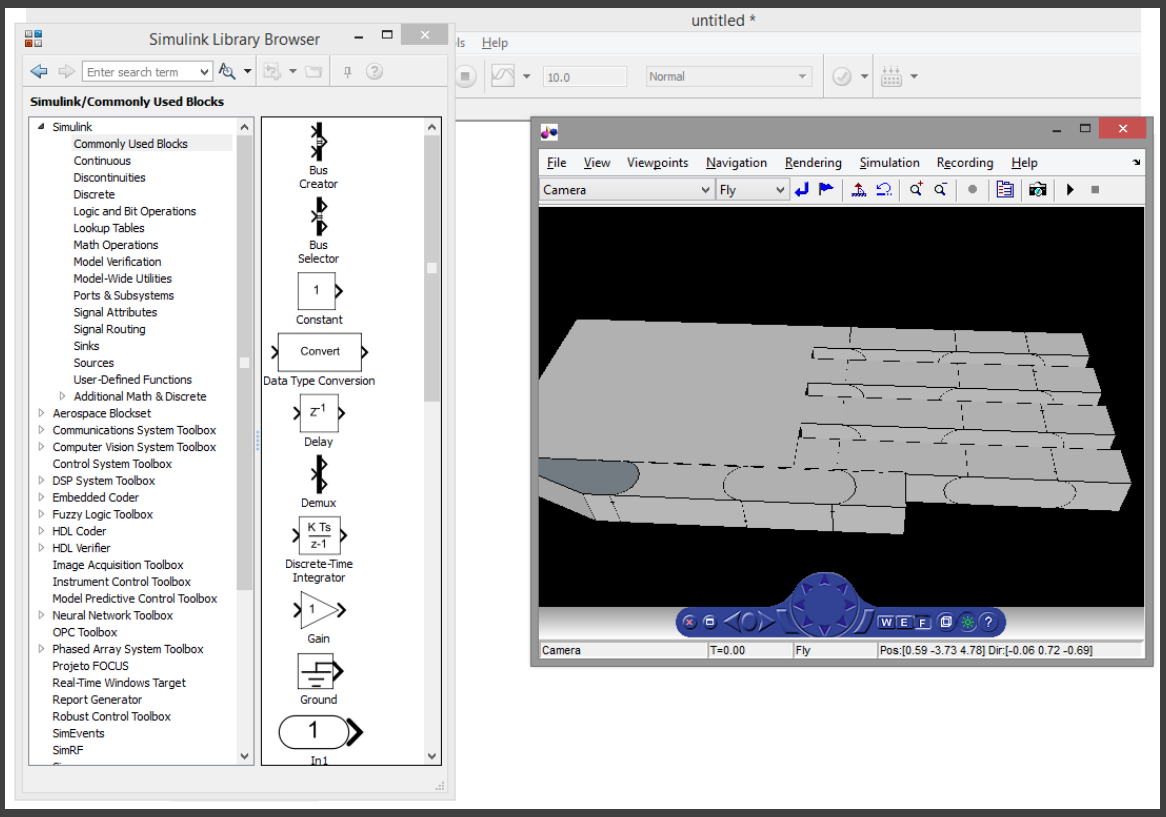
CURSO DE CURTA DURAÇÃO

Introdução ao Simulink®

Aplicações na engenharia



João Paulo Coelho,

Instituto Politécnico de Bragança – Escola Superior de Tecnologia e Gestão

Tatiana Pinho,

Universidade de Trás-os-Montes e Alto Douro – Escola de Ciências e Tecnologia

- Documentação de suporte -

Índice

1. Introdução ao Simulink

O Simulink é uma ferramenta de simulação integrada no *software* MATLAB, que permite modelar, simular e analisar sistemas dinâmicos. Neste sentido uma grande variedade de sistemas podem ser projetados e testados, nomeadamente sistemas de controlo, sistemas de processamento de sinal, sistemas de comunicação, sistemas logísticos, entre muitos outros.

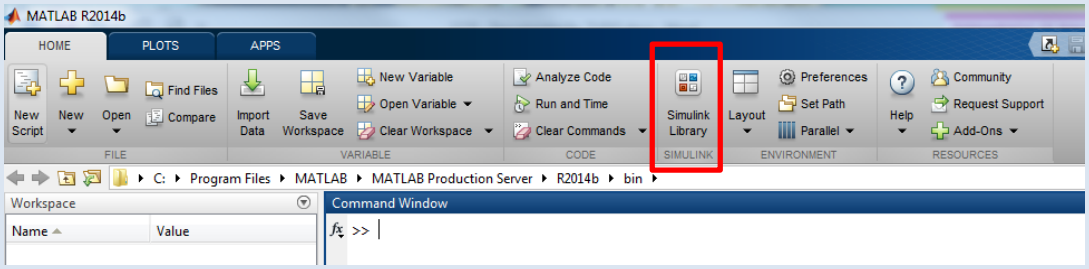
Ao contrário do MATLAB que utiliza a linha de comandos, o Simulink usa uma interface gráfica em que os modelos são criados sob a forma de diagramas de blocos, facilitando a interação com o utilizador. Ainda assim, esta ferramenta pode ser interligada com o MATLAB, podendo inclusive os modelos ser desenvolvidos através da linha de comandos.

O Simulink inclui uma biblioteca de blocos pré-definidos, mas o utilizador pode igualmente criar os seus próprios blocos.

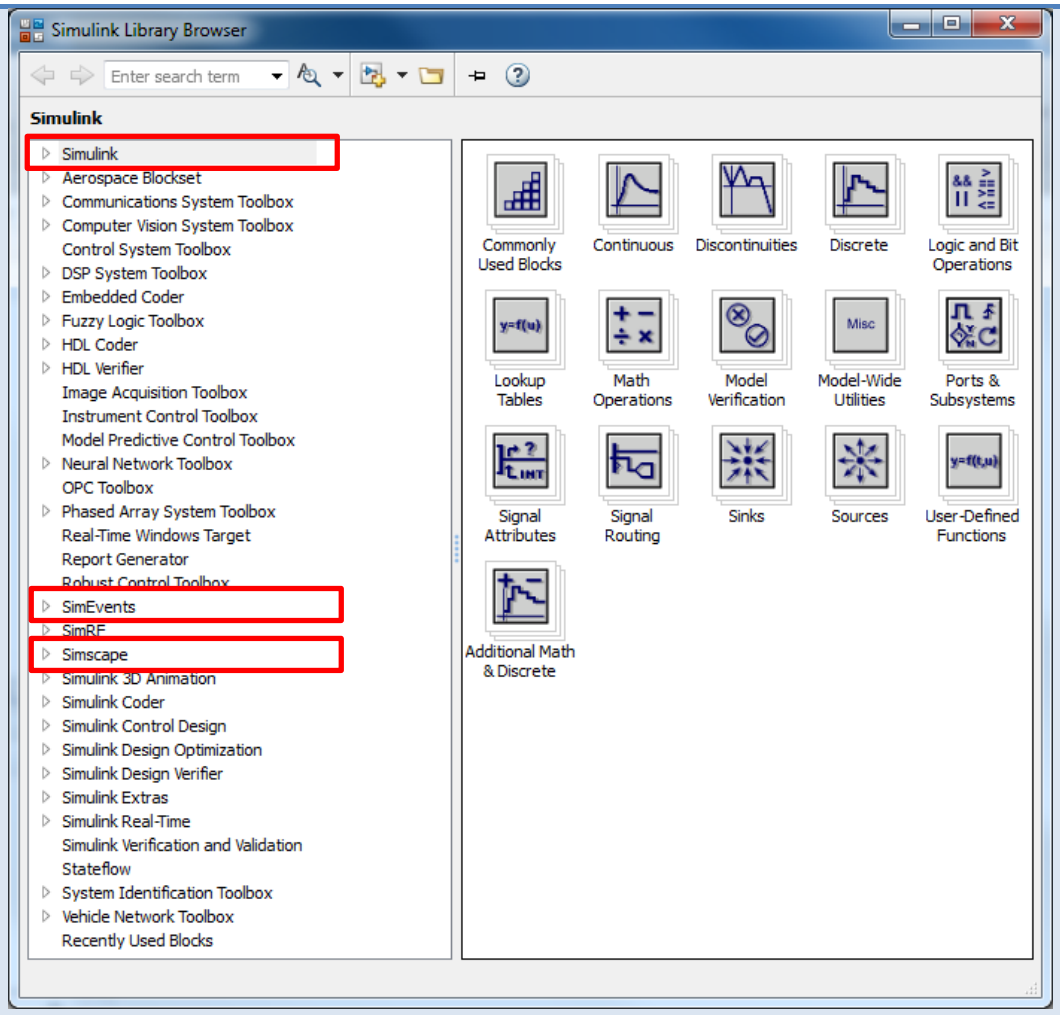
Para abrir a biblioteca do Simulink poderá fazê-lo através do comando

>> simulink

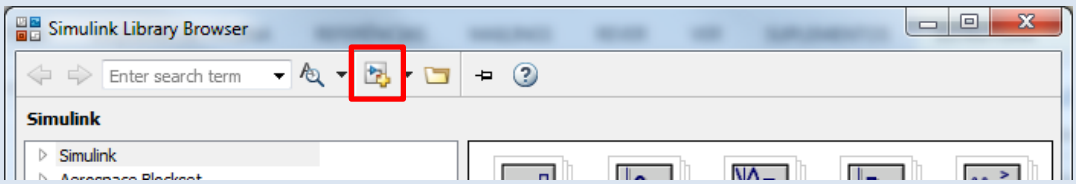
na janela de comandos do MATLAB ou pelo botão "Simulink Library":



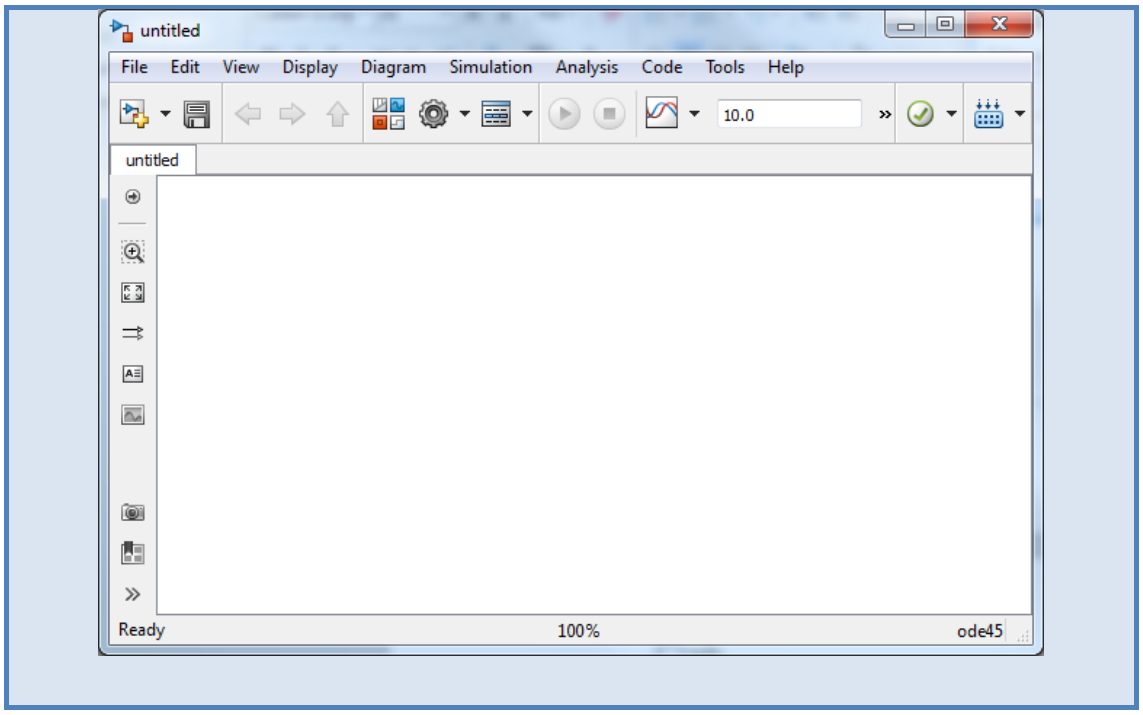
Esta ação irá abrir o "Simulink Library Browser", como apresentado na figura em baixo, em que se poderão pesquisar os blocos a utilizar no modelo. Para além dos blocos básicos da biblioteca do Simulink, outras bibliotecas estão disponíveis, como é o caso do Simscape e do SimEvents que serão utilizadas neste curso.



Para a modelação é necessário abrir um novo modelo através do botão "New Model":



Esta ação irá abrir uma tela em branco em que se podem desenvolver os novos modelos:



Exercício Flash...

Crie um novo modelo no Simulink e guarde-o com o nome “CCDSimulink”.

2. Operações lógicas e aritméticas

O Simulink permite fazer operações de lógica e de aritmética de forma simples. Vamos de seguida ver alguns exemplos deste tipo de operações.

Assim sendo, numa primeira fase, considerando-se duas constantes A = 1 e B = 0, é pretendido testar as três operações lógicas elementares (NOT, AND, OR) para as seguintes situações:

a) NOT A;
b) A AND B;
c) A OR B;
d) A AND (NOT B).

Vamos começar por resolver a alínea a).

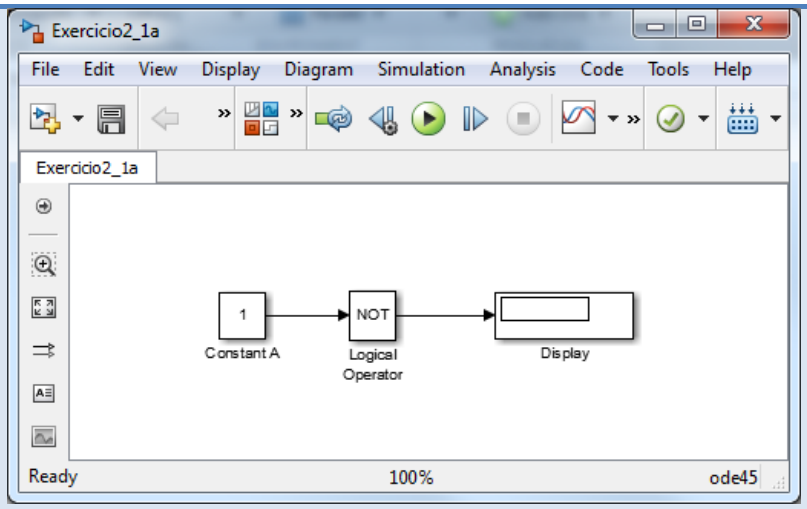
Num novo modelo para o Simulink vamos adicionar os blocos especificados na seguinte tabela, ignorando para já a coluna "Parâmetros":

Bloco	Biblioteca	Quantidade	Parâmetros
Constant	Sources	1	Constant value: 1
Logical Operator	Logic and Bit Operations	1	Operator: NOT
Display	Sinks	1	Default

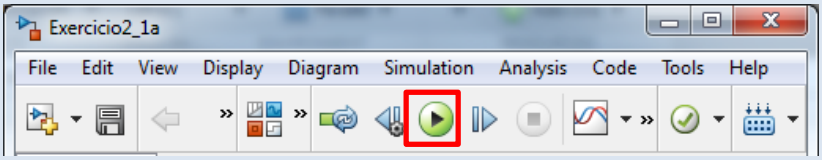
Para colocar os blocos no modelo deve arrastar o bloco pretendido da "Simulink Libray Browser".

