

# Laboratório de Introdução à Arquitetura de Computadores

IST – LEIC

2021/2022

## Introdução ao simulador

### Guião 1

9 a 13 de maio de 2022

(Semana 1)

#### 1 – Objetivos

Com este trabalho pretende-se que os alunos se familiarizem com a interface do simulador, por meio de um conjunto de circuitos exemplificativos.

#### 2 – O simulador

A disciplina Introdução à Arquitetura de Computadores utiliza um simulador que permite testar o funcionamento de todos os sistemas lecionados no seu âmbito, permitindo fazer experiências sem o ónus da montagem física dos circuitos e em qualquer lugar, não apenas durante a aula de laboratório.

O simulador pode ser executado em qualquer plataforma para a qual exista uma Java Virtual Machine (JVM), versão 11, 64 bits, com Windows, macOS ou Linux. Pode ser obtido para instalação no seu computador no *site* da disciplina, no Fenix. Basta executar o ficheiro descarregado (*simulador-data.jar*), que contém a data em que foi feito.

Antes, deverá instalar a máquina virtual de Java (**versão 11, 64 bits**). Pode usar qualquer distribuição de OpenJDK 11. Recomendamos a Temurin, que pode obter em <https://adoptium.net/installation/> para cada um dos sistemas operativos, seja usando um package manager, seja usando um dos installers (estão lá as instruções detalhadas).

Não se esqueça de selecionar a versão 11 e o JDK (o JRE não chega).

Depois do Java de 64 bits instalado, basta correr o ficheiro do simulador, como qualquer executável.

NOTA – Nem sempre a instalação corre bem, em particular quando já há outras versões de Java instaladas. O problema mais frequente é a associação entre o clique no ficheiro do simulador (um “.jar”, ou Java archive) e o programa que o executa (a máquina virtual do Java). Se o simulador não abrir a sua janela de edição, abra uma janela de comandos de linha e execute o seguinte comando: `java -jar simulador-data.jar`

Se o simulador abrir, então está tudo bem com a instalação do Java, mas tem de selecionar a plataforma Java de 64 bits por omissão para abrir todos os ficheiros “.jar”.

#### 3 – Novo simulador

O simulador tem estado a ser reescrito, com funcionalidades e interface melhoradas. Trata-se de um grande investimento numa ferramenta que é bastante complexa, e não está ainda

completa. Apesar de todo o cuidado empregue nos testes, é natural que surjam alguns erros. É provável que ao longo do semestre seja disponibilizada mais do que uma versão do simulador, à medida que os erros vão sendo corrigidos. Por essa razão, o nome do ficheiro do simulador inclui o dia em que foi disponibilizado.

A equipa docente agradece desde já a compreensão de todos. Eventuais erros detetados devem ser reportados para [jose.delgado@tecnico.ulisboa.pt](mailto:jose.delgado@tecnico.ulisboa.pt)

## 4 – Execução do trabalho de laboratório

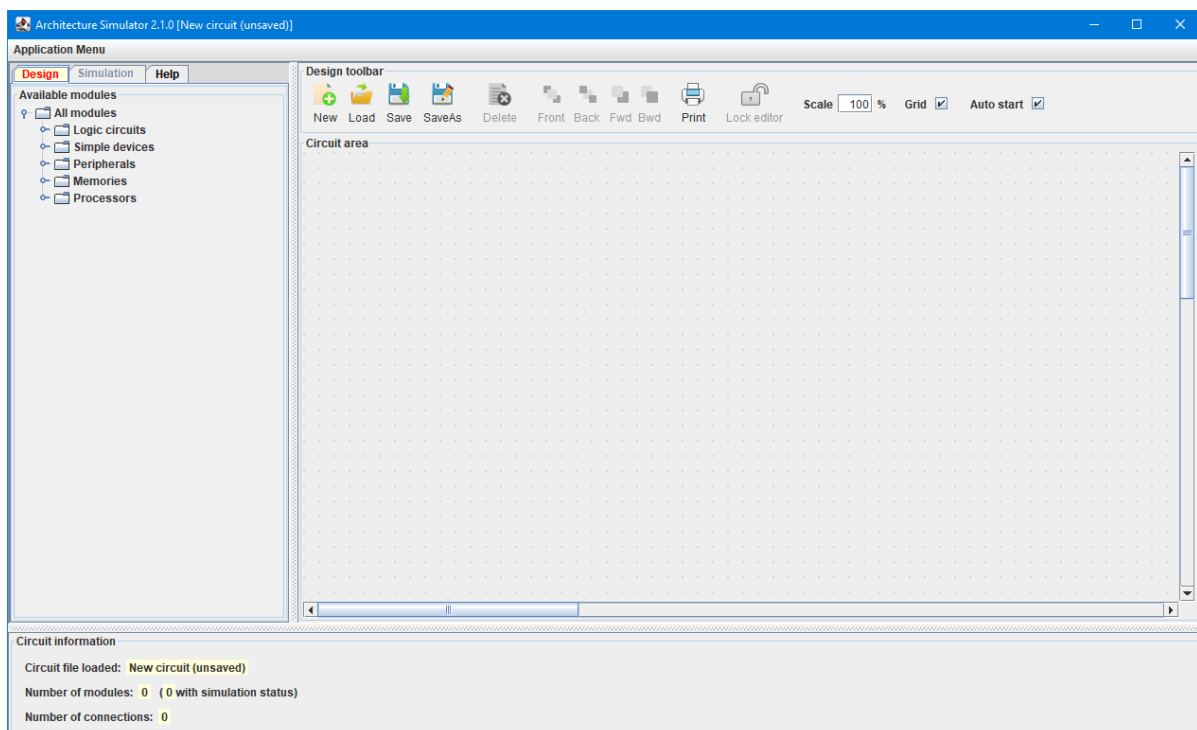
As secções seguintes ilustram alguns dos circuitos que pode montar e algumas das experiências que pode fazer com eles. Não se sinta limitado ao enunciado. Use a criatividade e experimente outras coisas (quer novos circuitos, quer novas experiências), mesmo depois do laboratório, em casa. A fazer é que se aprende!

