

Número de polos
: Número de polos disponíveis.

Fechada
: Estado inicial definido para a chave.

Rotação
: Orientação do elemento no circuito.

Espelhamento
: Espelhar o componente em um circuito.

Chave com comportamento semelhante a uma entrada
: Se o modelo for analisado, a chave se comportará como uma entrada, onde "aberto" corresponderá a '0' e "fechado" a '1'

13.2. Chave de dois polos

Chave de dois polos. Não haverá atraso. Uma mudança de sinal irá propagar imediatamente.

Saídas

A1
: Um das conexões da chave.

B1
: Um das conexões da chave.

C1
: Um das conexões da chave.

Atributos

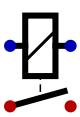
Bits de dados
: Número de bits de dados usados.

Rótulo
: Nome do elemento.

Número de polos
: Número de polos disponíveis.

Rotação
: Orientação do elemento no circuito.

Espelhamento
: Espelhar o componente em um circuito.



13.3. Relé

Relé é uma chave que pode ser controlada por uma bobina. Se uma corrente fluir pela bobina, a chave fechará ou abrirá. Não haverá diodo de flyback de modo que an direção da corrente não será relevante. A chave atuará se as entradas tiverem valores diferentes. O relé se comportará de forma semelhante à porta XOR.

Entradas

in1
: Uma das entradas para controle do relé.

in2
: Uma das entradas para controle do relé.

Saídas

A1
: Um das conexões da chave.

B1
: Um das conexões da chave.

Atributos

Bits de dados
: Número de bits de dados usados.

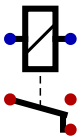
Rótulo
: Nome do elemento.

Número de polos
: Número de polos disponíveis.

Relé normalmente fechado.
: Se o relé estiver fechado, a entrada estará em nível baixo.

Rotação
: Orientação do elemento no circuito.

Espelhamento
: Espelhar o componente em um circuito.



13.4. Relé de dois polos

Relé é uma chave que pode ser controlada por uma bobina. Se uma corrente fluir pela bobina, a chave fechará ou abrirá. Não haverá diodo de flyback de modo que an direção da corrente não será relevante. A chave atuará se as entradas tiverem valores diferentes. O relé terá comportamento similar à porta XOR.

Entradas

in1
: Uma das entradas para controle do relé.

in2
: Uma das entradas para controle do relé.

Saídas

A1
: Um das conexões da chave.

B1
: Um das conexões da chave.

C1
: Um das conexões da chave.

Atributos

Bits de dados
: Número de bits de dados usados.

Rótulo
: Nome do elemento.

Número de polos
: Número de polos disponíveis.

Rotação
: Orientação do elemento no circuito.

Espelhamento
: Espelhar o componente em um circuito.

13.5. FET tipo P

Transistor de efeito de campo tipo P. A base será conectada à referência de tensão positiva e o transistor será simulado sem um diodo interno.

Entradas

G
: Porta

Saídas

S
: Fonte

D
: Dreno

Atributos

Bits de dados
: Número de bits de dados usados.

Unidirecional
: Transistores unidirectionais propagarão um sinal somente da fonte para o dreno. Eles são muito mais rápidos para simular do que os transistores bidirecionais. Desde que não haja realimentação do dreno para a fonte, nesse modo, o transistor não poderá causar curto-circuito de fios quando estiver conduzindo. Assim, será possível simular certos circuitos CMOS.

Rótulo
: Nome do elemento.

Rotação
: Orientação do elemento no circuito.

Espelhamento
: Espelhar o componente em um circuito.

13.6. FET tipol N

Transistor de efeito de campo tipo N. A base será conectada à terra e o transistor será simulado sem um diodo interno.

Entradas

G
: Porta

Saídas

D
: Dreno

S
: Fonte

Atributos

Bits de dados
: Número de bits de dados usados.

Unidirecional
: Transistores unidirectionais propagarão um sinal somente da fonte para o dreno. Eles são muito mais rápidos para simular do que os transistores bidirecionais. Desde que não haja realimentação do dreno para a fonte, nesse modo, o transistor não poderá causar curto-circuito de fios quando estiver conduzindo. Assim, será possível simular certos circuitos CMOS.

Rótulo
: Nome do elemento.

Rotação
: Orientação do elemento no circuito.

Espelhamento
: Espelhar o componente em um circuito.

13.7. Fusível

Fusível usado para construir uma memória programável uma única vez.

Saídas

out1
: Um das conexões da chave.

out2
: Um das conexões da chave.

Atributos

Programado
: Se o diodo estiver "programado" ou não ("blown"). Em uma porta flutuante FET, estará com carga. Poderá ser definido apenas pela letra 'p'.