De seguida deve configurar os blocos de acordo com a coluna "Parâmetros" da tabela anterior. Para abrir a janela de configuração dos blocos deverá clicar duas vezes sobre o mesmo.

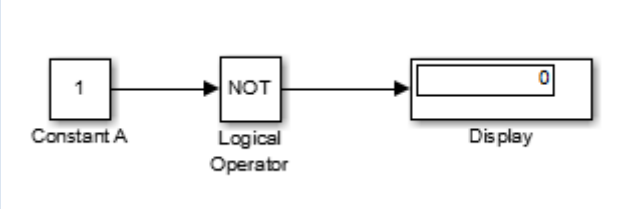
Deverá agora ligar os blocos inseridos na tela. Para ligar os blocos poderá carregar com o botão esquerdo do rato sobre a saída do primeiro bloco e arrastar até ao segundo bloco que se pretende ligar. Alternativamente pode selecionar o primeiro bloco, carregar em CTRL e de seguida selecionar o segundo bloco. O modelo deverá ter o seguinte aspeto:



Por fim, vamos iniciar a simulação (através do botão RUN):

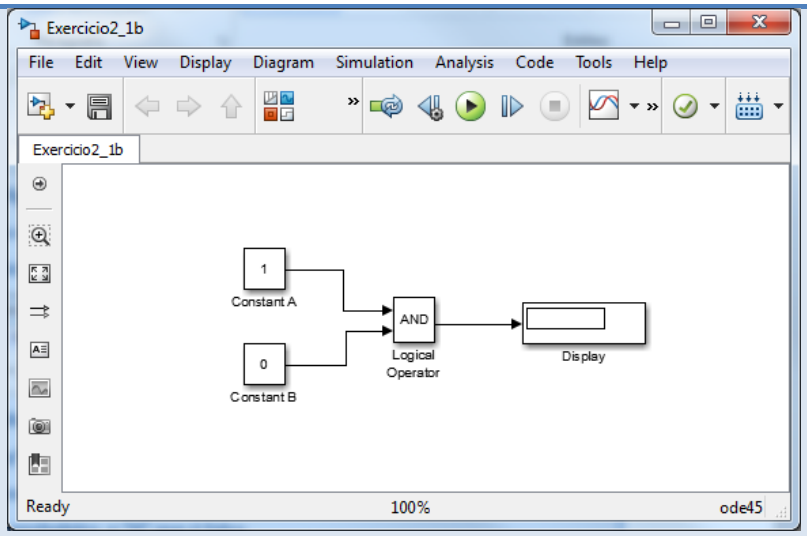


e verificar os resultados apresentados. Uma vez que a constante A foi inicialmente definida como 1, o "Display" deverá apresentar o resultado 0:

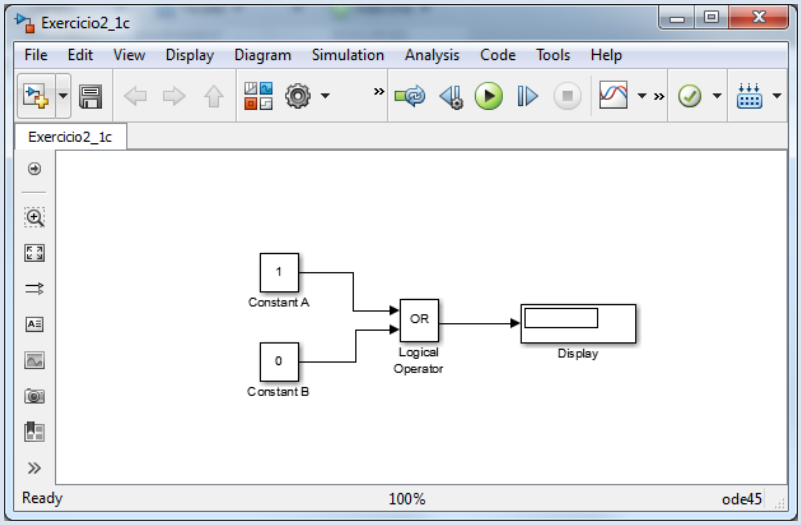


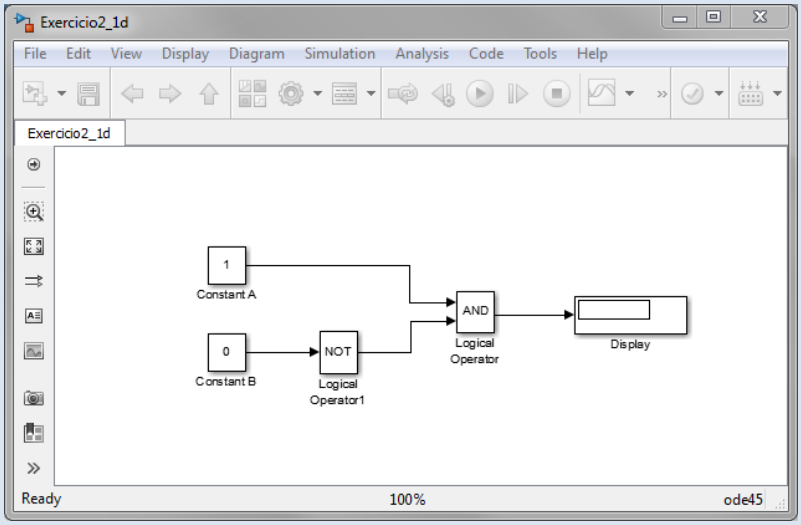
Nota: os operadores lógicos apresentam o resultado na forma binária, pelo que "1" significa que a operação é verdadeira, e "0" que é falsa.

Focando agora a alínea b), vamos adicionar outra constante e modificar o operador do bloco "Logical Operator" para "AND":



De modo semelhante, para as alíneas c) e d) temos, respetivamente:





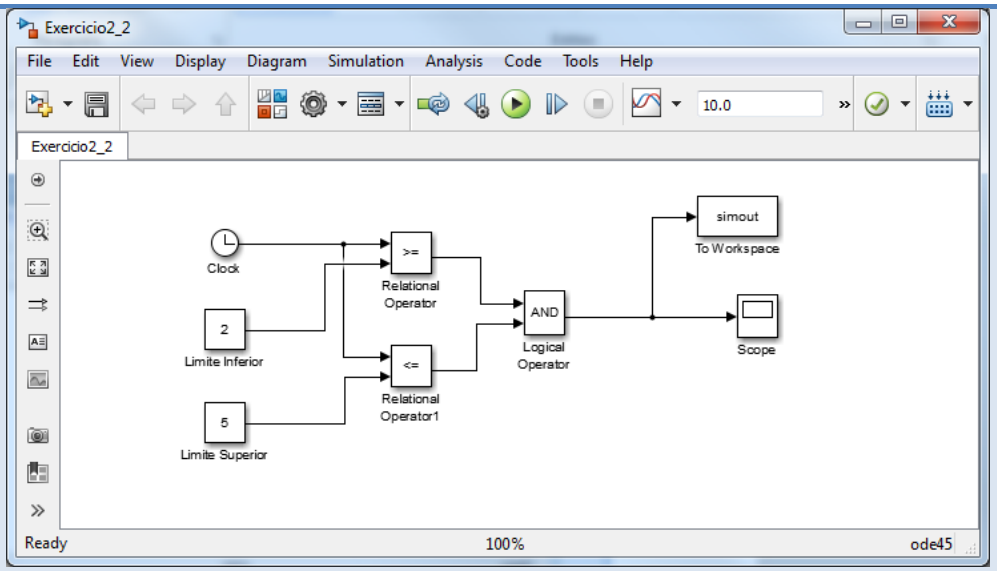
Exercício Flash...

De modo a simular o comando de um LED através de um interruptor (aqui representado pelo bloco "Display" em que "0" indica que está desligado e "1" que está ligado), adicione o bloco "Manual Switch" (para simular o interruptor) e outra constante com o valor 0 à alínea a) e verifique o estado do LED com as alterações no interruptor. Nota: uma vez que temos o operador lógico NOT, o estado do LED deverá ser oposto ao do interruptor.

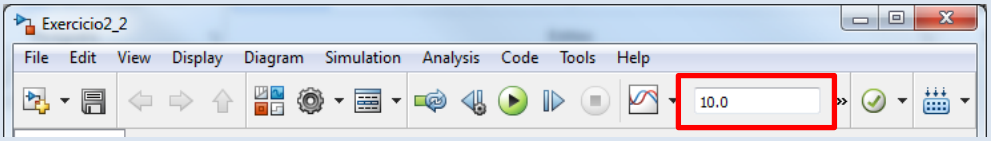
Pretende-se agora analisar o tempo de simulação e assinalar quando o mesmo está entre as constantes 2 e 5 (inclusive), representar graficamente o resultado e exportá-lo para o espaço de trabalho do MATLAB.

Para efetuar esta simulação comece por criar um novo modelo. De seguida adicione os blocos especificados na tabela seguinte, configure-os para com os parâmetros definidos na última coluna e ligue-os para criar a configuração apresentada na figura abaixo:

Bloco	Biblioteca	Quantidade	Parâmetros
Clock	Sources	1	Default
Constant	Sources	2	Constant value: 2 Constant value: 5
Relational Operator	Logic and Bit Operations	2	Relational operator: >= Relational operator: <=
Logical Operator	Logic and Bit Operations	1	Operator: AND
Scope	Sinks	1	Default
To Workspace	Sinks	1	Default

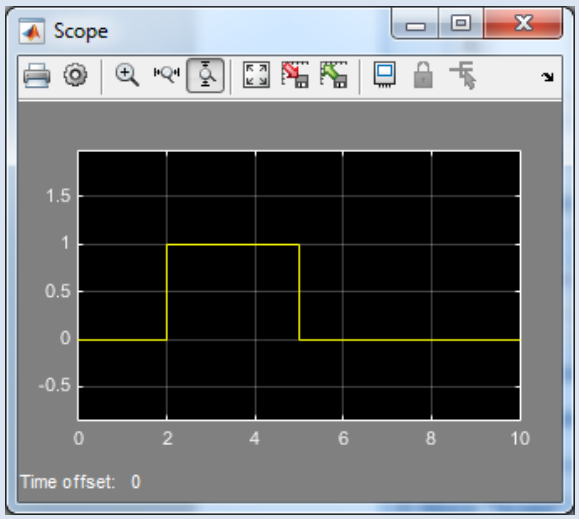


Como o próprio nome indica o bloco "Clock" irá fornecer o tempo de simulação que neste caso está definido para 10:



As unidades do tempo de simulação poderão ser segundos, minutos, horas, etc., conforme o que mais se adequar para o caso em estudo. O que importa é ser coerente ao longo de todo o modelo e utilizar sempre a mesma unidade.

O bloco "Scope" irá apresentar os resultados, que para este caso, deverá ter o seguinte aspeto (para abrir o gráfico faça duplo clique sobre o bloco após terminada a simulação):

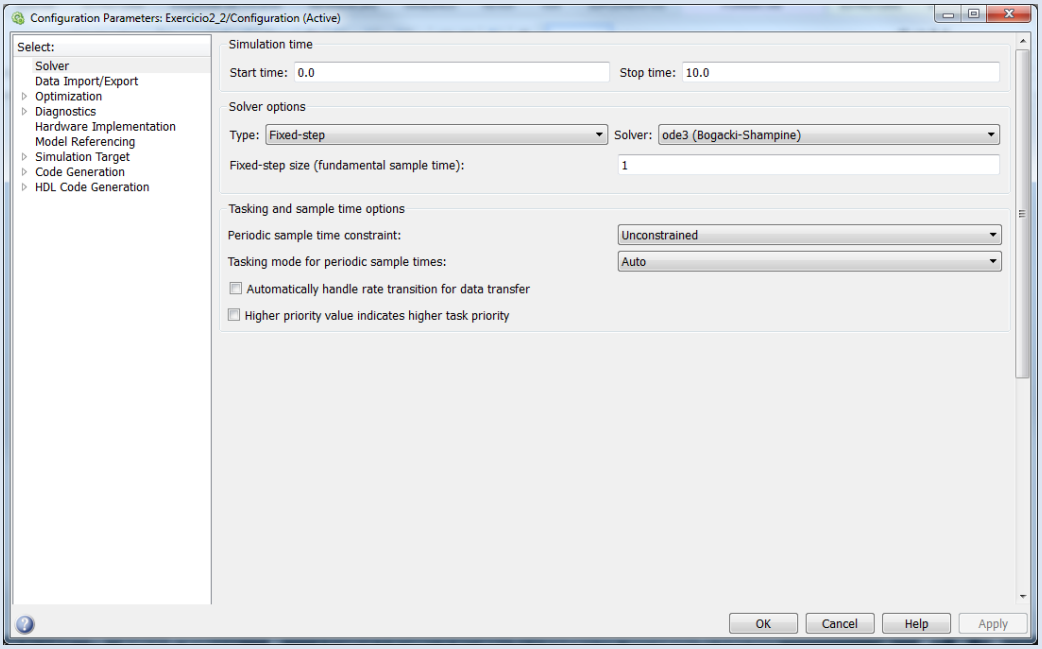


Conforme referido anteriormente o operador lógico AND retorna um sinal do tipo binário, pelo que quando a condição em análise for verificada, o seu resultado será 1. Assim sendo, e como pretendido, o gráfico mostra o sinal a 1 entre os instantes 2 e 5.