## 5 – Um circuito simples sem portas lógicas

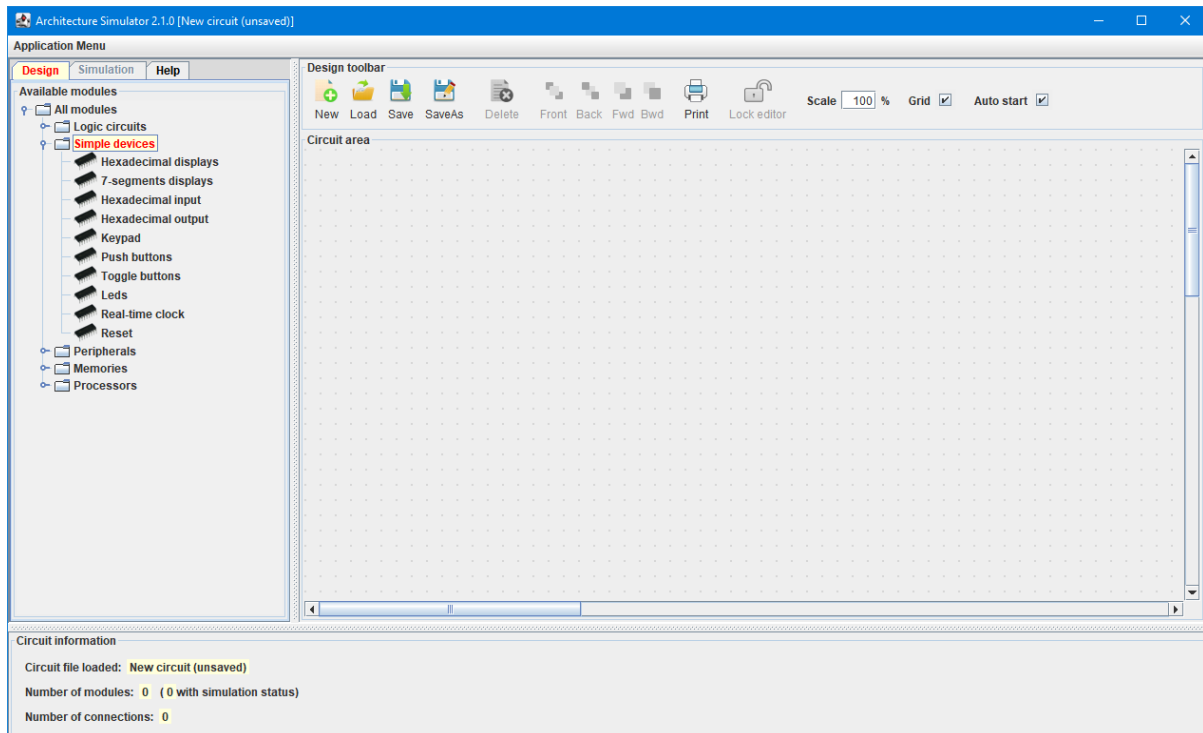
### 5.1 – Criação do circuito

Execute o simulador, fazendo clique duplo sobre o ficheiro “simulador-data.jar”. Deverá aparecer, após o tempo necessário para iniciar a máquina virtual Java, um ecrã como o seguinte.

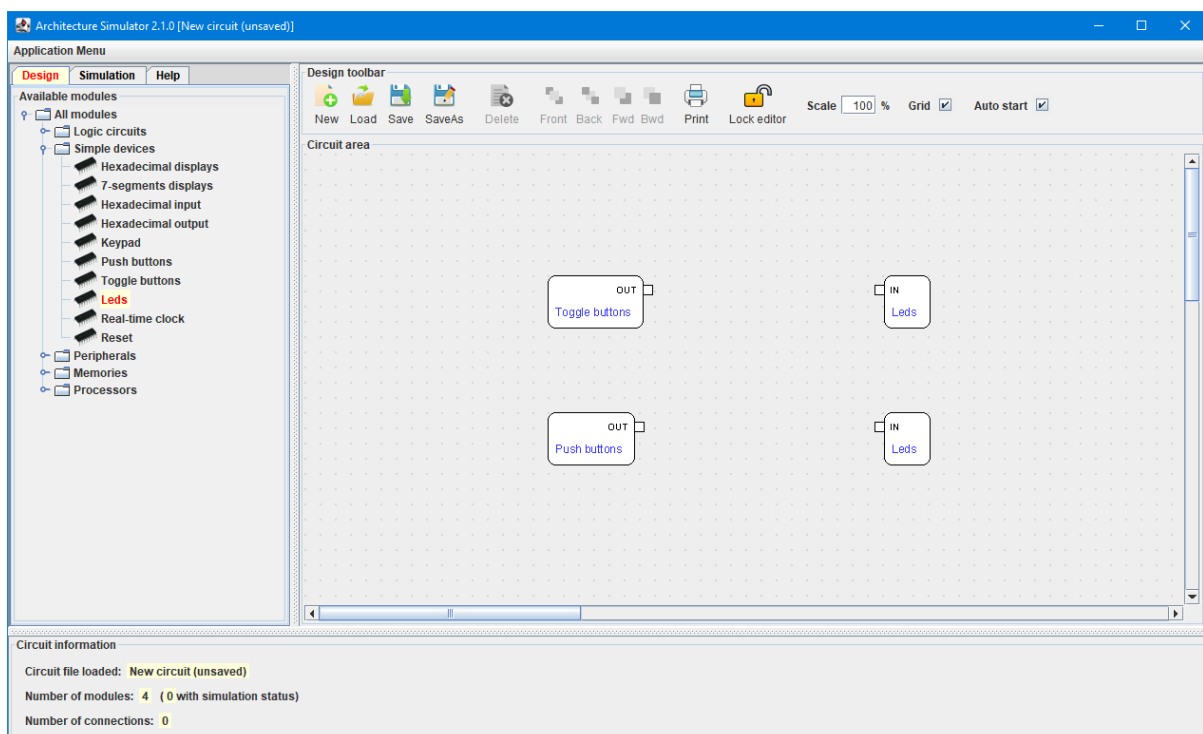
Se abrir com algum circuito que tenha aberto anteriormente, clique no botão **New** (📄) na “Design toolbar” para criar um circuito novo.



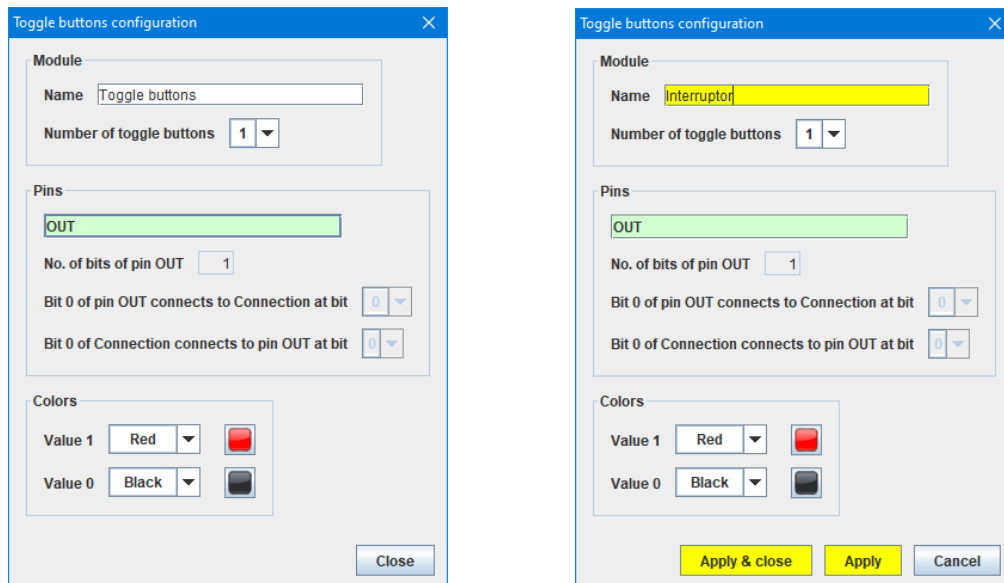
Faça duplo clique em “Simple devices”, o que faz abrir o conjunto de módulos disponíveis nesta categoria.



Clique em “Toggle buttons” e sem largar arraste para o espaço de trabalho, largando depois o botão do rato. Deve aparecer um dispositivo com o nome “Toggle buttons”. Idem para “Push buttons” e “Leds” (este último, faça duas vezes para ter dois dispositivos). No final, terá algo como a janela seguinte (os dispositivos podem ser arrumados fazendo clique e arrastando-os).