Rotação
: Orientação do elemento no circuito.

13.8. Diodo para VDD

Simples diodo unidirecional, usado para levar uma conexão para VDD. É usada para implementar uma OR de fios. Para isso é necessário conectar um resistor de pull down à saída do diodo. Na simulação o diodo se comportará como uma porta ativa com uma tabela-verdade trivalente: se a entrada estiver em nível alto, a saída também estará. Em todos os outros casos (entrada em nível baixo ou em alta impedância) a saída estará estado de alta impedância. De modo que dois diodos anti-paralelos poderão manter-se em nível alto, o que não seria possível com diodos reais. Esse é um diodo ideal: não haverá queda de tensão através de um diodo polarizado diretamente.

Entradas

in
: Se a entrada estiver em nível alto, a saída também estará. Em todos os outros casos, a saída estará estado de alta impedância.

Saídas

out
: Se a entrada estiver em nível alto, a saída também estará. Em todos os outros casos, a saída estará estado de alta impedância.

Atributos

Programado
: Se o diodo estiver "programado" ou não ("blown"). Em uma porta flutuante FET, estará com carga. Poderá ser definido apenas pela letra 'p'.

Rotação
: Orientação do elemento no circuito.

13.9. Diode para Terra

Simples diodo unidirecional, usado para levar uma conexão para VDD. É usada para implementar uma AND de fios. Para isso é necessário conectar um resistor de pull up à saída do diodo. Se a entrada estiver em nível baixo, a saída também estará. Em todos os outros casos (entrada em nível baixo ou em alta impedância) a saída estará estado de alta impedância. De modo que dois diodos anti-paralelos poderão manter-se em nível baixo, o que não seria possível com diodos reais. Esse é um diodo ideal: não haverá queda de tensão através de um diodo polarizado diretamente.

Entradas

in
: Se a entrada estiver em nível baixo, a saída também estará. Em todos os outros casos, a saída estará estado de alta impedância.

Saídas

out
: Se a entrada estiver em nível baixo, a saída também estará. Em todos os outros casos, a saída estará estado de alta impedância.

Atributos

Programado
: Se o diodo estiver "programado" ou não ("blown"). Em uma porta flutuante FET, estará com carga. Poderá ser definido apenas pela letra 'p'.

Rotação
: Orientação do elemento no circuito.

13.10. FET tipo P com porta flutuante

Transistor de efeito de campo tipo P com porta flutuante. A base será conectada à terra e o transistor será simulado sem um diodo interno. Se houver uma carga armazenada na porta flutuante, o transistor não conduzirá, mesmo se a porta estiver em nível baixo.

Entradas

G
: Porta

Saídas

S
: Fonte

D
: Dreino

Atributos

Bits de dados
: Número de bits de dados usados.

Rótulo
: Nome do elemento.

Programado
: Se o diodo estiver "programado" ou não ("blown"). Em uma porta flutuante FET, estará com carga. Poderá ser definido apenas pela letra 'p'.

Rotação
: Orientação do elemento no circuito.

Espelhamento
: Espelhar o componente em um circuito.

13.11. FET tipo N com porta flutuante

Transistor de efeito de campo tipo N com porta flutuante. A base será conectada à terra e o transistor será simulado sem um diodo interno. Se houver uma carga armazenada na porta flutuante, o transistor não estará conduzindo ainda que a porta esteja em nível alto.

Entradas

G
: Porta

Saídas

D
: Dreno

S
: Fonte

Atributos

Bits de dados
: Número de bits de dados usados.

Rótulo
: Nome do elemento.

Programado
: Se o diodo estiver "programado" ou não ("blown"). Em uma porta flutuante FET, estará com carga. Poderá ser definido apenas pela letra 'p'.

Rotação
: Orientação do elemento no circuito.

Espelhamento
: Espelhar o componente em um circuito.

13.12. Porta de transmissão

Uma porta de transmissão real é construída com apenas dois transistores. Por isso é frequentemente usada para se economizar transistores quando da implementação em silício.

Entradas

S
: entrada de controle

¬S
: entrada de controle invertida

Saídas

A
: entrada A

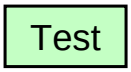
B
: entrada B

Atributos

Bits de dados
: Número de bits de dados usados.

Rotação
: Orientação do elemento no circuito.

14. Diversos



14.1. Caso de teste

Descrever um caso de teste. Em um teste de caso será possível descrever o comportamento de um circuito. Isso poderá ser automaticamente verificado, ou seja, se o comportamento realmente corresponde à descrição, ou não, e assim sendo, uma mensagem de erro será mostrada. Exportável para VHDL/Verilog.