Conforme era pedido, os resultados da simulação foram também enviados para o espaço de trabalho do MATLAB através do bloco "To Workspace" com o nome "simout". Para aceder aos resultados escreva o seguinte comando:

```
>> [simout.data  simout.time]
```

Como pode verificar uma matriz de 2 colunas aparece, em que a primeira coluna representa o sinal e a segunda o tempo. No entanto se verificarmos o passo do tempo de simulação não é o mais adequado dificultando a visualização dos resultados. Vamos alterar isto de modo a que o tempo seja apresentado com intervalos de 1 unidade. Para tal, no modelo criado, no separador "Simulation" selecione "Model Configuration Parameters". Em "Solver options" vamos definir o "Type" para "Fixed-step" e o "Fixed-step size (fundamental sample time)" para 1:



Carregue "OK", execute de novo a simulação e na linha de comandos escreva novamente:

```
>> [simout.data  simout.time]
```

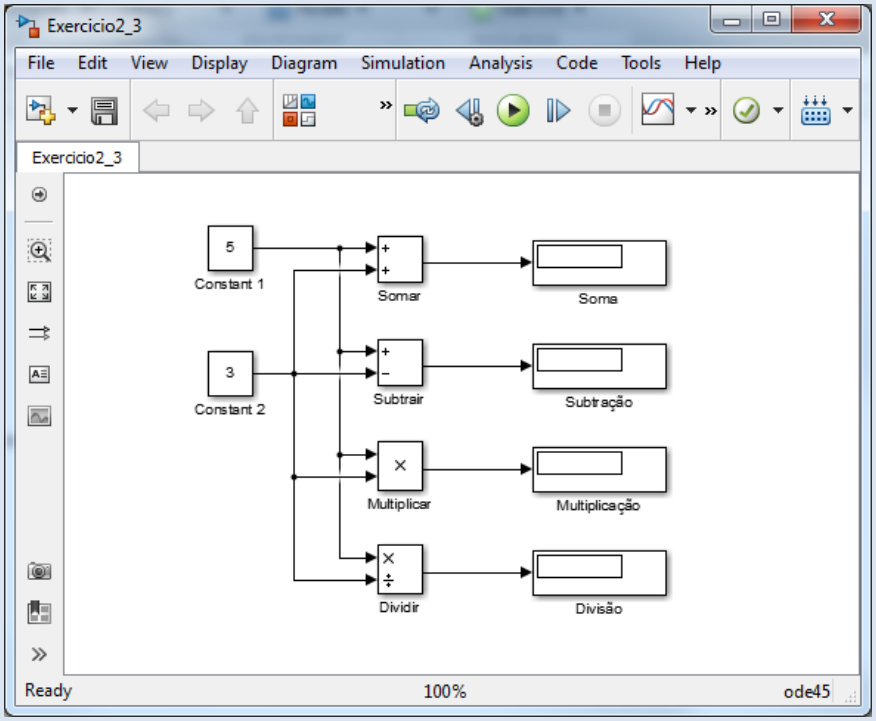
Desta forma a matriz criada possui 2 colunas com apenas 11 linhas, facilitando a sua análise.

De modo a compreender melhor as operações aritméticas no Simulink, vamos agora efetuar uma simples calculadora com as operações de soma, subtração, multiplicação e divisão para duas constantes.

Crie um novo modelo para o Simulink. Na nova tela criada vamos adicionar os seguintes blocos:

Bloco	Biblioteca	Quantidade	Parâmetros
Constant	Sources	2	Constant value: 5 Constant value: 3
Add	Math Operations	2	List of signs: ++ List of signs: +-
Product	Math Operations	1	Default
Divide	Math Operations	1	Default
Display	Sinks	4	Default

Configure os blocos "Constant" e "Add" e ligue-os da seguinte forma:



Simule o modelo e verifique os resultados.

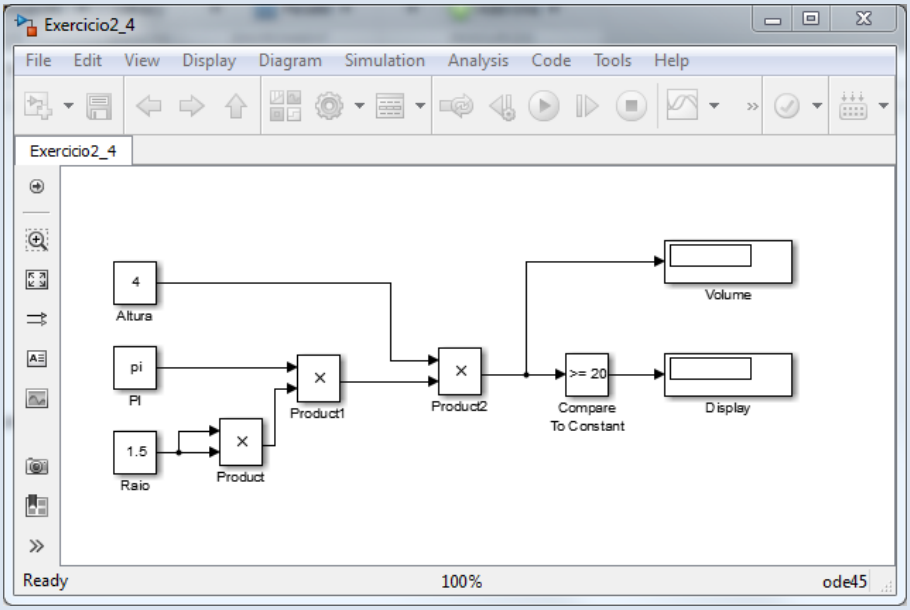
Exercício Flash...

Desenhe um modelo que efetue a soma e multiplicação de dois números e apresente a divisão dos resultados.

Por fim, pretende-se calcular o volume de um cilindro ($V = \pi r^2 h$) e verificar se o resultado é maior ou igual que uma constante pré-definida.

Num novo modelo coloque os blocos especificados na seguinte tabela de acordo com o diagrama subsequente.

Bloco	Biblioteca	Quantidade	Parâmetros
Constant	Sources	3	Constant value: 4 Constant value: pi Constant value: 1.5
Product	Math Operations	3	Default
Compare To Constant	Logic and Bit Operations	1	Operator: >= Constant value: 20
Display	Sinks	2	Default



Configure os blocos das constantes para os valores de π (no bloco introduzir pi), do raio (1.5 cm) e da altura (4 cm). Para além disso, configure o bloco "Compare To Constant" para verificar se o volume calculado é maior ou igual a 20.

De seguida inicie a simulação e verifique o resultado do volume e da operação de comparação ("1" –> verdadeiro; "0" –> falso).

3. Controlo de um Motor

É possível simular o comportamento de qualquer sistema físico no Simulink. Para isso basta ter um modelo que represente o seu comportamento dinâmico. De modo a ilustrar essa competência considera-se o caso da modelação de um motor elétrico DC de íman permanente. Em particular o dispositivo disponível comercialmente pela referência EMG-30. Trata-se de um motor desenhado para operar com uma tensão nominal de 12V e que integra, para além de uma caixa redutora de 30:1, um codificador incremental de posição. O seu aspeto exterior encontra-se representado na figura.

Figura 1. Aspeto do motor EMG - 30 que iremos modelar no Simulink.

Modelo Matemático

A simulação de um sistema físico requer um modelo capaz de representar o seu comportamento. Esse modelo é, por norma, uma simplificação do sistema real e a sua complexidade é a suficiente para capturar as dinâmicas relevantes. Para o caso do motor elétrico DC esse modelo será descrito sob a forma de uma equação diferencial de segunda ordem obtida a partir da conjugação das leis de Newton para a rotação com as leis de Kirchhoff para os circuitos elétricos.

Quando um motor elétrico gira, e devido ao fenómeno de indução, uma diferença de potencial aparece aos terminais da armadura. Essa tensão é designada por força contraelectromotriz e é proporcional à velocidade angular do motor. Se k_e for essa constante de proporcionalidade então:

$$e(t) = k_e \cdot \omega(t) \tag{1.1}$$

onde $e(t)$ se refere à força contraelectromotriz, medida em volt e $\omega(t)$ a velocidade angular do rotor em rad/s.

Relativamente à componente mecânica, o que gira num motor é o rotor que possui uma determinada massa e geometria. Logo possui um dado momento de inércia J . Adicionalmente existe atrito entre as partes móveis do motor. O valor desse atrito é definido através da especificação da constante de atrito k_b . A Figura 2 representa esquematicamente cada uma dessas grandezas:

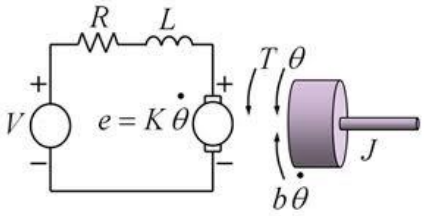


Figura 2. Representação esquemática de um motor DC.

À esquerda nessa figura tem-se a componente elétrica do motor e à direita a componente mecânica. Relativamente à primeira, seja $v(t)$ a tensão aplicada ao rotor e $i(t)$ a corrente que nele circula. Pela lei das malhas de Kirchhoff obtém-se:

$$v(t) - e(t) - L\frac{di(t)}{dt} = Ri(t) \tag{1.2}$$

Atendendo à equação (1.1) a expressão anterior toma a seguinte forma:

$$v(t) - k_e\omega(t) - L\frac{di(t)}{dt} = Ri(t) \tag{1.3}$$

Relativamente à componente mecânica tem-se:

$$\tau_b(t) = k_b\omega(t) \tag{1.4}$$

para o momento de binário associado à força de atrito e:

$$\tau_m(t) = k_m i(t) \tag{1.5}$$

para o momento de binário desenvolvido pelo motor. Note-se que a constante k_m é expressa em Nm/A.

Atendendo agora à segunda lei de Newton para o movimento rotacional fica:

$$\tau_m(t) - \tau_b(t) = J\alpha(t) \tag{1.6}$$

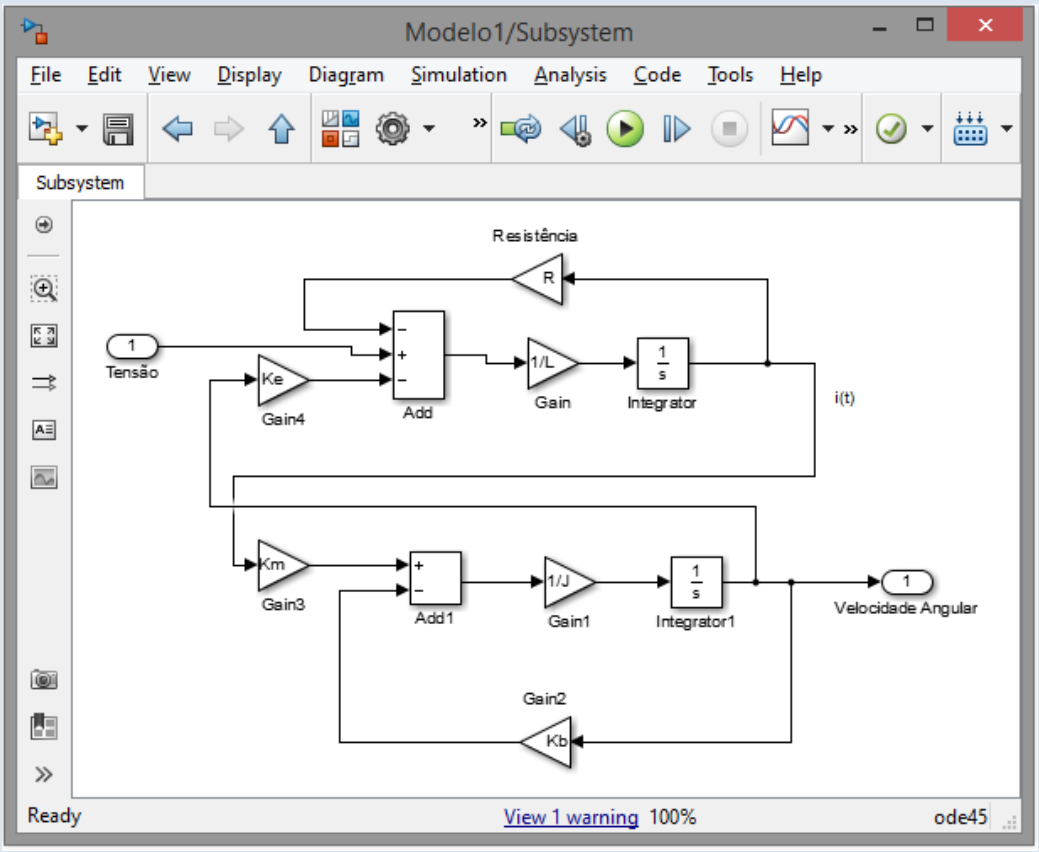
onde $\alpha(t)$ se refere à aceleração angular em rad/s^2

Assim sendo,

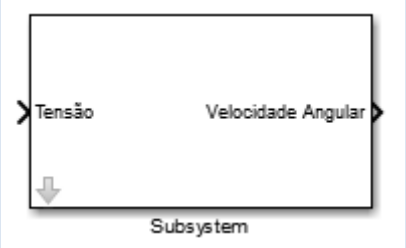
$$k_m i(t) - k_b\omega(t) = J\frac{d\omega(t)}{dt} \tag{1.7}$$

Com base nas equações (1.2) e (1.7), e recorrendo ao Simulink, vamos modelar o comportamento do motor no que diz respeito à velocidade do rotor em função da tensão aplicada ao rotor.

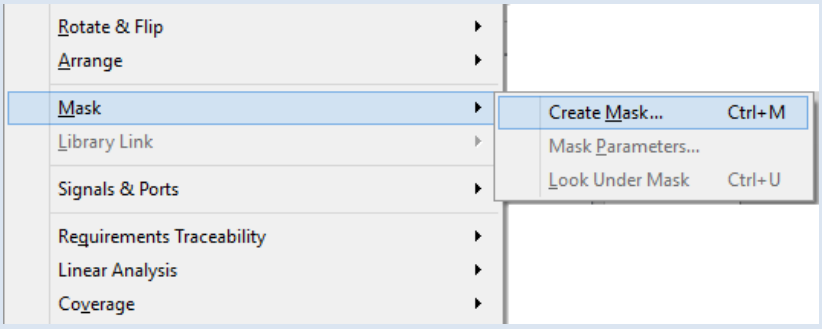
Crie um novo modelo para o Simulink. Na nova tela vamos construir o seguinte diagrama de blocos:



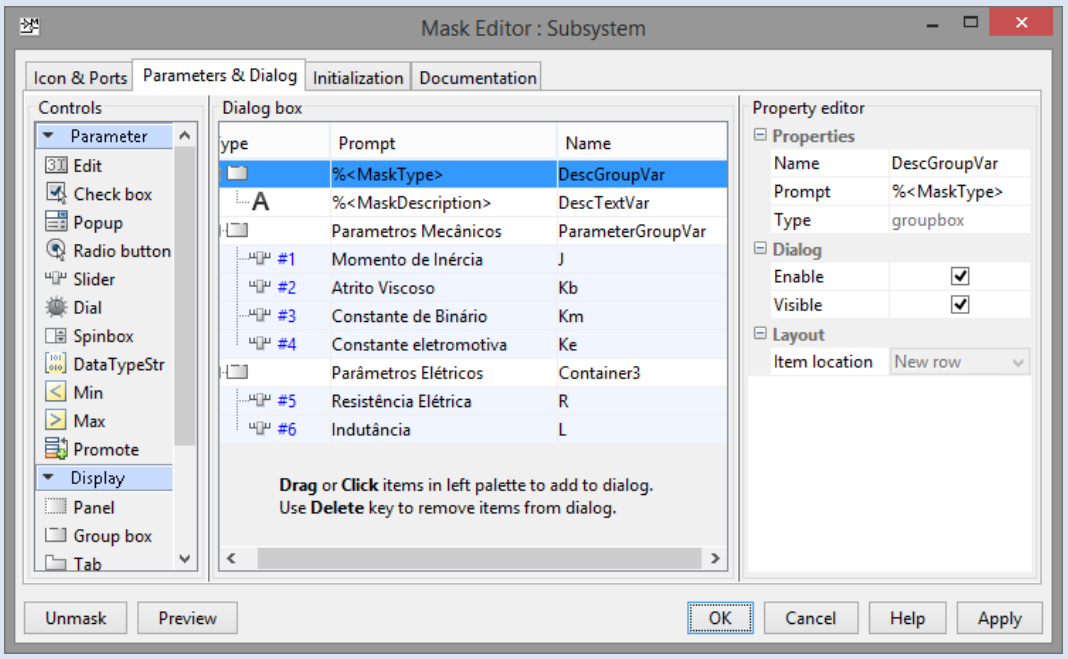
Os blocos utilizados estão nas paletas <Sources>, <Sinks>, <Continous> e <Math Operations>. Agora utilizando o dispositivo apontador selecione todos os objetos e pressionando o botão direito escolha, no menu que aparece, a opção "Create Subsystem from Selection". O resultado é a criação do seguinte subsistema:



Novamente com o botão direito do rato sobre este bloco selecione a opção "Create Mask..." do item "Mask".