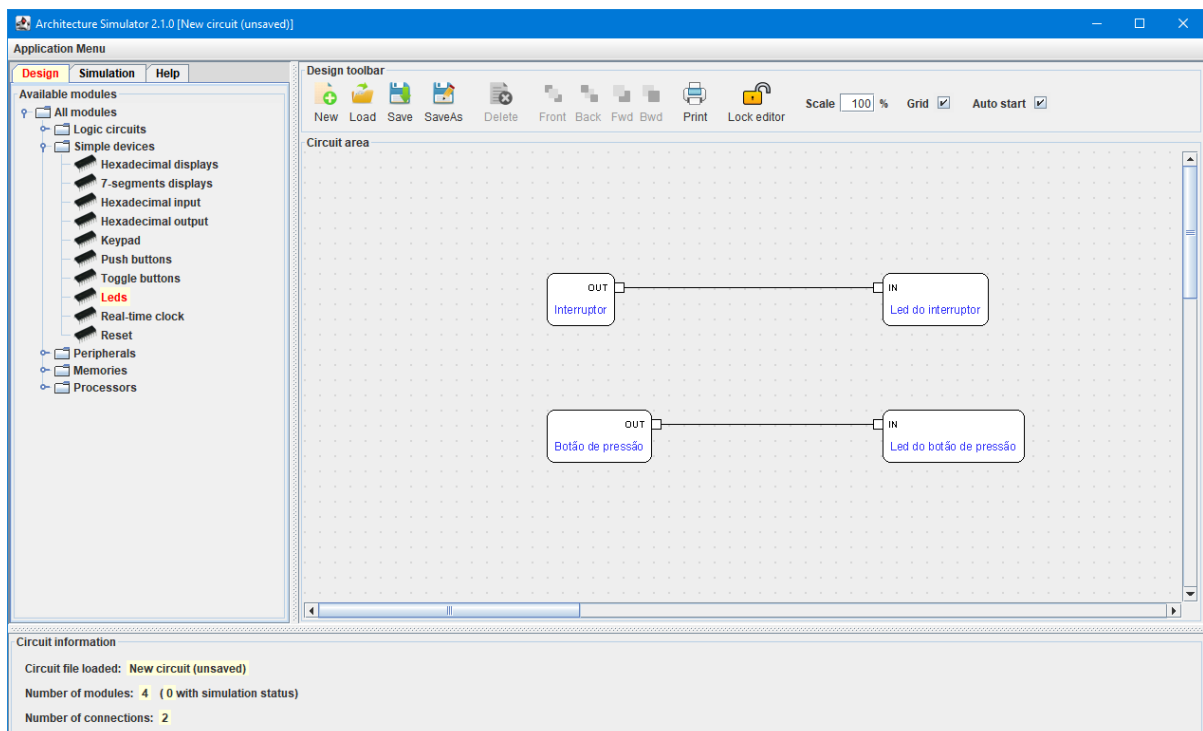
Para mudar o nome do dispositivo “Toggle Buttons”, por exemplo, faça duplo clique nele, o que faz aparecer a sua janela de configuração:



Mude-lhe o nome para “Interruptor” (ou algo a seu gosto). Este dispositivo é um interruptor que em simulação (secção 5.2) alterna entre 0 e 1 (“toggle”) em cada clique e mantém o último estado quando se larga o botão do rato, ao contrário dos botões de pressão (“push buttons”).

Pode também mudar as cores com que o botão aparece. Mude também o nome dos outros dispositivos, para serem mais facilmente identificáveis. Em cada um, tem de carregar em “Apply & close” antes de passar ao próximo.

Para ligar o interruptor e o botão de pressão ao seu led respetivo, clique no pino OUT de cada um e arraste até ao pino IN de cada um dos “Leds”, largando então o botão do rato.

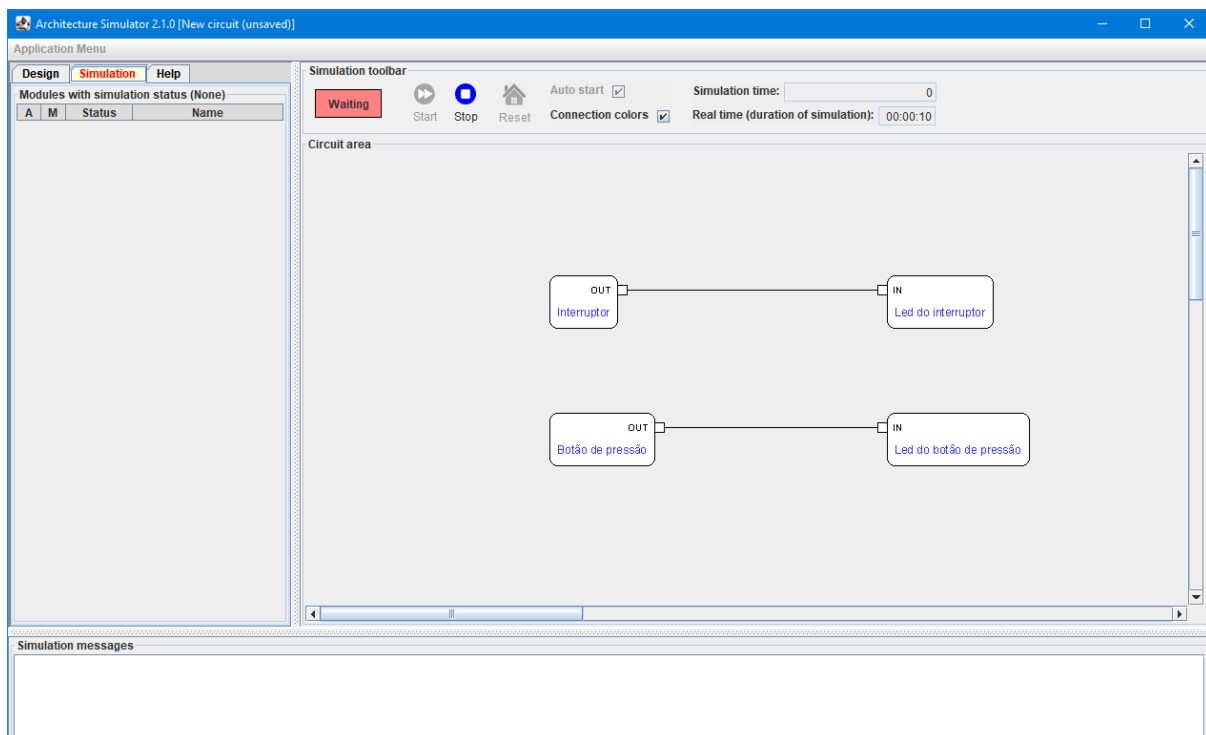


Se quiser, pode guardar este circuito fazendo clique nos botões **Save** (📁) ou **SaveAs** (📁). Pode mais tarde recuperá-lo, simplesmente abrindo o simulador ou fazendo clique no botão **Load** (📁).

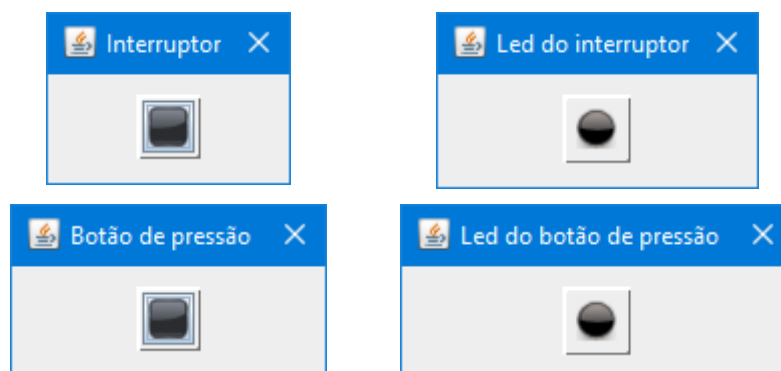
## 5.2 – Simulação do circuito

O circuito está construído. Pode fechar a edição (protegendo o circuito de toques acidentais nos componentes), com o botão **Lock** editor (🔒).

Passe agora para “Simulation” (nas tabs, em cima do lado esquerdo). Dado que o simulador está por omissão em “auto start”, começa logo a simular, esperando por eventos (fazer clique no interruptor, por exemplo).