Atributos

Rótulo
: Nome do elemento.

Dados para teste
: Descrição de um caso de teste. Detalhes de sintaxe poderão ser encontrados na caixa de diálogo para ajuda no editor de caso de teste.

Habilitado
: Habilitar ou desabilitar o componente.

15. Diversos - Decoração

Texto

15.1. Texto

Mostrar um texto no circuito. Não afeta a simulação. O texto poderá ser alterado pela caixa do diálogo do atributo.

Atributos

Descrição
: Uma breve descrição do elemento e como usá-lo.

Tamanho da fonte
: Definição do tamanho de fonte a ser usado nesse texto.

Rotação
: Orientação do elemento no circuito.

Orientação
: Posição de coordenada relativa ao texto.

Aderência à grade
: Se selecionado, o componente se alinhará à grade.

Texto

15.2. Retângulo

Mostrarã um retângulo no circuito. Isso não afetará a simulação. Se um sinal negativo (-) for usado no cabeçalho, esse será omitido.

Atributos

Rótulo
: Nome do elemento.

Largura
: Largura em unidades da grade

Altura
: Altura em unidades da grade

Tamanho da fonte
: Definição do tamanho de fonte a ser usado nesse texto.

Texto interno
: Colocar texto interno ao retângulo.

Texto embaixo
: Colocar texto embaixo do retângulo.

Texto à direita
: Colocar texto à direita do retângulo.

Aderência à grade
: Se selecionado, o componente se alinhará à grade.

16. Diversos - Genérico

16.1. Inicialização genérica

Código a ser executado ao iniciar-se um circuito genérico diretamente. Se um circuito genérico tiver que ser iniciado diretamente, tal componente deverá estar presente. Exportável para VHDL/Verilog.

Atributos

Rótulo
: Nome do elemento.

Habilitado
: Habilitar ou desabilitar o componente.

Parametrização genérica
: Comandos para generalizar um circuito.

Código

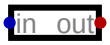
16.2. Código

Código a ser executado quando um circuito genérico for incorporado. Poderá ser usado, por exemplo, para adicionar componentes ou conexões a um circuito. Exportável para VHDL/ Verilog.

Atributos

Parametrização genérica
: Comandos para generalizar um circuito.

17. Diversos - VHDL/Verilog



Externo

17.1. Externo

Componente que executará um processo externo para calcular uma função lógica. Será usado para especificar o comportamento de um componente mediante VHDL ou Verilog. Até o momento apenas os simuladores ghdl (VHDL) e Icarus Verilog têm suporte. A identificação do componente deverá coincidir com o nome da entidade ou módulo! Exportável para VHDL/ Verilog.

Entradas

in

Saídas

out

Atributos

Rótulo
: Nome do elemento.

Largura
: Largura do símbolo a ser usada se o circuito for componente de outro.

Entradas
: Entradas do processo externo. Lista de nomes de sinais separados por vírgulas. Após cada nome de sinal, separado por dois-pontos, um número de bits pode ser especificado. As entradas de 8 bits de um somador poderão ser descritas como "a:8,b:8,c_in".

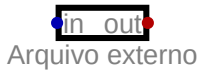
Saídas
: As saídas do processo externo. Lista de nomes de sinais separados por vírgulas. Após cada nome de sinal, separado por dois-pontos, um número de bits pode ser especificado. As saídas de 8 bits de um somador poderão ser descritas como "a:8,b:8,c_in". "s:8,c_out".

Código de programa
: Definição do código de programa a ser executado na aplicação externa.

Aplicação
: Definição de qual aplicação usar.

Opções GHDL
: Opções que serão usadas para todos os passos de processamento pelo GHDL.

Iverilog - Opções
: Opções que serão usadas para todos os passos de processamento pelo Iverilog.



17.2. Arquivo externo

Componente que executará um processo externo para calcular uma função lógica. Será usado para especificar o comportamento de um componente mediante VHDL ou Verilog. A simulação real do comportamento deverá ser feita por meio de um simulador externo. A identificação do componente deverá coincidir com o nome da entidade ou módulo! Exportável para VHDL/Verilog.

Entradas

: in

Saídas

: out

Atributos

Rótulo
: Nome do elemento.

Largura
: Largura do símbolo a ser usada se o circuito for componente de outro.

Entradas
: Entradas do processo externo. Lista de nomes de sinais separados por vírgulas. Após cada nome de sinal, separado por dois-pontos, um número de bits pode ser especificado. As entradas de 8 bits de um somador poderão ser descritas como "a:8,b:8,c_in".