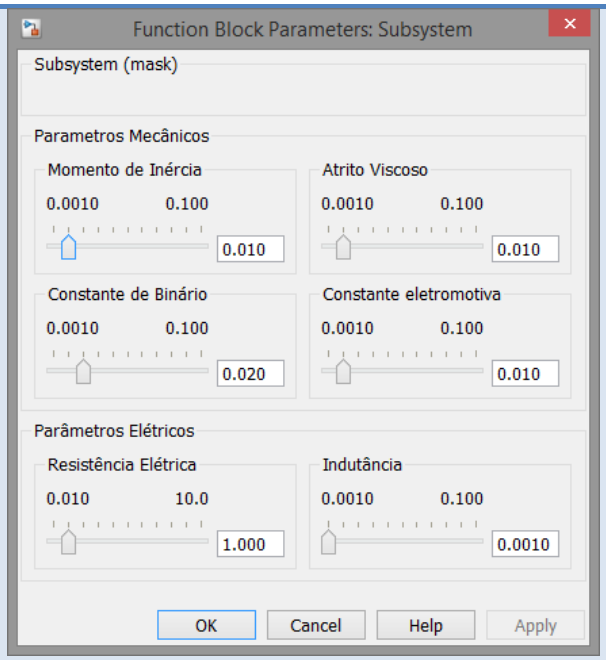
Na janela que se irá abrir escolhemos a TAB "Parameters & Dialog" e inserimos seis parâmetros como vê na figura seguinte.



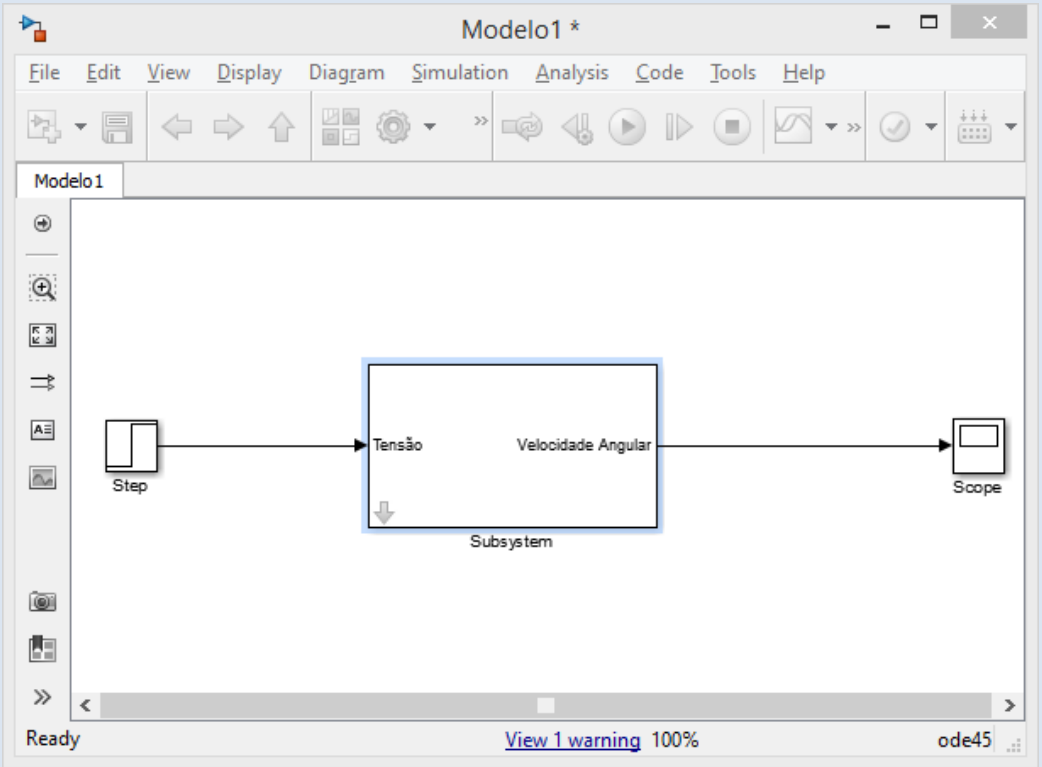
Todos os parâmetros serão do tipo "Slider" de acordo com as seguintes especificações:

	Name	Value	Minimum	Maximum	Item Loc.	Orientation
Momento de Inércia	J	0.01	0.001	0.1	-	Horizontal
Atrito Viscoso	Kb	0.01	0.001	0.1	Current Row	Horizontal
Constante de Binário	Km	0.02	0.001	0.1	New Row	Horizontal
Constante Eletromotiva	Ke	0.01	0.001	0.1	Current Row	Horizontal
Resistência Elétrica	R	1	0.01	10	-	Horizontal
Indutância	L	0.001	0.0001	0.1	Current Row	Horizontal

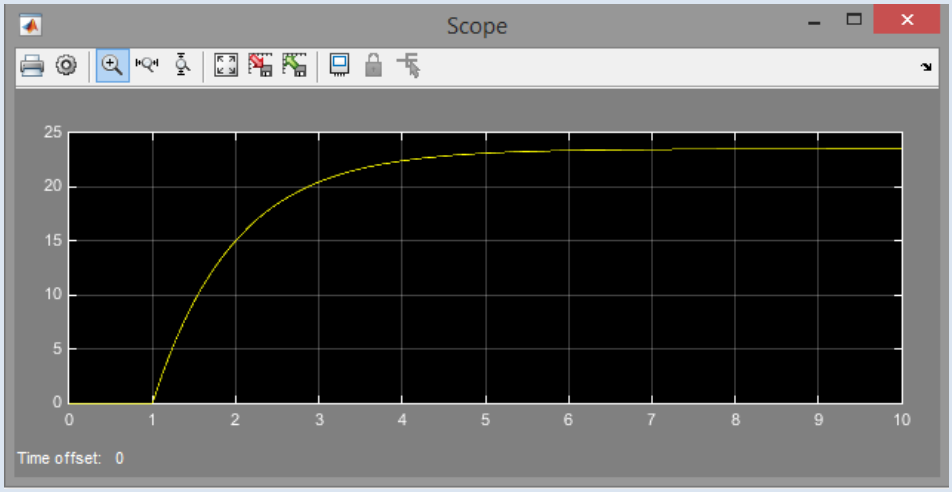
Depois de inseridos os parâmetros, a máscara do nosso modelo fica com o seguinte aspeto:



Vamos agora analisar a resposta do motor a um degrau de tensão. Ou seja vamos fazer variar instantaneamente a tensão aplicada entre 0 e 12 V e vamos registar a velocidade do rotor em função do tempo. Para isso complemente o modelo anterior com os objetos "Step" e "Scope" conforme se mostra em baixo. Em seguida pressione o botão 



O resultado da simulação deve ser apresentado sob a forma de um gráfico com o seguinte aspeto:



Exercício Flash...

Sabendo que a velocidade angular é a derivada em ordem ao tempo da posição angular, modifique o modelo anterior de modo a que o utilizador tenha acesso, não só à velocidade, mas também à posição do rotor.

Pretende-se agora controlar a velocidade do motor de acordo com os seguintes requisitos:

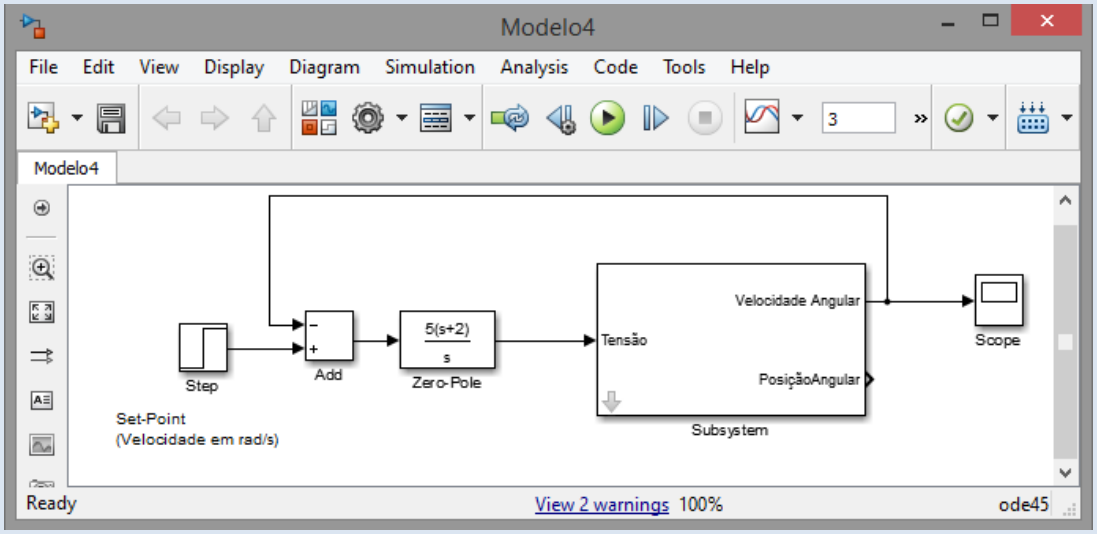
- Erro nulo em regime permanente;
- Sobre-elevação inferior a 5%;
- Tempo de estabelecimento em torno de 1 segundo.

Para isso projetou-se o seguinte compensador proporcional-integral (PI):

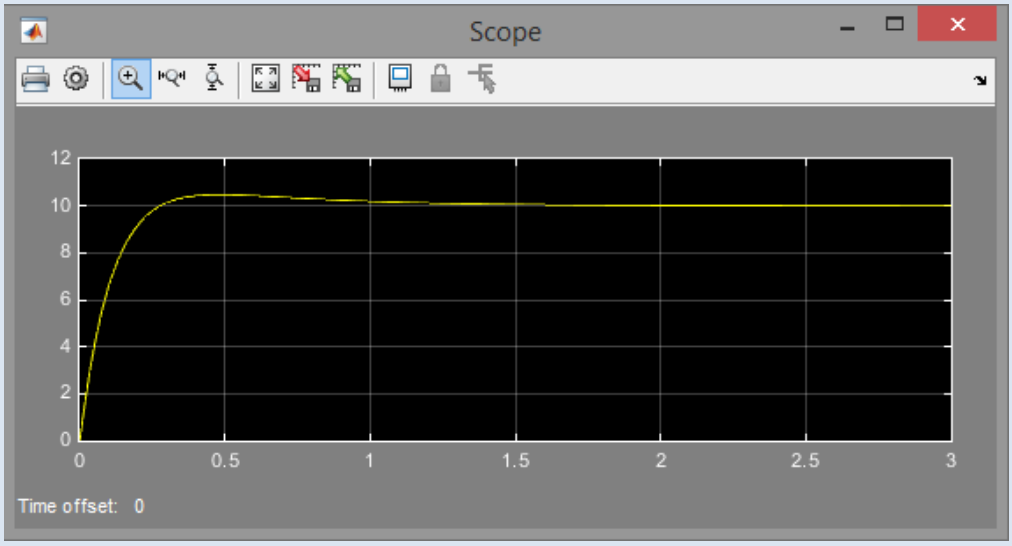
$$G(s) = \frac{5(s+2)}{s}$$

Onde se refere à variável complexa de Laplace. No Simulink as funções de transferência são introduzidas no modelo bastando escolher, na paleta "Continous", o bloco "Transfer Fcn" ou

"Zero-Pole". Utilizando este último procedemos à criação do seguinte sistema em malha fechada:



A simulação fornece o seguinte resultado:



Por inspeção visual confirma-se que os critérios de projeto são atingidos.