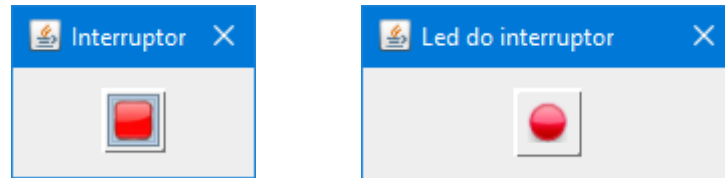


A seguir, precisa de abrir as janelas de interface destes quatro dispositivos, para os poder controlar/visualizar. Tal é feito com duplo clique sobre cada dispositivo. Pode alargar cada uma das janelas até o título estar totalmente visível. Pode também deslocar as janelas para onde quiser no ecrã.

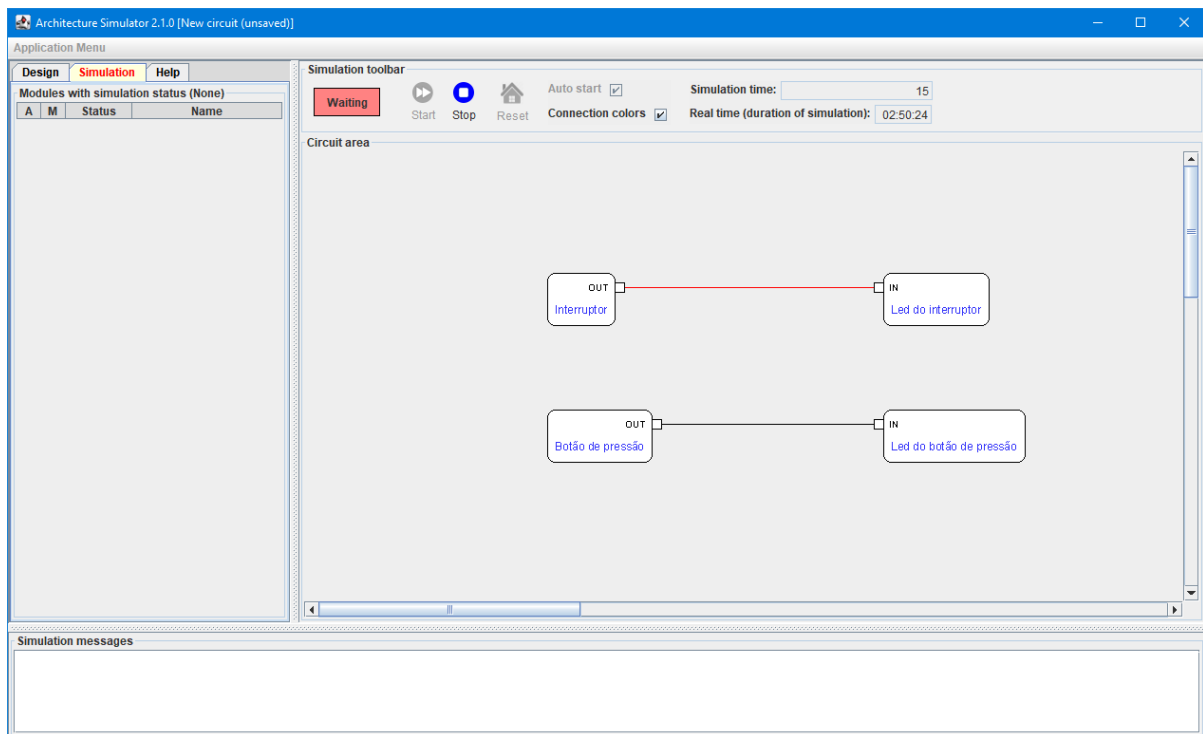


O interruptor e botão de pressão a preto significam que nas suas saídas está 0. Os leds a preto indicam que nas suas entradas está o mesmo valor.

Agora faça clique no botão preto dentro da janela do interruptor. Este muda para vermelho, pois a sua saída (OUT) passou para 1 e o led fica também vermelho, mostrando o novo valor presente na sua entrada (IN).



Na janela de simulação, a ligação entre o interruptor e o led fica também a vermelho, para sinalizar que a ligação tem o valor 1 (a ligação a preto tem valor 0). Isto apenas se verifica nas ligações entre pinos de um bit, como é o caso.



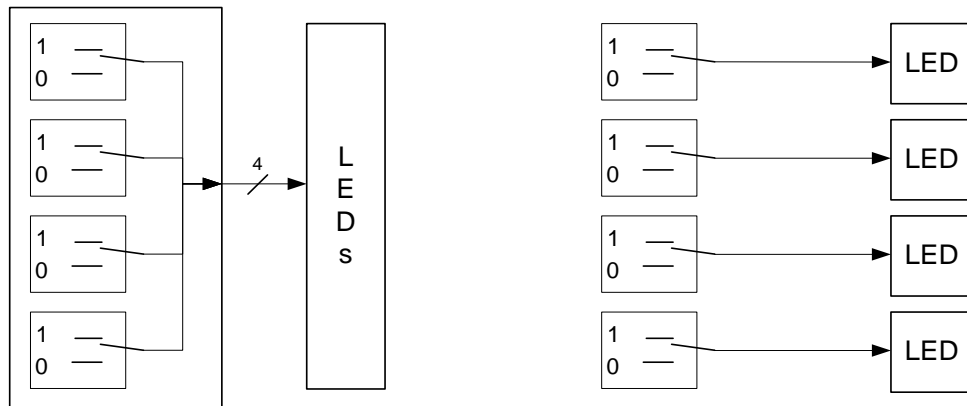
Faça o mesmo para o botão de pressão. Note que neste caso, ao contrário do que sucede com o interruptor) o led apaga-se quando se larga o botão. Há assim forma de gerar um sinal que:

- alterna entre 0 e 1 de cada vez que se carrega no botão do rato e se larga (interruptor), ou
- está a 1 apenas enquanto se está a carregar no botão do rato (botão de pressão).

### 5.3 – Interruptores e leds de vários bits

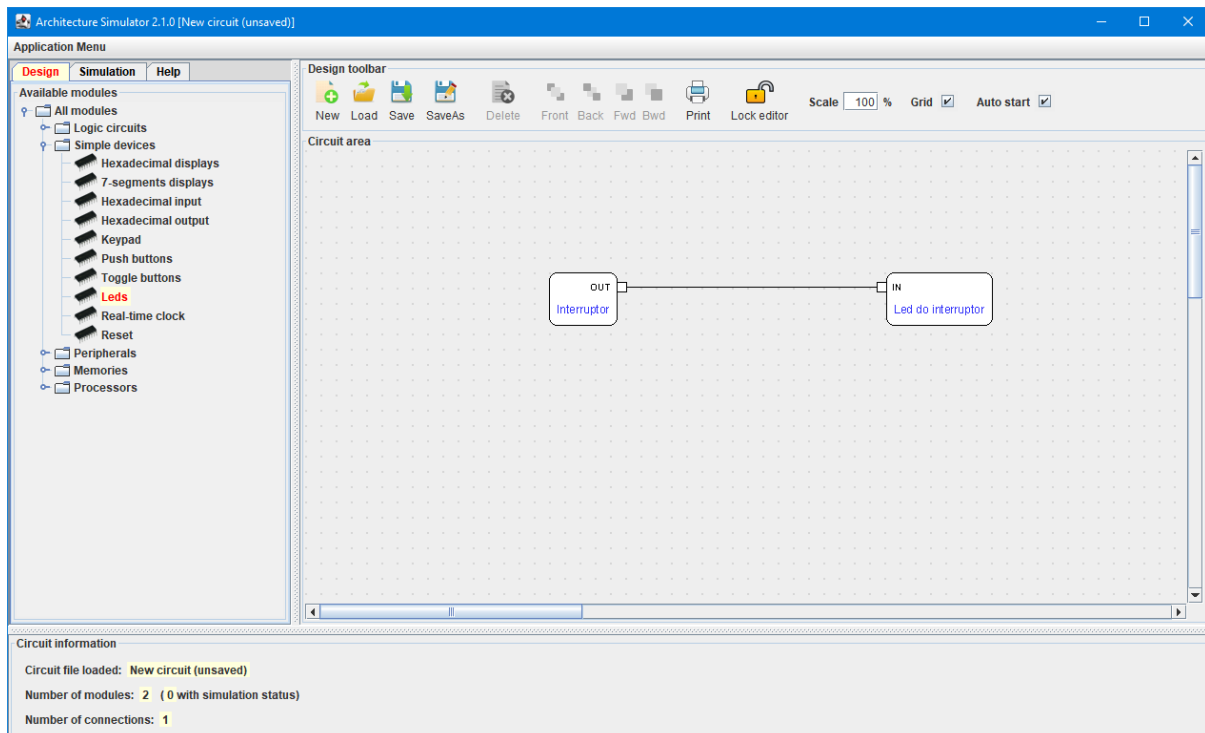
Tanto os interruptores como os leds (e muitos outros módulos) podem ser configurados para ter vários bits. Por exemplo, o lado esquerdo da figura seguinte representa um interruptor de 4 bits ligado a um led também de 4 bits. Nestas condições, a ligação entre os dois fica automaticamente com 4 bits. Do lado direito da figura aparece um circuito funcionalmente equivalente, mas com quatro interruptores e quatro leds individuais.