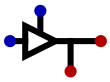
Saídas
: As saídas do processo externo. Lista de nomes de sinais separados por vírgulas. Após cada nome de sinal, separado por dois-pontos, um número de bits pode ser especificado. As saídas de 8 bits de um somador poderão ser descritas como "a:8,b:8,c_in". "s:8,c_out".

Código de programa
: O arquivo contendo o código de programa a ser executado pela aplicação externa.

Aplicação
: Definição de qual aplicação usar.

Opções GHDL
: Opções que serão usadas para todos os passos de processamento pelo GHDL.

Iverilog - Opções
: Opções que serão usadas para todos os passos de processamento pelo Iverilog.



17.3. Controle do pino

Lógica de controle para um pino bidirecional. Esse componente será necessário apenas no contexto para a geração de código VHDL ou Verilog, a fim de se criar uma porta bidirecional em HDL. Se não desejar usar uma porta de entrada e saída bidirecional em FPGA, não o use! O componente não poderá ser usado em circuito incorporado. Apenas será permitido no circuito de nível mais alto! Exportável para VHDL/Verilog.

Entradas

wr
: O dado a ser emitido.

oe
: Ativar a saída.

Saídas

rd
: Os dados a serem lidos.

pin
: O conector para o pino corrente. Apenas um sinal de saída deverá estar conectado aqui.

Atributos

Bits de dados
: Número de bits de dados usados.

Rotação
: Orientação do elemento no circuito.

Espelhamento
: Espelhar o componente em um circuito.

18. Diversos

18.1. Fonte

Não possui qualquer função. Garante que a alimentação (VDD) e o terra (GND) estarão conectados. Poderá ser usada em circuitos 74xx para ligar os pinos de alimentação, e assim testar se corretamente conectados.

Entradas

VDD
: Deverá ser conectada à fonte de alimentação (VDD)!

GND
: Deverá ser conectado ao terra (GND)!

Atributos

Rótulo
: Nome do elemento.

Rotação
: Orientação do elemento no circuito.

18.2. Distribuidor bidirecional

Poderá ser usado para barramento de dados e simplificará especialmente a construção de módulos de memória em um pacote DIL, assim com a implementação desse tipo de barramento.

Entradas

OE
: Quando definido, o valor no terminal de dados comum D será a saída para os bits D[i], se não, os bits D[i] terão como saída o valor D comum.

Saídas

D
: A conexão de dados comum.

D0
: O primeiro bit de dado 0 do barramento do distribuidor.

Atributos

Bits de dados
: Número de bits de dados usados.

Rotação
: Orientação do elemento no circuito.

Espelhamento
: Espelhar o componente em um circuito.

Propagação
: Configuração de como irão se propagar as entradas e saídas no circuito.

18.3. Reiniciar

A saída desse componente será mantida em nível alto durante a inicialização do circuito. Após o circutio se estabilizar, a saída irá para o nível baixo. Se a saída for invertida, irá se comportar de forma oposta. Exportável para VHDL/Verilog.

Saídas

Reset
: Saída de reinicialização.

Atributos

Rótulo
: Nome do elemento.

Saída invertida
: Se definido, inverterá a saída.

Rotação
: Orientação do elemento no circuito.

18.4. Pausa

Se esse componente for usado em um circuito, o botão "Run To Break" entre "Start" e "Stop" estará habilitado. Esse botão irá o clock para o circuito até uma borda de subida na entrada do componente ser detectada. Esse elemento poderá ser usado para depuração ao prover o clock ao circuito em qualquer ponto de parada (breakpoint). Poderá ser usado também para implementar uma instrução BRK em linguagem de montagem (assembly). Será possível simular um processador e interromper sua execução quando a instrução BRK for alcançada. O clock de avanço rápido somente poderá ser usado se o clock de tempo real estiver desabilitado.

Entradas

brk
: Irá suspender uma simulação rapidamente se uma borda de subida for detectada.

Atributos

Rótulo
: Nome do elemento.

Habilitado
: Habilitar ou desabilitar o componente.

Ciclos de temporização
: Se a quantidade de ciclos for alcançada sem um sinal de interrupção, uma situação de erro se estabelecerá.

Rotação
: Orientação do elemento no circuito.

Parar

18.5. Parar

Uma borda de subida na entrada irá parar a simulação. Terá o mesmo efeito que pressionar o botão Parar na barra de ferramentas.

Entradas

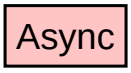
stop
: Um borda de subida irá parar a simulação.

Atributos

Rótulo
: Nome do elemento.

Entradas invertidas
: É possível selecionar entradas cujos valores serão invertidos.