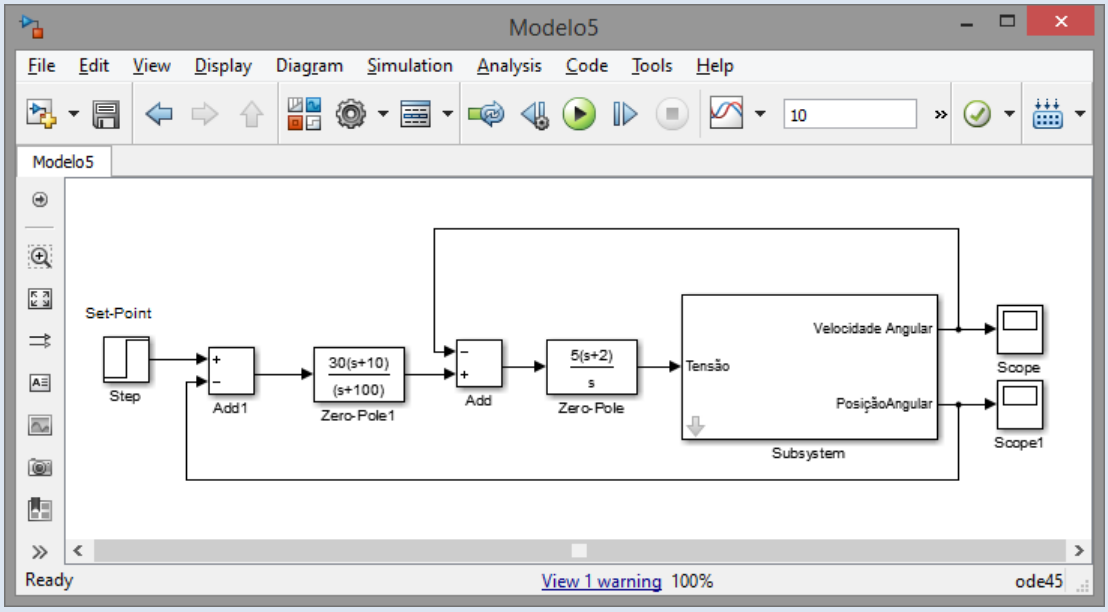
Em muitas aplicações de engenharia, nomeadamente na robótica, é necessário controlar o posicionamento de juntas. De modo a garantir que a posição pretendida é atingida independentemente de perturbações o motor DC que temos vindo a utilizar não pode ser utilizado em malha aberta. Assim desenha-se um controlador que garanta as seguintes especificações:

- Erro nulo em regime permanente;
- Sem sobre-elevação;
- Tempo de resposta inferior a 2 segundos;

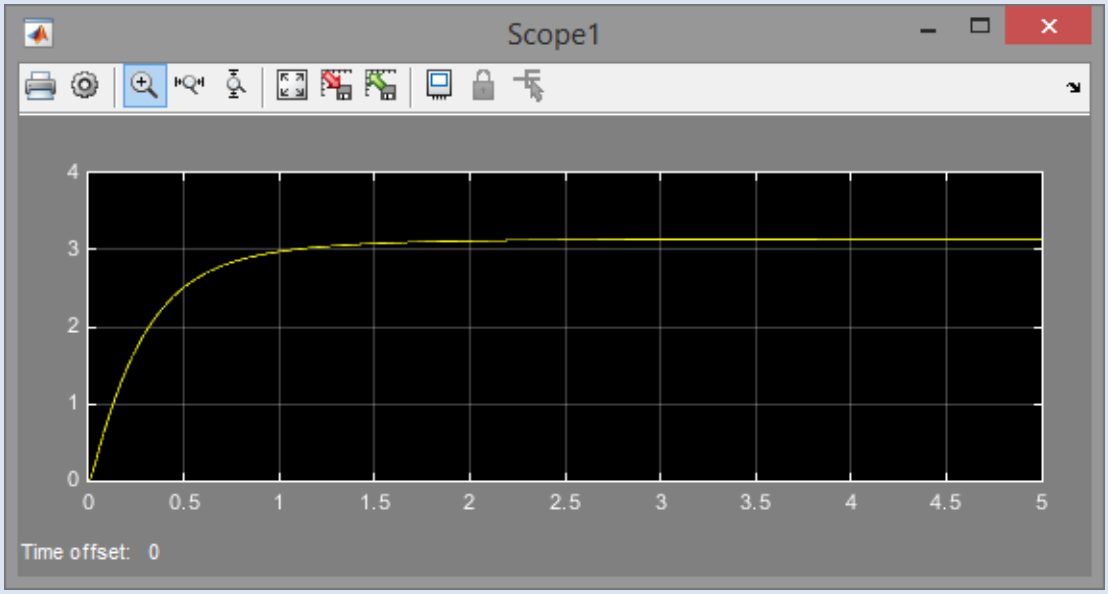
Para isso desenhou-se um compensador avanço de fase com a seguinte função de transferência:

$$G(s) = \frac{30(s+10)}{s+100}$$

O modelo é então expandido de modo a incluir esta segunda malha de controlo mais externa:



O resultado da simulação a um degrau (de posição igual a π radianos) é apresentado a seguir:



Exercício Flash...

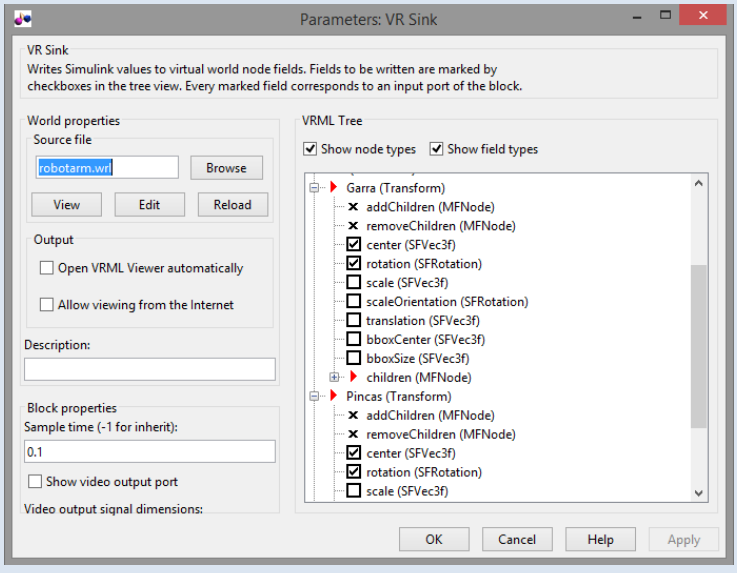
Coloque o pólo deste último controlador uma década atrás do zero e repita a simulação. Agora temos um controlador atraso de fase. O que pode concluir?

De modo a tornar a simulação mais próxima da aplicação real o Simulink permite a integração de modelos dinâmicos como representações 3D do processo físico a simular. De forma a ilustrar este conceito vamos interligar o servomotor realizado anteriormente com o modelo 3D de um manipulador robótico. Em particular imagine-se que o servomotor projetado permite a atuação da garra do robot.

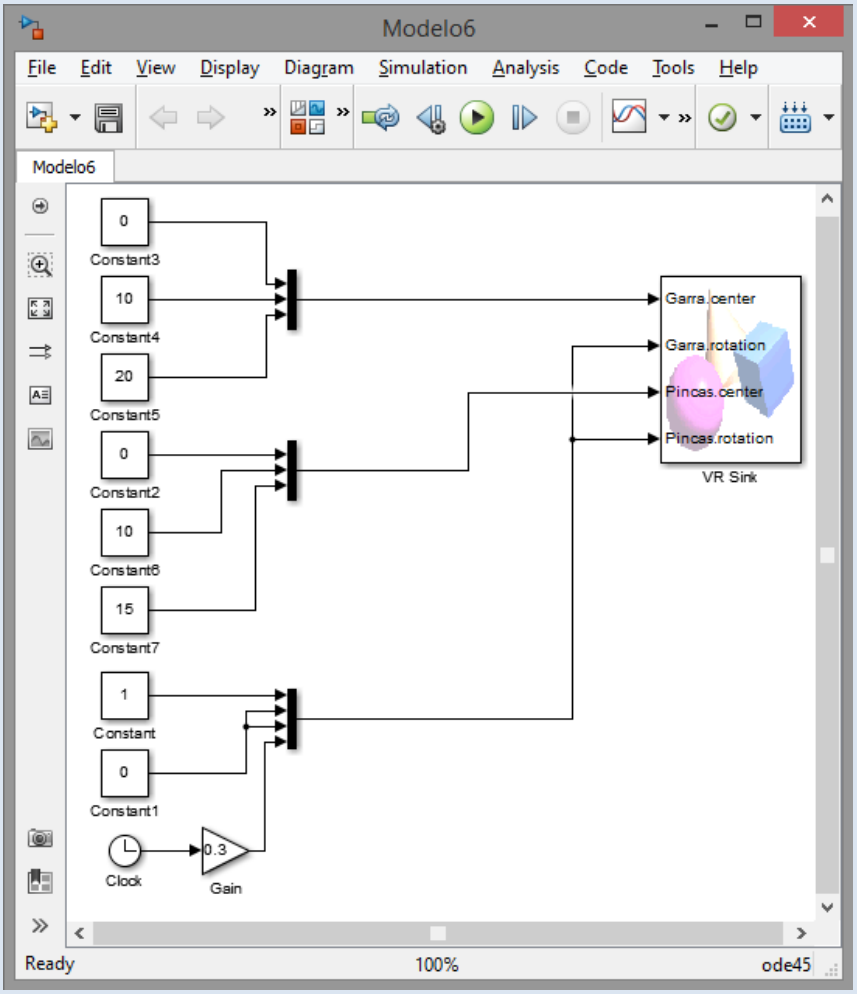
Antes de mais vamos precisar do modelo 3D do manipulador em VRML 2.0. O modelo que vamos utilizar deve ser descarregado de http://www.ipb.pt/~jpcoelho/.... .

O modelo original, desenhado para Sketchup, foi desenvolvido por uma terceira parte e encontra-se disponível em:
https://3dwarehouse.sketchup.com/model.html?id=f0b833f948bb46e4d8333eb16342c317
Apenas foi convertido para VRML com ligeiras alterações. Todos os créditos deste modelo vão para o autor original.

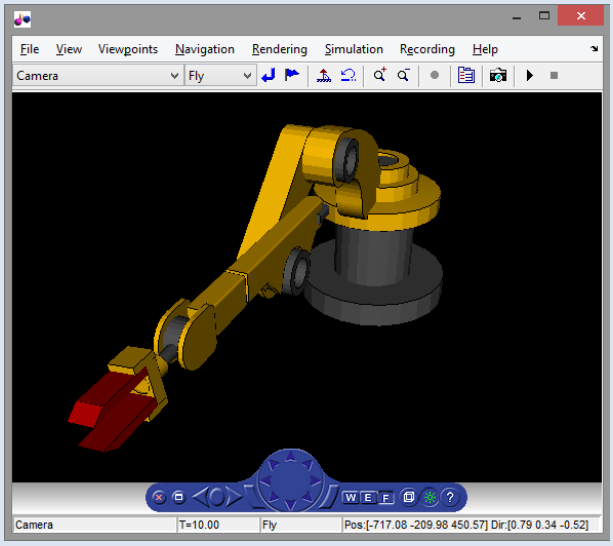
Depois do *download* completo abra um novo modelo no Simulink e introduza o bloco "VR Sink" que pode ser encontrado na paleta "Simulink 3D Animation". Abra o bloco com duplo "click", forneça a localização do ficheiro VRML que descarregou, e selecione as seguintes opções:



Em seguida associe ao bloco "VR Sink" as seguintes ligações:



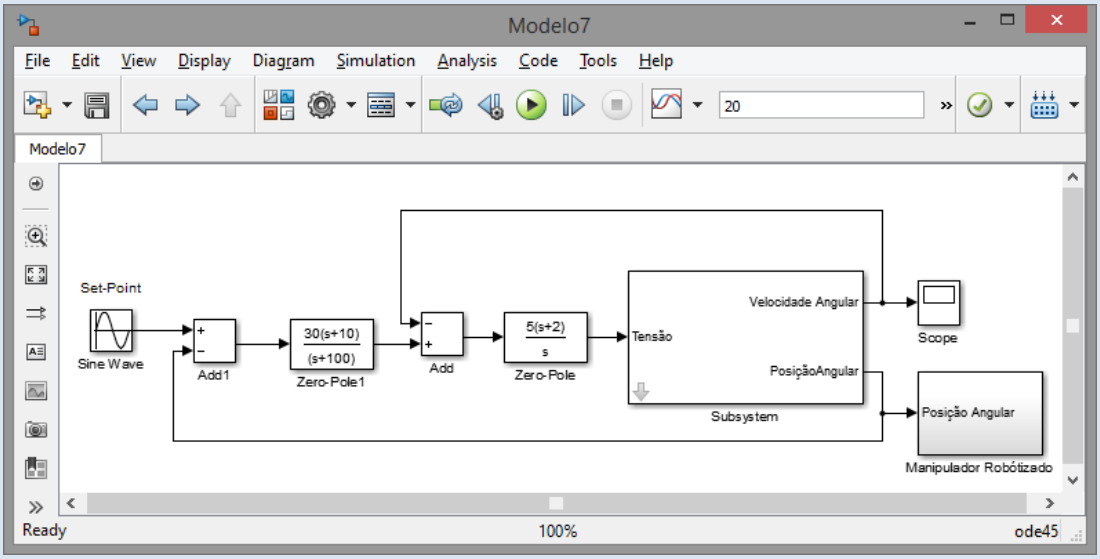
Executando a simulação observa-se o movimento circular da garra numa janela como a que se mostra na figura subsequente:



Exercício Flash...

A partir do modelo de Simulink anterior converta-o num subsistema com uma entrada designada por "Posição Angular".

Para finalizar esta parte vamos assumir que o servomotor será utilizado como atuador para movimentar a garra robótica. A integração destes dois modelos pode ser visto na figura que se segue:



Assuma um tempo de simulação igual a 20 segundos e o sinal de comando como sinusoidal com amplitude igual a π e frequência igual a 0.5 rad/s.

4. Eletrónica de potência: PowerLib

O Simulink também permite a simulação de sistemas eletrónicos. Em particular existe uma *Toolbox* especialmente desenhada para aplicações de eletrónica de potência. De modo a ilustrar esta competência vamos apresentar o exemplo do desenho de um conversor DC/AC que pode ser aplicado, por exemplo, em sistemas de ligação de painéis fotovoltaicos à linha de distribuição da rede elétrica. Um paradigma de interligação designado por "*grid-tie inverter*". Note-se, no entanto, que existem vários pormenores relativos a este tipo de sistemas que não serão modelados. Entre eles se destaca o elemento necessário para garantir que os painéis fotovoltaicos operem no ponto de máxima potência e a etapa de sincronização de fase com a rede de distribuição elétrica. Em concreto pretende-se modelar as primeiras duas partes do seguinte sistema:

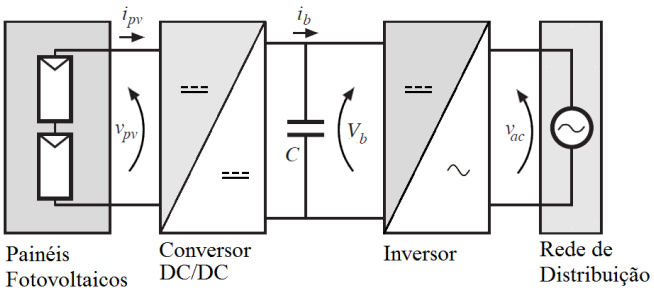


Figura 3. Diagrama simplificado de um sistema solar de ligação à rede.

Modelação dos Painéis Fotovoltaicos

A energia de origem fotovoltaica tem vindo a ocupar uma fatia cada vez maior no panorama das fontes de energia renovável. No presente a Alemanha é o país da Europa com maior produção de energia recorrendo a esta forma. Sendo uma forma de energia ubíqua no nosso planeta existem diversos problemas associados à sua exploração. Por um lado a sua eficiência e por outro a sua variabilidade ao longo do ciclo diário.