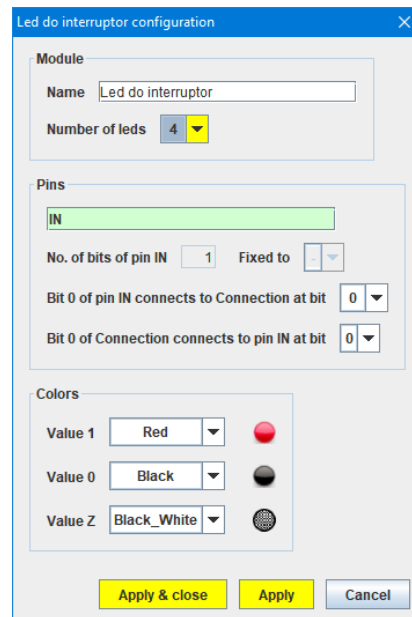
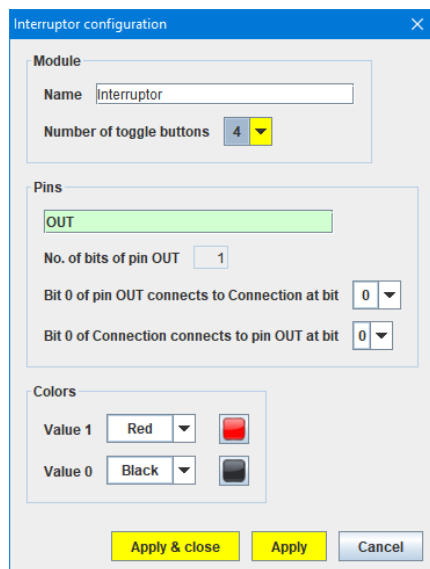
Qualquer dos circuitos se pode simular, mas o circuito do lado esquerdo tem apenas duas janelas de controlo (uma com quatro botões e outra com quatro leds), enquanto no circuito do lado direito se teriam de ter quatro janelas de interruptores e quatro de leds, todos independentes.



O circuito seguinte mostra um interruptor de 4 bits ligado a um led de 4 bits. Pode obtê-lo a partir do circuito anterior selecionando o botão de pressão e o respetivo led e carregando na tecla Delete ou no botão **Delete** da toolbar (🗑️).

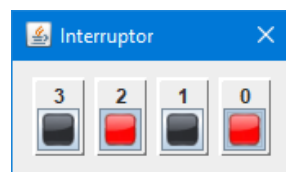
Em termos visuais não se distingue de um interruptor de 1 bit ligado a um led de 1 bit, mas fazendo duplo clique sobre o interruptor em modo Design faz abrir a janela de configuração do interruptor, permitindo configurá-lo para 4 bits. Idem para o led (ver janelas a seguir).





Passando depois para simulação, fazendo duplo clique no interruptor e no led faz abrir as janelas de simulação destes dispositivos, mostrando efetivamente que são de 4 bits. Cada interruptor ou led individual tem por cima a ordem do bit respectivo.

Carregando em cada um dos botões do interruptor, consegue-se controlar o led respectivo de forma independente dos restantes.



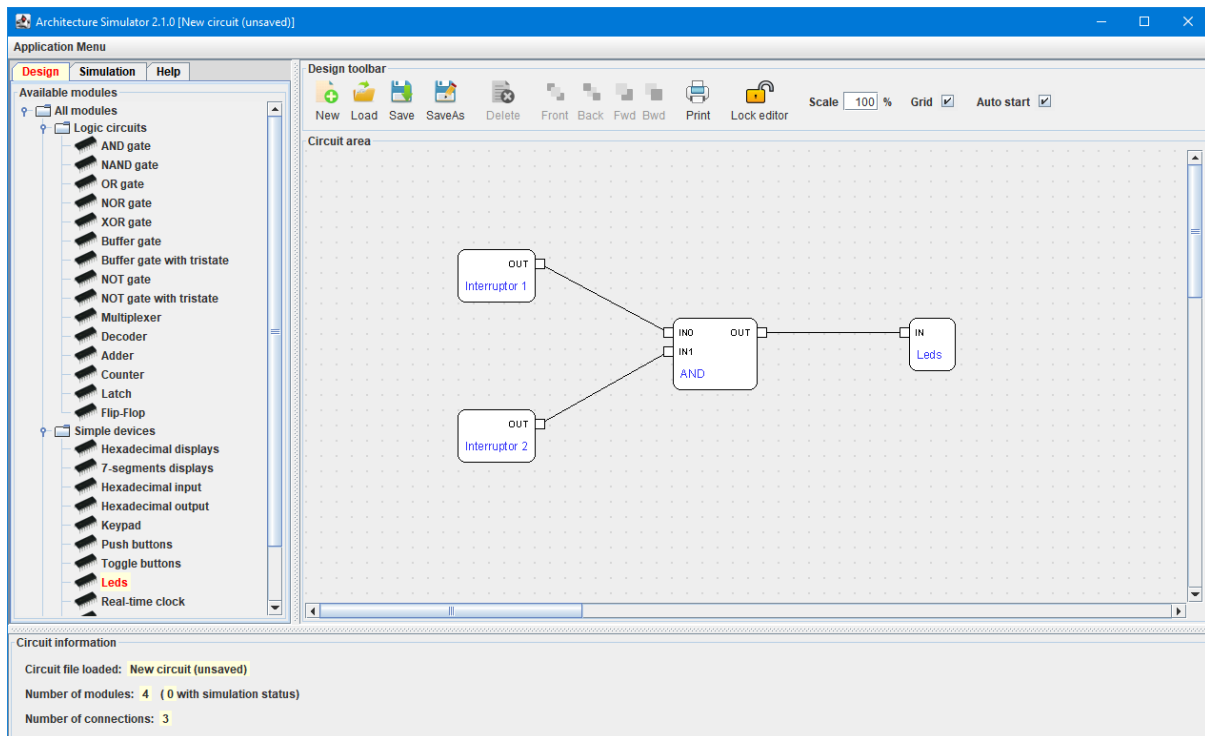
## 6 – Circuito com AND

Passa agora para “Design” e carregue no botão **New** (🔧) para limpar o circuito.