Rotação
: Orientação do elemento no circuito.



18.6. Temporização assíncrona

Será permitido a temporização de um circuito sequencial assíncrono, tal como um pipeline de Muller. O circuito deverá ser iniciado em modo de passo único e deverá alcançar um estado estável logo ao iniciar-se. O circuito poderá ser então iniciado interativamente ou mediante uma porta de reinicialização. Não é permitido usar um componente de clock comum nesse modo.

Atributos

Iniciar o clock de tempo real
: Se habilitado o clock da execução será iniciado juntamente com a ligação do circuito

Frequência/Hz
: Frequência em tempo real a ser usada no clock de tempo real

E Biblioteca

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 with strobe
7427: triple 3-input NOR gate
7428: quad 2-input NOR buffer
7430: 8-input NAND gate
7432: quad 2-input OR gate
7440: dual 4-input NAND buffer
7442: 4-line BCD to 10-line decimal decoder
7447: BCD to 7-segment decoder, active low
7448: BCD to 7-segment decoder, active high
7451: 2-input/3-input AND-NOR gate
7454: 2-3-2-3-line AND NOR gate
7455: 2 wide 4-input AND-NOR gate
7458: dual AND OR gate
7474: dual D-flip-flop
7476: dual J-K flip-flops with preset and clear
7480: Gated Full Adder with Complementary Inputs and Complementary Sum Outputs
7482: 2-bit binary full adder

7483: 4-bit binary full adder
7483Real: 4-bit binary full adder, real gates
7485: 4-bit comparator
7486: quad 2-input XOR gate
7489: 64-bit RAM
7490: asynchronous two - five - decimal addition counter
7493: 4-bit Binary Counter. Connect QA to CKB and clock on CKA for full 4-bit counter. Connect QB to R1 and QD to R2 for a BCD counter.
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
74125: Quadruple bus buffer gates with 3-state outputs (active low output enable)
74126: Quadruple bus buffer gates with 3-state outputs (active high output enable)
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
74175: quad D-flip-flop
74181: 4-bit arithmetic logic unit
74182: look-ahead carry generator
74189: 64-Bit Random Access Memory with 3-STATE Outputs
74190: Presettable synchronous 4-bit bcd up/down counter
74191: Presettable synchronous 4-bit binary up/down counter
74193: Synchronous 4-Bit Up/Down Binary Counter with Dual Clock
74194: 4-Bit Bidirectional Universal Shift Register
74194real: 4-Bit Bidirectional Universal Shift Register, Databook implementation.
74198: 8-bit shift register
74238: 3-line to 8-line decoder/demultiplexer
74244: octal 3-state buffer/line driver/line receiver
74245: octal bus transceivers with 3-state outputs
74247: BCD to 7-segment decoder, active low, tails on 6 and 9
74248: BCD to 7-segment decoder, active high, tails on 6 and 9
74253: dual tri state 4-line to 1-line data selectors/multiplexers

74257: quad 2-line to 1-line data selectors/multiplexers (3-state output)
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
74299: 8-Input Universal Shift/Storage Register with Common Parallel I/O Pins
74373: octal transparent latches
74374: octal positive-edge-triggered flip-flops
74377: Octal D Flip-Flop with enable
74382: 4-Bit Arithmetic Logic Unit
74540: octal buffer/line driver, inverted
74541: octal buffer/line driver
74573: octal transparent latches, different pinout compared to 74373
74574: octal positive-edge-triggered flip-flops, different pinout compared to 74374
74590: 8-bit binary counter with tri-state output registers
74595: 8-Bit Shift Registers with 3-State Output Registers
74670: 3-state 4-by-4 Register File
74682: 8-bit digital comparator
74688: 8-bit identity comparator
74779: 8-Bit Bidirectional Binary Counter with 3-STATE Outputs
74804: hex 2-input NAND gate https://www.ti.com/lit/ds/symlink/sn74as804b.pdf
74805: hex 2-input NOR gate http://www.ti.com/lit/ds/symlink/sn54as805b.pdf
74808: hex 2-input AND gate http://www.ti.com/lit/ds/symlink/sn54as808b.pdf
74832: hex 2-input OR gate http://www.ti.com/lit/ds/symlink/sn54as832b.pdf
744002: dual 4-input NOR gate
744017: Johnson decade counter with 10 decoded outputs
744075: triple 3-input OR gate
7440105: 4-Bit x 16-Word FIFO Register
A623308A: 8K X 8 BIT CMOS SRAM
RAM32Bit: Ein 32-Bit-Speicher, der Bytezugriff ermöglicht und auch mit nicht ausgerichteten Speicheradressen umgehen kann.