O elemento básico de uma célula fotovoltaica é a junção PN. Um modelo frequentemente utilizado na literatura consiste basicamente num díodo em paralelo com uma fonte de corrente:

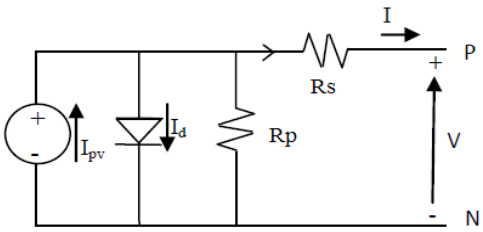


Figura 4. Modelo simplificado de uma célula fotovoltaica.

E a expressão matemática equivalente é:

$$I = I_{PV} - I_0\left(\exp\left(q\frac{V - R_s I}{nKT}\right) - 1\right) - \frac{V - R_s I}{R_p} \tag{1.8}$$

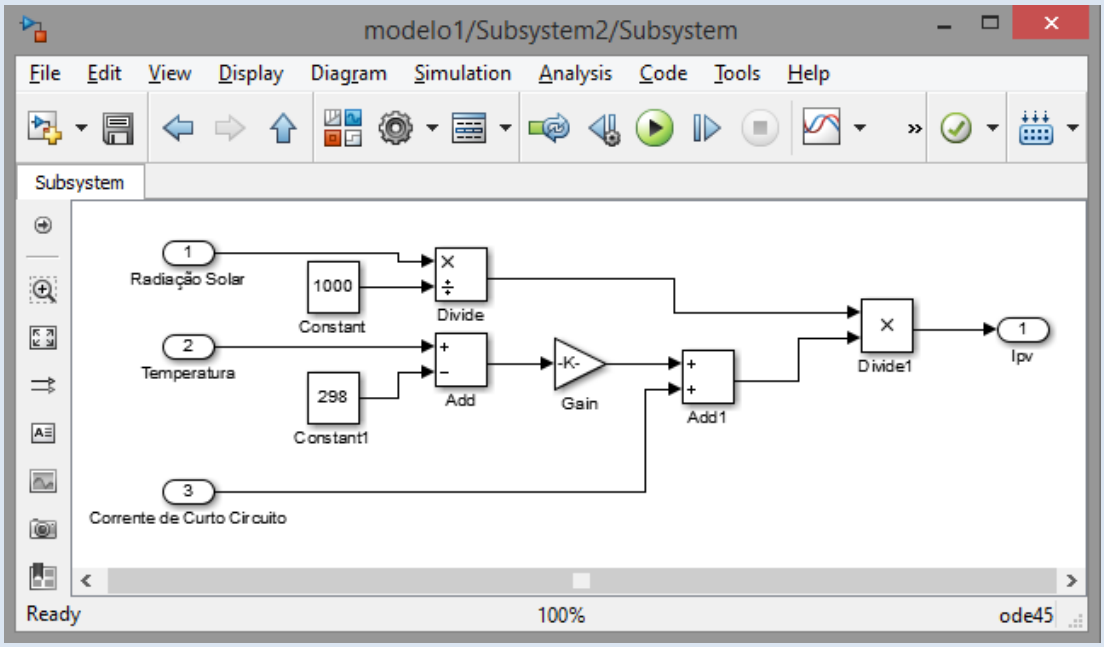
Onde I e V representam a corrente e a tensão da célula fotovoltaica, as resistências R_s e R_p as resistências série e paralela, I_{PV} a corrente de origem fotoelétrica, I_0 é a corrente reversa de saturação, n é um fator de ajuste, K é a constante de Boltzman (1.3807× 10^{-23} m^2 kg s^{-2} K^{-1}) e T é a temperatura absoluta em kelvin.

Por outro lado, a corrente de origem fotovoltaica, I_{PV} pode ser aproximada por:

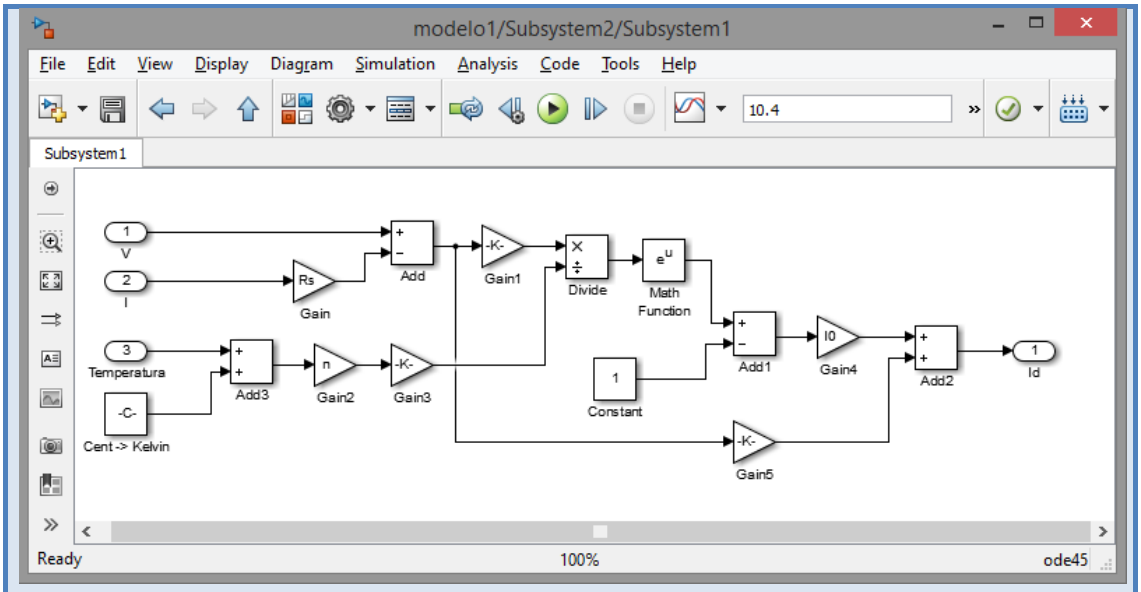
$$I_{PV} = \left[I_{CC} + k_i\left(T - 298\right)\right]\frac{R}{1000} \tag{1.9}$$

Onde $k_i = 0.0017$ A/ºC, R é a irradiância solar em W/m^2 e I_{CC} a corrente de curto-circuito.

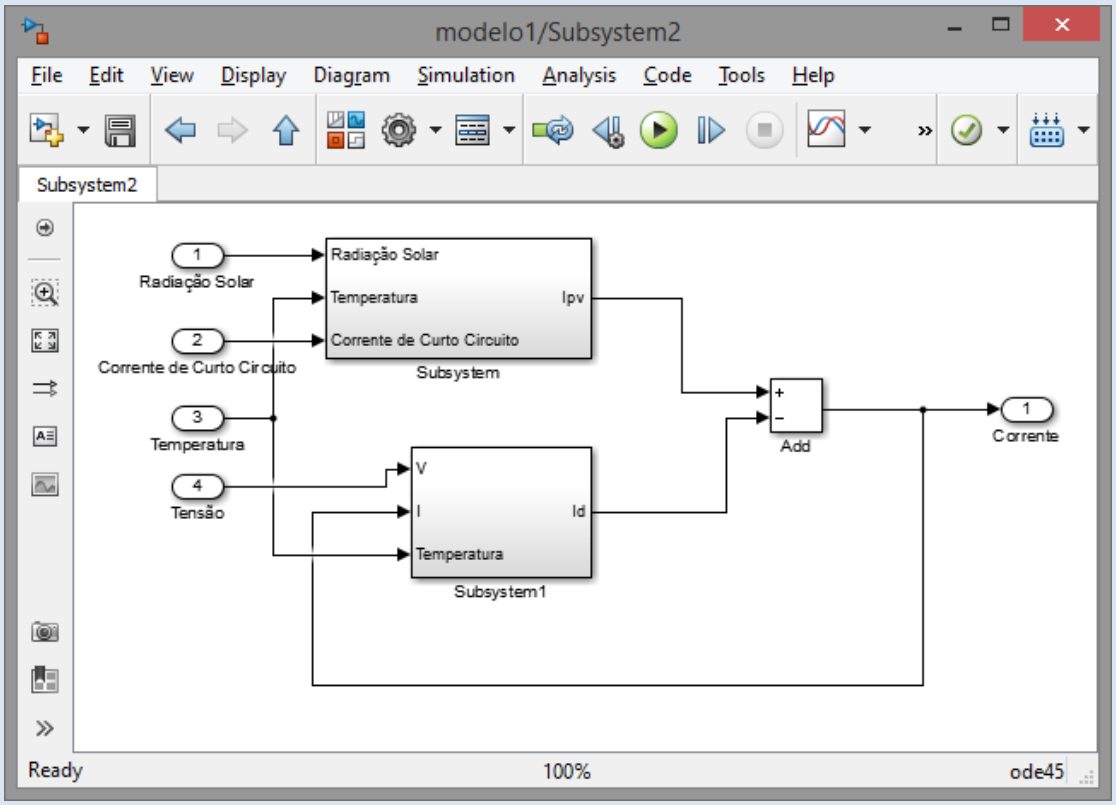
Vamos começar por criar um modelo em Simulink para a equação (1.9). O modelo deve possuir três entradas, T,R e I_{cc}, e a saída I_{PV}.



Agora, para completar a equação (1.8), adicionam-se os restantes termos da expressão conforme se mostra em baixo:

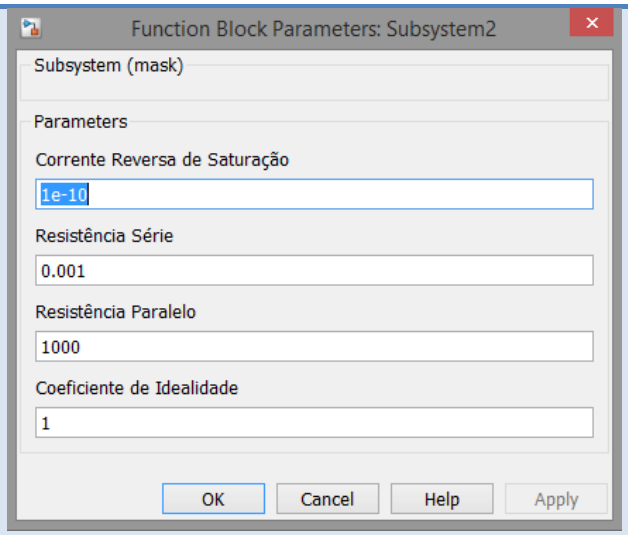


E finalmente...

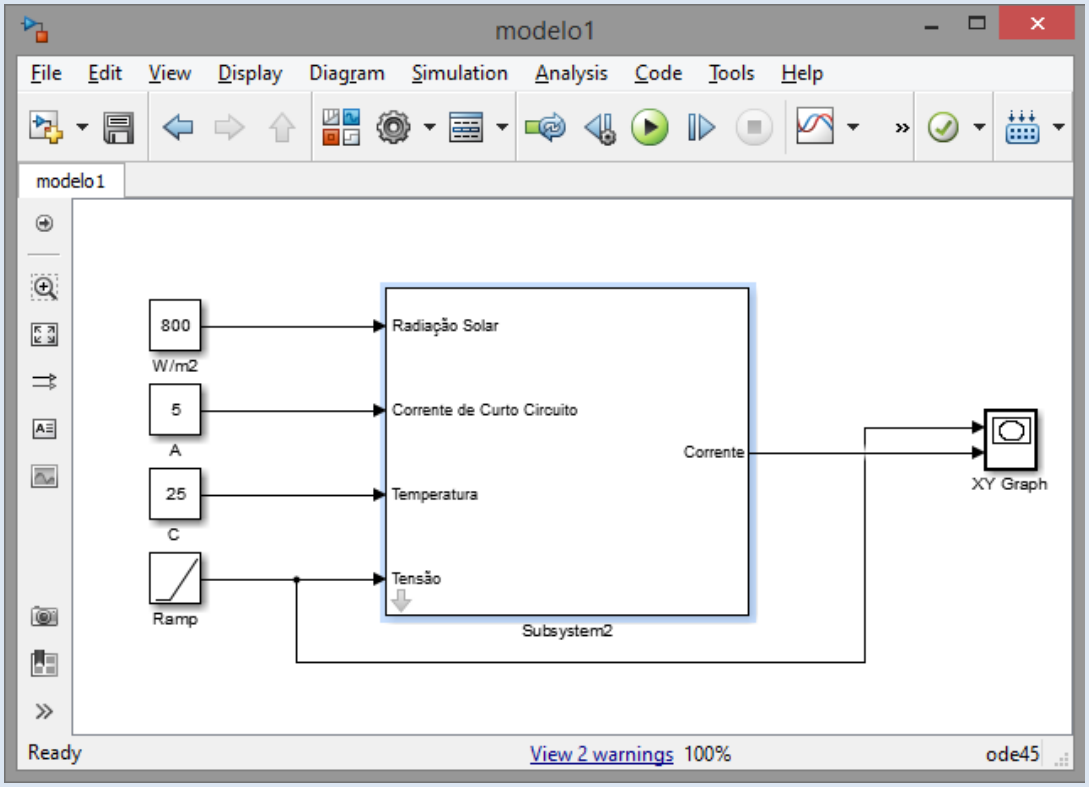


As variáveis K, n, q e Io devem ser definidas, pelo utilizador, acrescentando para isso uma máscara. Os quatro itens devem poder ser modificados através de "edit boxes" onde os valores por defeito devem ser:

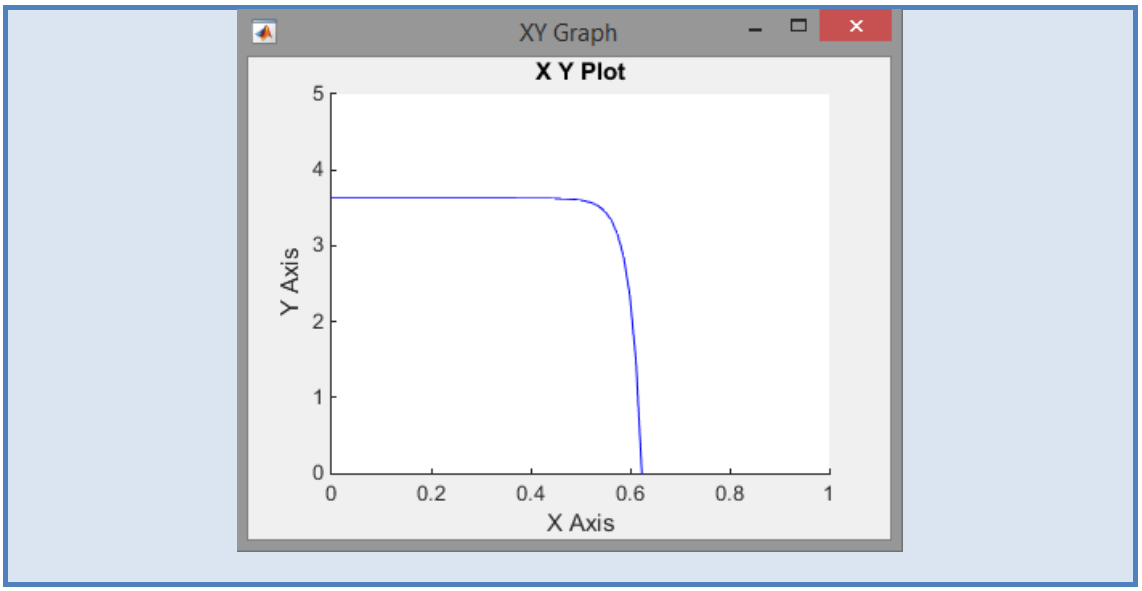
I0 = 1e-10, Rs = 1mΩ, Rp = 1kΩ e n=1



Utilizando o modelo da célula fotovoltaica vamos obter a sua característica V-I considerando uma corrente de curto-circuito de 5A, intensidade de radiação solar igual a 800 W/m^2, temperatura ambiente de 25ºC, rampa com inclinação igual a 0.06 e tempo de simulação de 10.4 segundos. A figura que se segue mostra as ligações que devem ser executadas. Observe que o XY Graph deve ter como limites [0 1] para o eixo das abcissas e [0 5] para o eixo das ordenadas.