Construa o circuito seguinte, com dois interruptores (“Toggle Buttons”), um AND e um led. O AND está disponível em “Logic circuits”, na palette de módulos do lado esquerdo.

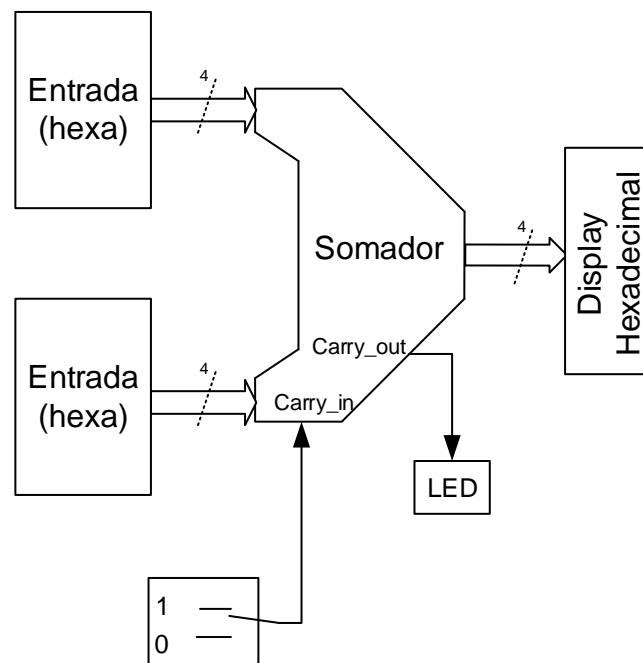
Passa depois para “Simulation” e faça duplo clique nos interruptores e no led para abrir as suas janelas de simulação. Verifique que só quando ambos os interruptores estão a 1 é que o led acende, tal como seria de esperar de um AND.





## 7 – Somador

No simulador, monte o circuito seguinte, um somador de 4 bits com dois operandos:



Este circuito inclui os seguintes módulos:

- **Hexadecimal Input** (entradas hexadecimais, nos “Simple devices”). Há dois, um para especificar cada operando do somador. Faça duplo clique em cada módulo. Por omissão já estão configurados para 4 bits (um dígito hexadecimal), mas pode dar-lhes nomes mais sugestivos, como por exemplo “Operando A” e “Operando B”;

Hexadecimal input configuration

Module

Name: Operando A

Number of hexadecimal digits: 1

Pins

OUT

No. of bits of pin OUT: 4

Bit 0 of pin OUT connects to Connection at bit: 0

Bit 0 of Connection connects to pin OUT at bit: 0

Buttons: Apply & close, Apply, Cancel

Hexadecimal input configuration

Module

Name: Operando B

Number of hexadecimal digits: 1

Pins

OUT

No. of bits of pin OUT: 4

Bit 0 of pin OUT connects to Connection at bit: 0

Bit 0 of Connection connects to pin OUT at bit: 0

Buttons: Apply & close, Apply, Cancel

- **Adder** (somador, nos “Logic circuits”). Faça duplo clique no módulo e configure-o para 4 bits;

Adder configuration

Module

Name: Adder

Delay: 5 (simulation time units)

Number of bits: 4

Number of gate inputs: 2

Pins

CARRY\_IN

CARRY\_OUT

IN0

IN1

OUT

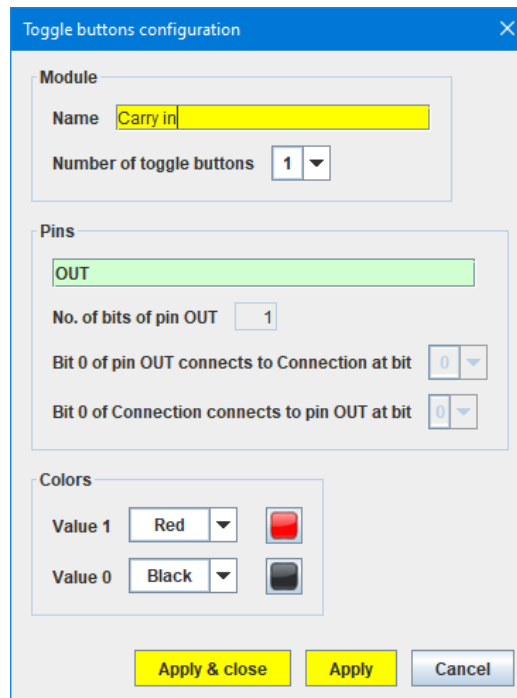
No. of bits of pin CARRY\_IN: 1 Fixed to: -

Bit 0 of pin CARRY\_IN connects to Connection at bit: 0

Bit 0 of Connection connects to pin CARRY\_IN at bit: 0

Buttons: Apply & close, Apply, Cancel

- **Toggle buttons** (interruptor, nos “Simple devices”), usado para especificar a entrada de transporte (*Carry\_in*) do somador. Faça duplo clique no módulo. O número de bits por omissão (um) está certo, mas pode-se mudar o nome para ficar mais explícito;



**Toggle buttons configuration**

**Module**

Name:

Number of toggle buttons:


**Pins**


No. of bits of pin OUT:

Bit 0 of pin OUT connects to Connection at bit:

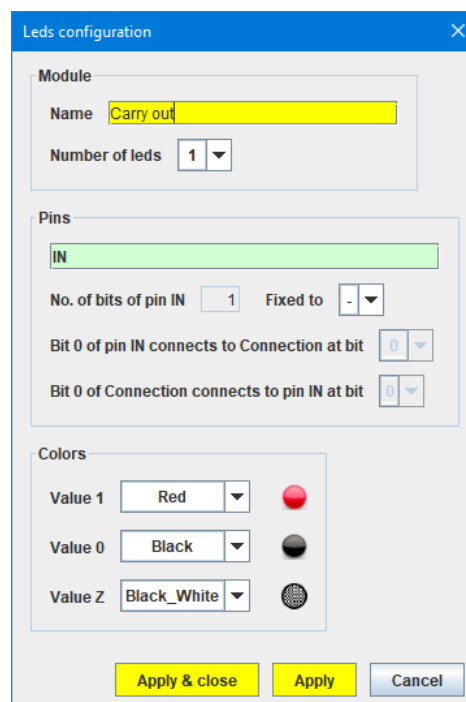
Bit 0 of Connection connects to pin OUT at bit:

**Colors**

Value 1:  

Value 0:  

- **Leds** (nos “Simple devices”), usado para ver o transporte (*Carry\_out*) produzido pelo somador. Faça duplo clique no módulo. O número de bits por omissão (um) está certo, mas pode-se mudar o nome para ficar mais explícito;



**Leds configuration**

**Module**

Name:

Number of leds:


**Pins**


No. of bits of pin IN:  Fixed to:


Bit 0 of pin IN connects to Connection at bit:

Bit 0 of Connection connects to pin IN at bit:

**Colors**

Value 1:  

Value 0:  

Value Z:  

- **Hexadecimal displays** (display hexadecimal, nos “Simple devices”), usado para verificar qual o valor do resultado da soma. O número de dígitos hexadecimais por omissão (um) está certo, não sendo necessário alterar a sua configuração.

Faça as ligações entre os módulos e guarde o circuito num ficheiro (sem extensão ou com extensão “.cir”), fazendo clique nos botões **Save** (📁) ou **SaveAs** (📁). Na secção 9 irá recuperá-lo, fazendo clique no botão **Load** (📁) ou *drag & drop* do ficheiro para a zona de edição do circuito no simulador.