O resultado da simulação encontra-se representado em baixo.



Exercício Flash...

A partir do modelo de Simulink anterior obtenha a curva P-V e identifique visualmente o ponto de máxima potência.

Modelação do Conversor Boost

Um conversor de tensão do tipo Boost é um circuito eletrónico, pertencente á família dos conversores DC-DC comutado, capaz de elevar a tensão do sinal aplicado à sua entrada. Esta capacidade de elevar a tensão, com uma eficiência acima dos 90%, faz deste dispositivo um dos blocos fundamentais dos inversores para painéis fotovoltaicos onde a tensão DC fornecida por *strings* de painéis deve ser elevada para valores que permitam posteriormente a sua ligação à rede de distribuição de energia.

O seu princípio de operação transcende o objetivo desta introdução ao Simulink. Apresenta-se apenas o circuito normalmente utilizado na Figura 5.

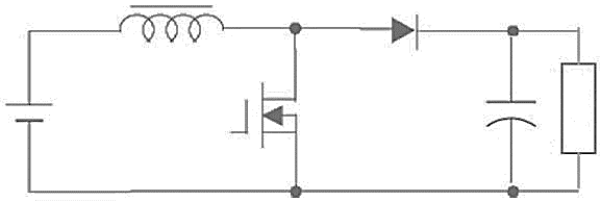


Figura 5. Conversor DC-DC do tipo Boost.

A montante tem-se uma fonte DC que, sem perda de generalidade, pode ser uma string de painéis fotovoltaicos. Assumindo o conversor como ideal, a tensão aplicada à carga da saída é igual a:

$$V_{saída} = \frac{V_{entrada}}{1-D} \tag{1.10}$$

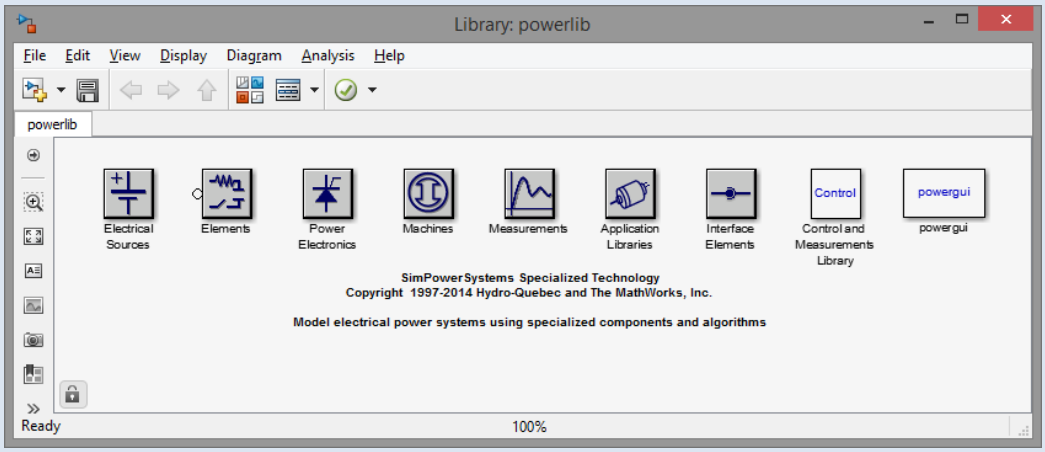
Onde D se refere ao ciclo de trabalho da onda quadrada aplicada à gate do MOSFET.

Admita-se que se possui três painéis fotovoltaicos, ligados em série, cada um com 30V de tensão nominal. Adicionalmente pretende-se elevar a tensão para 320V. A simulação do comportamento dinâmico do conversor pode ser feita em Simulink recorrendo, por exemplo, à Toolbox "PowerLib".

Na linha de comandos do MATLAB vamos escrever:

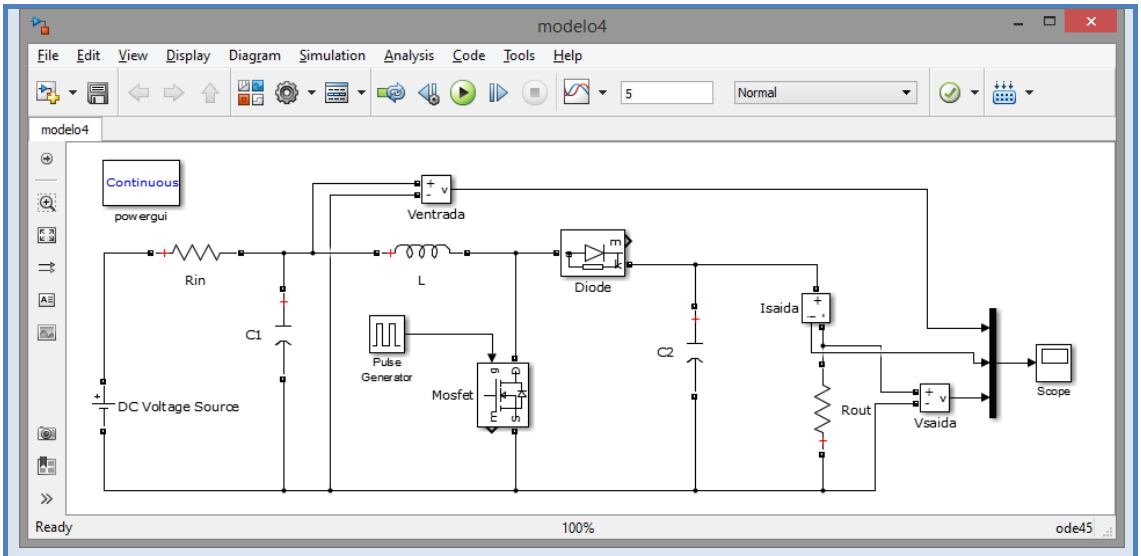
```
>> powerlib
```

Uma janela abre com um conjunto de blocos com operações ligadas à eletrónica de potência.

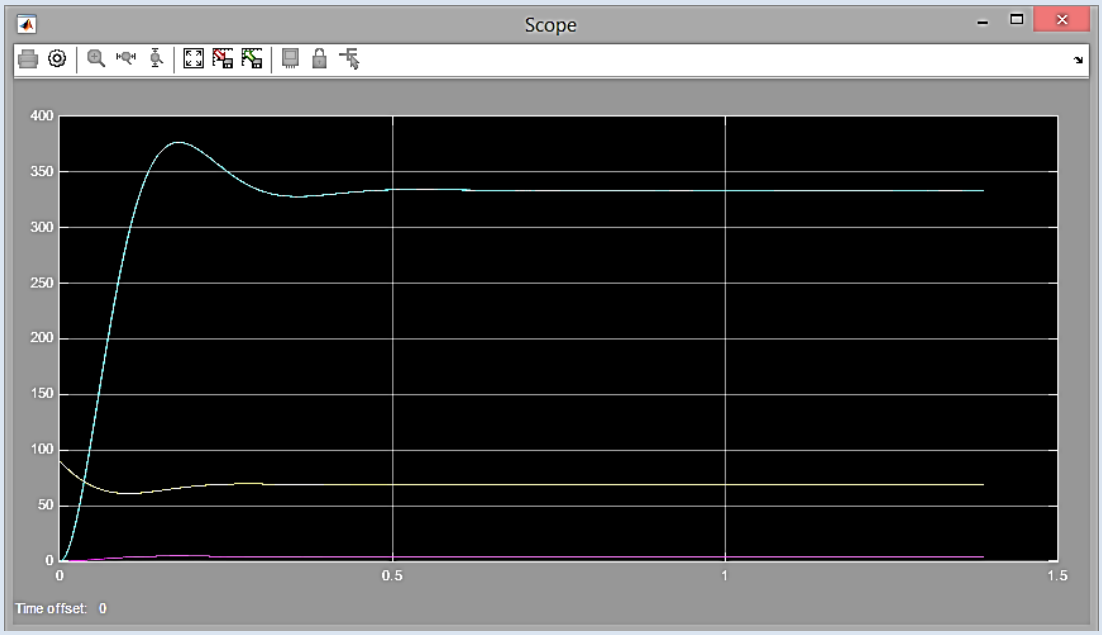


Abrindo um novo modelo constrói-se o circuito da figura em baixo atendendo às seguintes características:

Rin		1Ω
Rout		80Ω
C1		100uF
C2		820uF
L		150mH
Pulse Generator	Amplitude	10V
	T	25us
	Pulse Width	80%



O resultado da simulação encontra-se representado em baixo.



Exercício Flash...

Admita que a tensão dos painéis fotovoltaicos desceu para 45V. Calcule o novo valor do ciclo de trabalho e verifique se a simulação fornece o resultado esperado.

5. Modelos baseados em eventos: SimEvents

Outra das potencialidades do Simulink é a modelação de sistemas baseados em eventos através da biblioteca SimEvents. De um modo sucinto, um sistema baseado em eventos é um sistema cujo estado varia de acordo com a ocorrência de eventos discretos. Imagine por exemplo um modelo de um restaurante. Neste caso, a mudança de estado do sistema só ocorrerá com a existência de eventos como a chegada ou partida de um cliente, a existência de filas de espera à entrada do restaurante, entre outros.

O SimEvents pode ser utilizado para vários tipos de sistemas e para análise de diversas situações. Como exemplos podem referir-se as cadeias de abastecimento, logística, sistemas de comunicação de dados, processos de manufatura, serviços de escalonamento, etc., em que situações como a gestão de recursos, congestão ou atrasos nos processos ou escalonamento de tarefas podem ser estudadas.

Esta biblioteca do Simulink possui diversos blocos, conforme se pode observar na Figura 6. Para uma melhor compreensão das funcionalidades dos blocos, existem alguns conceitos que deverão ser entendidos, nomeadamente:

- Entidades – o que se move ao longo do sistema (por exemplo, pessoas, veículos, pacotes de dados, produtos);
- Atributos – dados numéricos associados às entidades (por exemplo, identidade, capacidade de transporte, tamanho);
- Servidores – onde as atividades são executadas. A execução das atividades é representada em termos de tempo de processamento (por exemplo, tempo de atendimento de um cliente, tempo de deslocação de um veículo);
- Filas – locais onde existe acumulação de entidades;
- *Sinks* – locais de "destruição" das entidades. Representam a saída das entidades do sistema, como por exemplo, a saída do cliente do restaurante ou o armazenamento de um produto após a conclusão da sua produção.

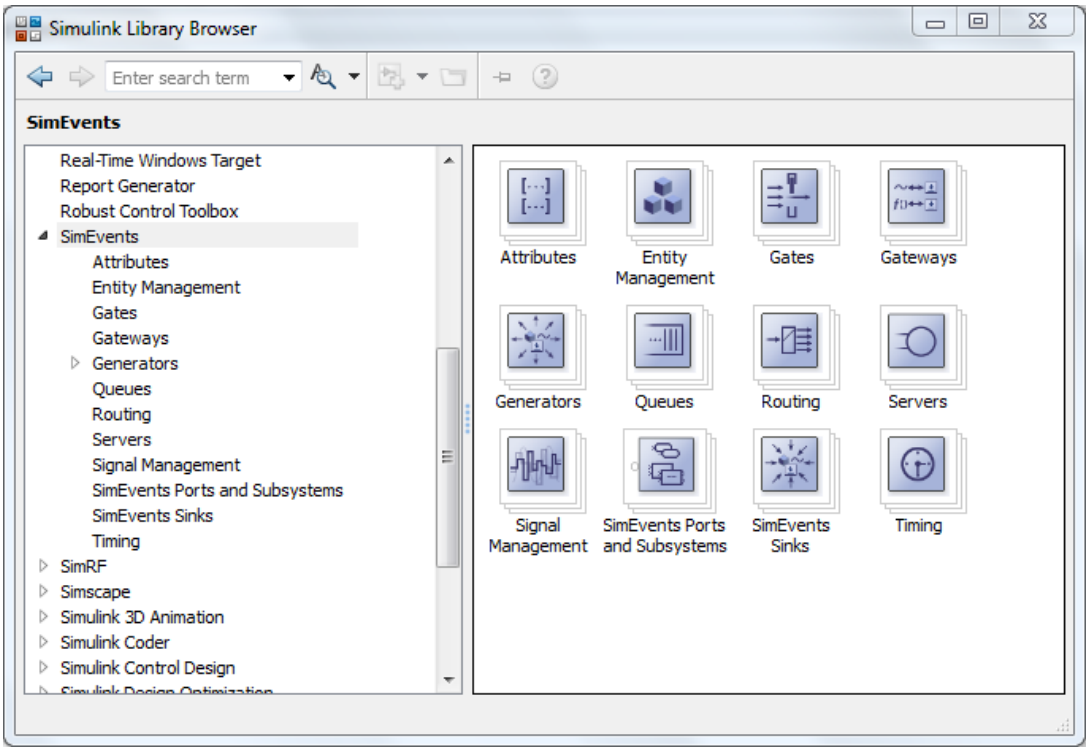
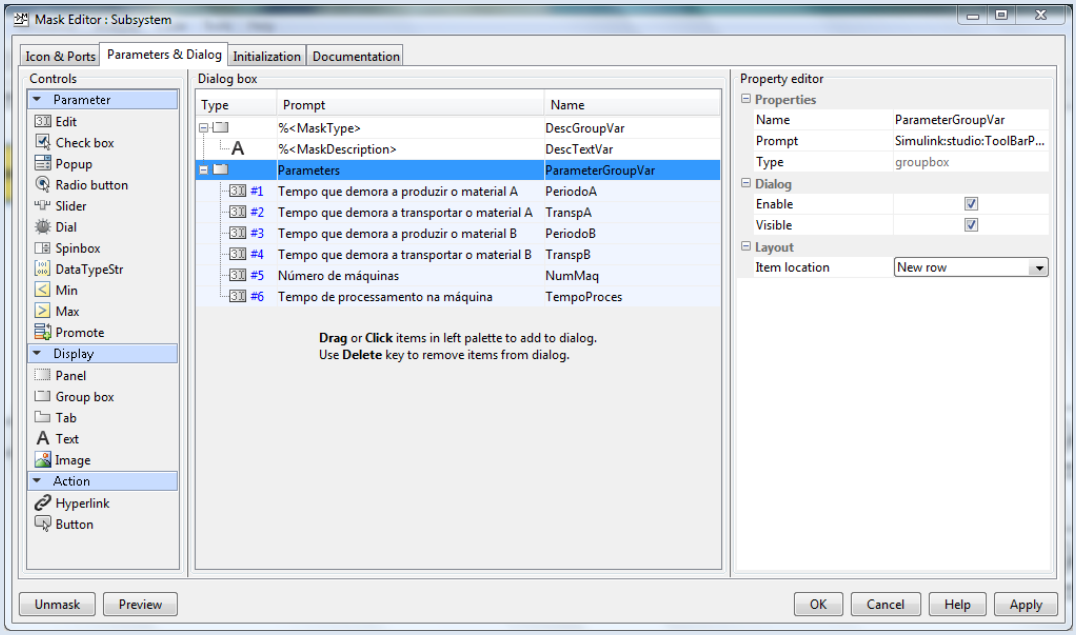


Figura 6. Biblioteca SimEvents.