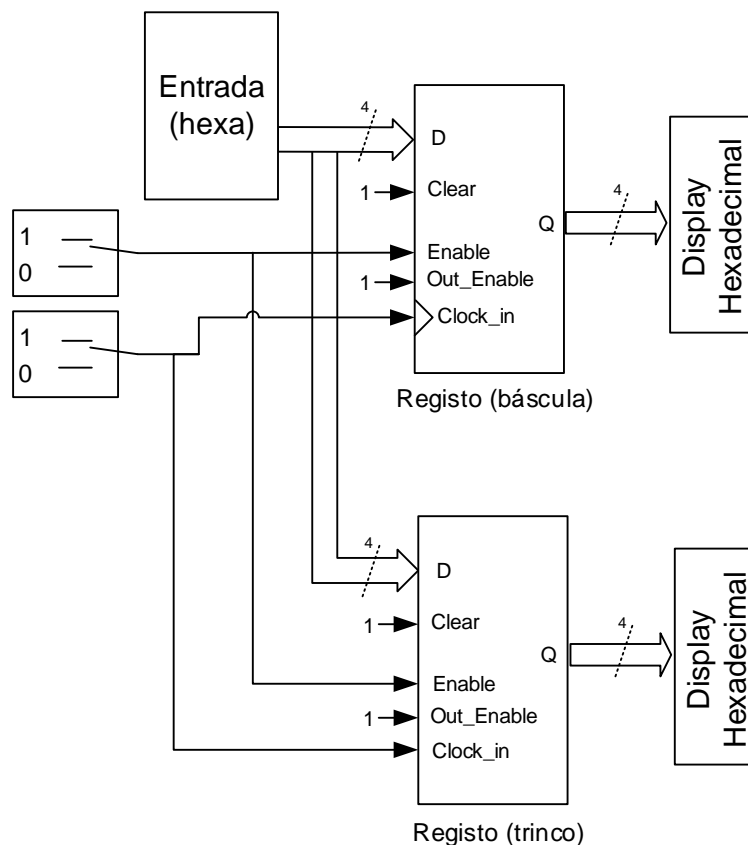
Passe o simulador para Simulação e abra os painéis de simulação de todos os objetos (exceto o somador, que não tem). Verifique que:

- Com Carry\_in = 0, o somador sabe somar (experimente algumas combinações dos valores de entrada, todos com 4 bits);
- Carry\_out vai a 1 quando a soma dos operandos dá transporte 1 (soma > FH);
- Com Carry\_in = 1, o somador adiciona, na prática, mais uma unidade;
- O somador funciona e dá o mesmo resultado quer se considere que os operandos (com 4 bits) estão em notação sem sinal como em notação de complemento para 2. Experimente, por exemplo especificar 8 numa entrada hexadecimal e 7 na outra. Se considerarmos que os dois números estão representados:
  - sem sinal, então  $8 + 7 = 15$ , ou seja, F em hexadecimal;
  - com sinal, então o 8 em complemento para 2 (1000 em binário) é na realidade -8, que somado com 7 dá -1, ou seja, FH (um valor com todos os bits a 1 corresponde a -1).

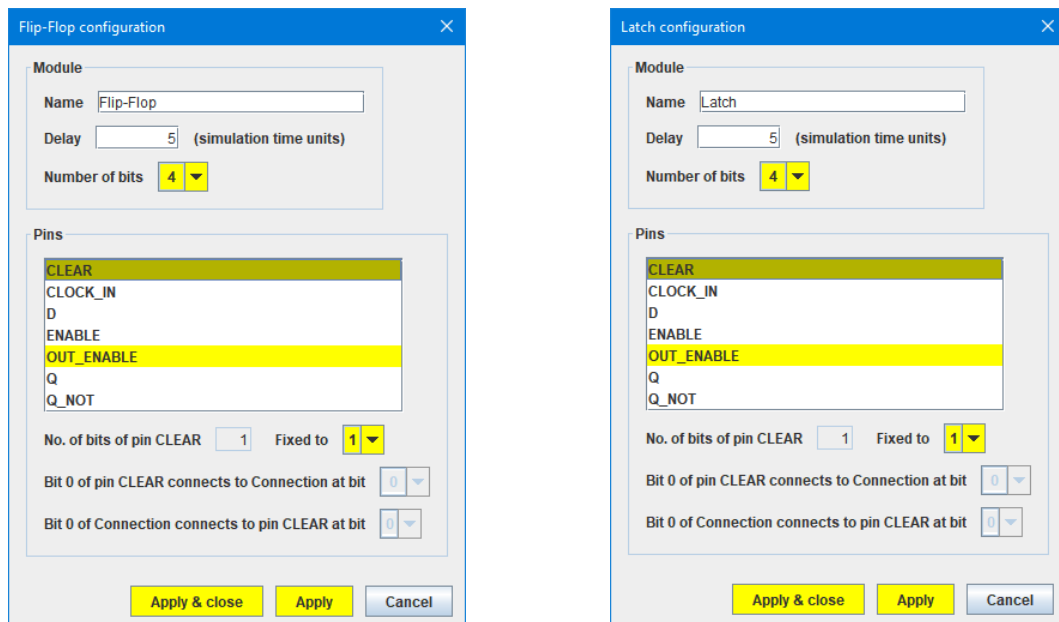
Termine a simulação, passando para Design na janela do simulador.

## 8 – Registo de 4 bits

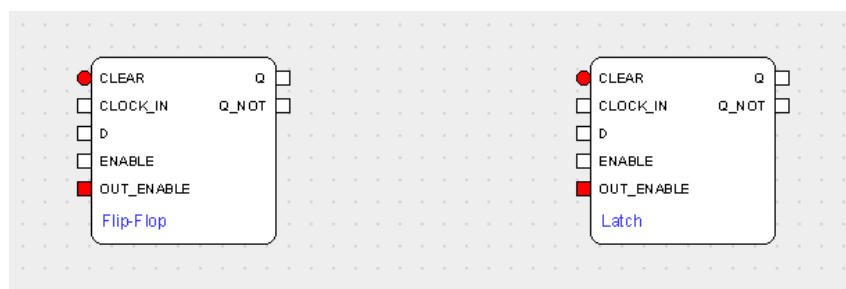
Monte o circuito seguinte, com dois registos de 4 bits, um com básculas (Flip-Flop) e outro com trincos (Latch).



Ambos têm de ser configurados para 4 bits e a sua entrada Clear forçada a 1 (selecione o pino e o “Fixed to”), para não estar ativa (se estivesse a 0, forçava zero na saída dos registos). A entrada Out\_Enable, que a 0 permite desligar as básculas e trincos internos da saída do registo (colocando a saída em *tristate*), deve também estar forçada a 1.



Os pinos forçados a 1 aparecem a vermelho (os forçados a 0 aparecem a preto).



Faça as ligações entre os módulos e guarde o circuito num ficheiro, fazendo clique nos botões **Save** (📁) ou **SaveAs** (💾).

Passa o simulador para Simulação e abra os painéis de simulação de todos os objetos (exceto o somador, que não tem). Verifique que:

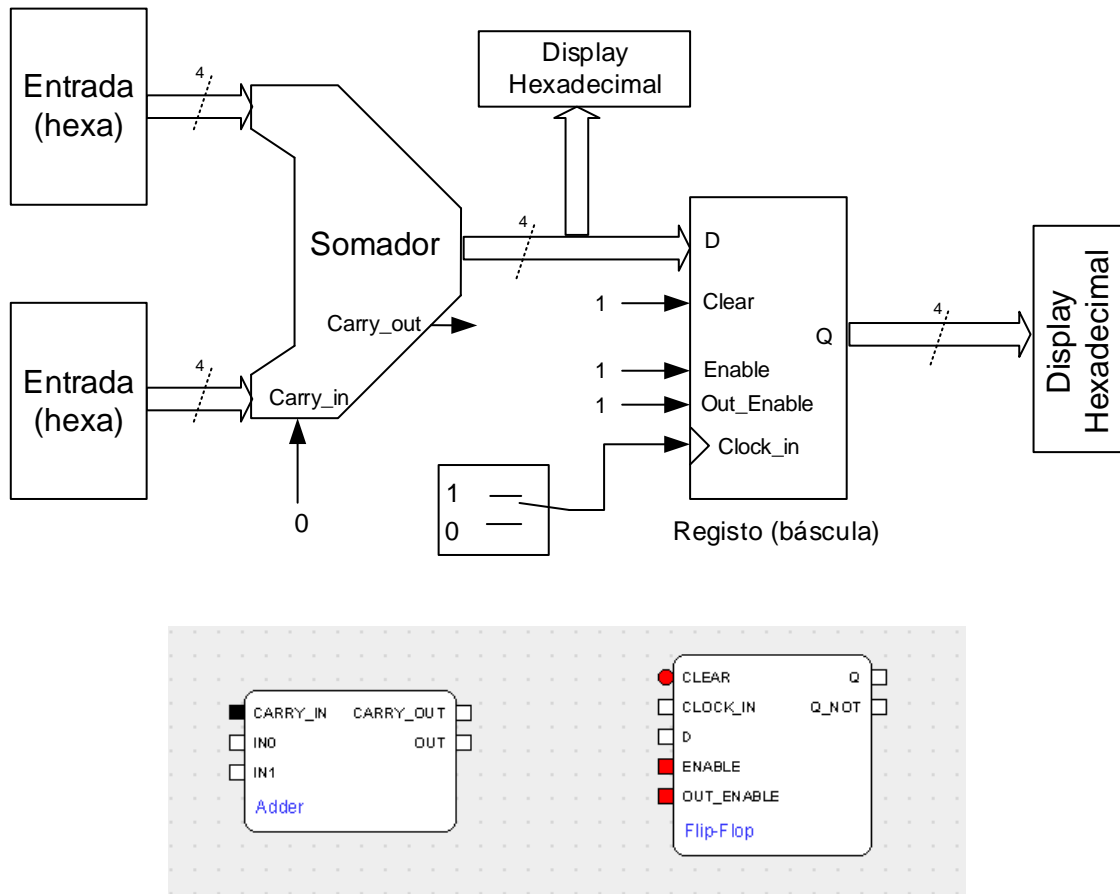
- Os valores que colocar na entrada hexadecimal passam para a saída dos registos apenas:
  - quando** o interruptor que está ligado ao *clock* (relógio) passa de 0 para 1 (flanco ascendente), no caso da báscula (Flip-Flop);
  - enquanto** o *clock* (relógio) está a 1, no caso do trinco (Latch).
- Só quando o interruptor ligado aos pinos Enable estiver 1 os registos podem memorizar o valor da entrada (se o Enable estiver a 0, a saída não muda);

Termine a simulação, passando o simulador para Design.

## 9 – Somador com registo

Recupere o circuito que guardou na secção 7, fazendo clique no botão **Load** (📁) ou *drag & drop* do ficheiro para a zona de edição do circuito no simulador. Altere-o para circuito seguinte.

Configure os módulos Somador (Adder) e Registo com básculas (Flip-Flop) para 4 bits. O *clock* do registo liga a um interruptor de 1 bit. Force a 0 a entrada Carry\_in do somador. Force a 1 as entradas Clear, Enable e Out\_enable do registo.



Faça as ligações entre os módulos e guarde o circuito num ficheiro, fazendo clique nos botões **Save** (📁) ou **SaveAs** (📁).

Passa o simulador para Simulação e abra os painéis de simulação de todos os objetos (exceto o somador e o registo, que não têm).

Verifique o funcionamento do circuito, nomeadamente a diferença de comportamento entre os dois mostradores de 7 segmentos, quando varia os valores dados nos módulos de entrada hexadecimal e depois quando atua no interruptor. O registo só memoriza o valor no flanco ascendente do relógio.

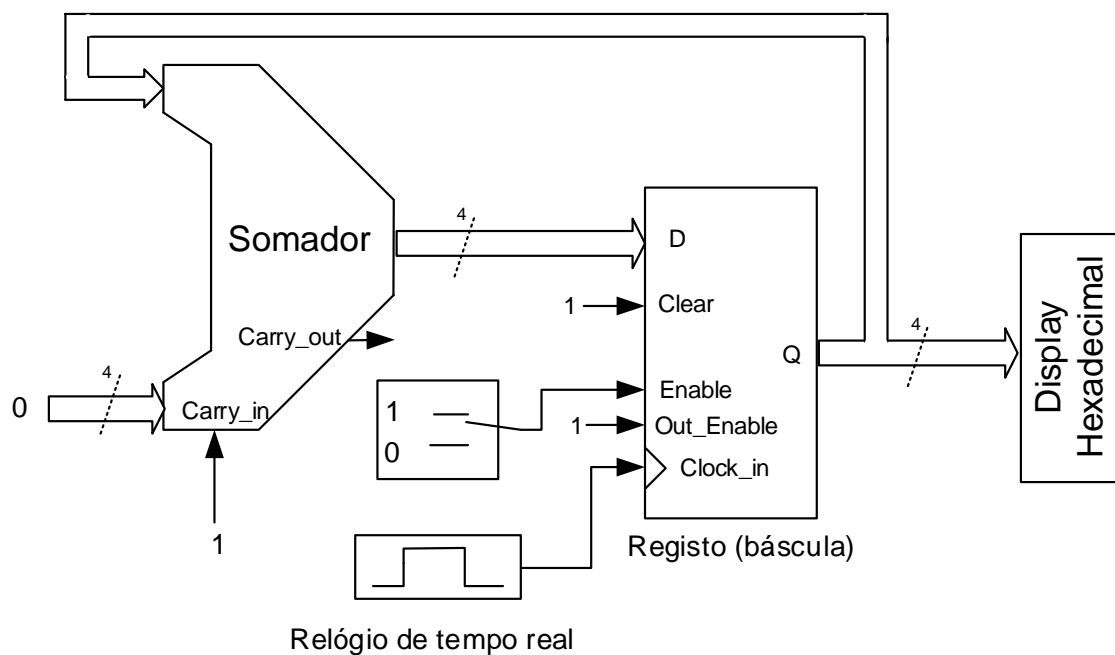
Termine a simulação, passando o simulador para Design.

## 10 – Contador de 4 bits

Altere o circuito que montou na secção 8 para o circuito seguinte, que inclui agora um gerador de relógio de tempo real (módulo “Real-time clock”).

Sugestão: faça uma cópia do ficheiro do circuito anterior, carregue a cópia no simulador, fazendo clique no botão **Load** (📁) ou *drag & drop* do ficheiro para a zona de edição do circuito no simulador, e faça as alterações necessárias.

Configure os módulos Somador (Adder) e Registo com básculas (Flip-Flop) para 4 bits.  
Force a 0 uma das entradas do somador e a 1 a sua entrada Carry\_in. Force ainda a 1 as entradas do registo Clear e Out\_Enable.



Faça as ligações entre os módulos e guarde o circuito num ficheiro, fazendo clique nos botões **Save** (💾) ou **SaveAs** (💾).

Passe o simulador para Simulação e abra os painéis de simulação de todos os objetos (exceto o somador e o registo, que não têm).

Veja qual a funcionalidade do circuito, analisando o comportamento do display à medida que o sinal de saída do relógio vai evoluindo. Verifique que com o interruptor a 0 o contador deixa de contar. Experimente também mudar o período do Relógio de tempo real (mesmo com a simulação a correr).

Termine a simulação, passando o simulador para Design.