De modo a compreender melhor esta ferramenta, vamos analisar um exemplo que simula uma linha de montagem para um produto constituído por duas partes e observar o comportamento do mesmo quando sujeito a algumas alterações.

Uma empresa quer alargar a sua produção criando um novo produto formado por duas partes (Partes A e B) que já fabrica na sua instalação. Cada uma das partes tem uma etapa de preparação e tem de ser transportada para a zona de montagem. O novo produto, resultante da junção das duas partes, necessita depois de um equipamento de processamento em que é feita a esterilização, lavagem e secagem. Uma vez acabado o processamento, o produto segue, de forma aleatória, para dois setores (Setores A e B). No entanto, a empresa necessita adquirir o equipamento para executar as tarefas de processamento e pretende saber quantas máquinas necessita para que o novo produto entre imediatamente em processamento e assim aumentar a produção.

Crie um novo modelo para o Simulink. Vamos começar por definir alguns dos parâmetros que vão ser utilizados no modelo. Num novo subsistema crie uma máscara com os seguintes parâmetros do tipo "Edit":



Numa fase inicial vamos atribuir os seguintes valores aos parâmetros:

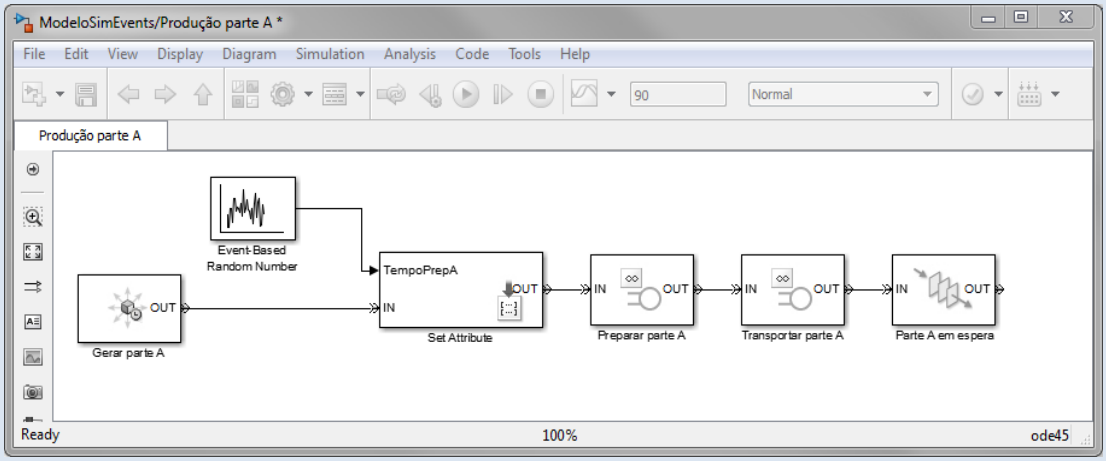
- PeriodoA = 2;
- TranspA = 5;
- PeriodoB = 2;
- TranspB = 5;
- NumMaq = 1;
- TempoProces = 7;

Para além disso na TAB "Initialization" escreva os seguintes comandos:

```
>> assignin('base','PeriodoA',PeriodoA);
>> assignin('base','TranspA',TranspA);
>> assignin('base','PeriodoB',PeriodoB);
>> assignin('base','TranspB',TranspB);
>> assignin('base','NumMaq',NumMaq);
>> assignin('base','TempoProces',TempoProces);
```

Conforme referido no início os modelos baseados em eventos possuem entidades. Neste caso, irão existir 3 tipos de entidades iniciais: as que representam a Parte A, as que representam a Parte B e as que representam as máquinas de processamento.

Vamos de seguida criar a parte do modelo que representa a geração, preparação e transporte da Parte A a utilizar no novo produto:

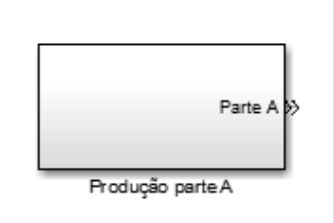


Os blocos utilizados encontram-se descritos na seguinte tabela:

Bloco	Biblioteca	Quantidade	Parâmetros
Time-Based Entity Generator	Generators/Entity Generators	1	Period: PeriodoA
Event-Based Random Number	Generators/ Event-Based Random Number	1	Distribution: Uniform Minimum: 0 Maximum: 2
Set Attribute	Set Attribute	1	Attribute Name: TempoPrepA Value From: Signal port
Infinite Server	Servers	2	Service time from: Attribute Attribute name: TempoPrepA Service time from: Dialog Service time: TranspA
FIFO Queue	Queues	1	Capacity: Inf

Como pode reparar existem algumas entradas/saídas dos blocos da forma “>” e outras da forma “>>”. Isto significa que nas ligações a portas “>” estão associados sinais, enquanto que nas ligações entre portas “>>” circulam entidades.

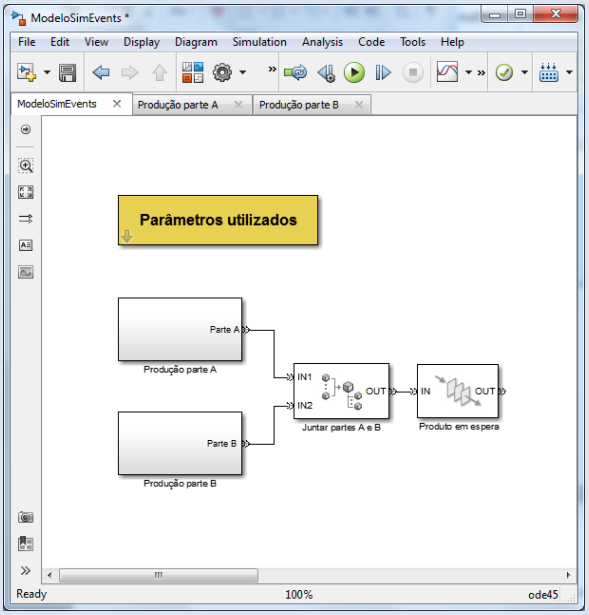
Crie um subsistema com os blocos apresentados que ficará com a seguinte forma:



Exercício Flash...

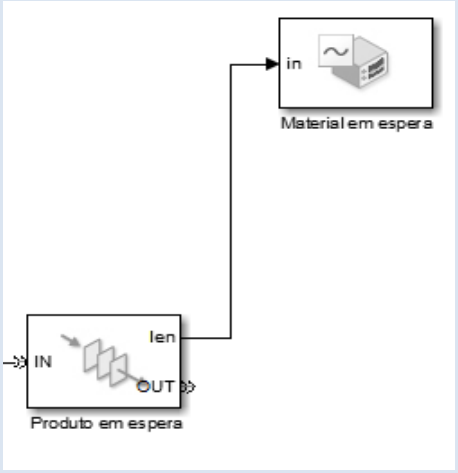
Partindo do subsistema criado para a Parte A, efetue as modificações necessárias para gerar, preparar e transportar a Parte B para o novo produto.

Uma vez modelada a produção das duas partes é necessário proceder à sua junção e colocar o novo produto criado em espera para o posterior processamento. Para tal, adicione os blocos "Entity Combiner" e "FIFO Queue" ao modelo:

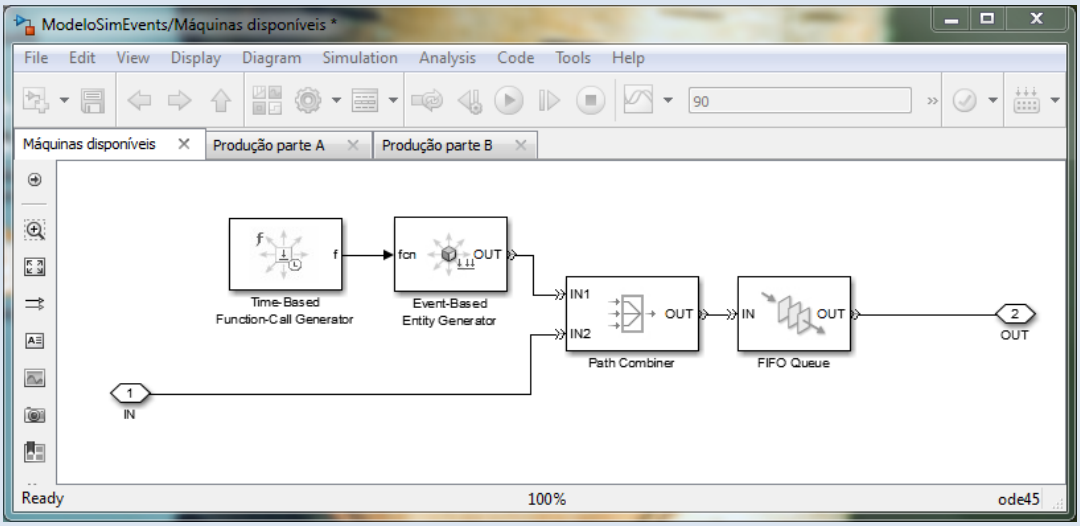


O bloco "Entity Combiner" irá criar uma nova entidade que resulta da combinação das duas entidades anteriores e exige a exitência das duas para que a nova entidade seja criada. Em casos em que posteriormente seja necessário aceder aos atributos das entidades originais a opção "Retain structure in departing entity" do "Entity Combiner" deve estar assinalada.

Sabendo que a empresa pretende que o novo produto entre imediatamente em processamento, é interessante adicionar um "Signal Scope" para verificar a quantidade de material existente, em média, na fila de espera. Assim, no TAB "Statistics" do "FIFO Queue" adicionado marque a opção "Average queue length, len" e ligue a nova saída criada ao "Signal Scope" da seguinte forma:



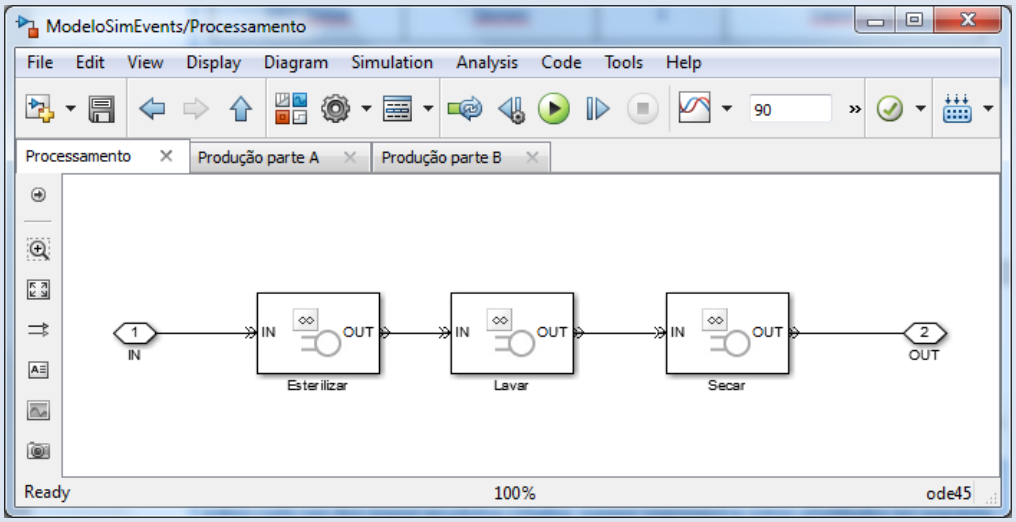
Agora que já temos o novo produto criado é necessário modelar a parte correspondente às máquinas existentes para o processamento. Para tal, adicione o seguinte subsistema com os blocos descritos na tabela abaixo:



Bloco	Biblioteca	Quantidade	Parâmetros

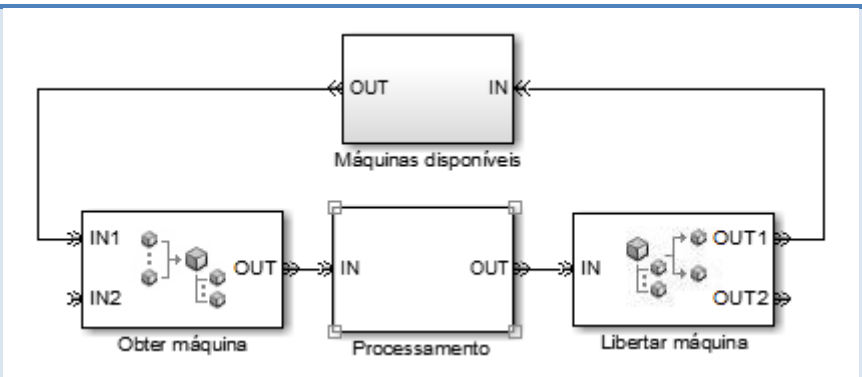
Time-Based Function-Call Generator	Generators/ Function-Call Generators	1	Event generation mode: Only at simulation start Number of simultaneous event repetitions per period: NumMaq
Event-Based Entity Generator	Generators/ Entity Generators	1	Generate entities upon: Function call from port fcn
Path Combiner	Routing	1	Number of entity input ports: 2
FIFO Queue	Queues	1	Capacity: Inf

Sabendo que cada máquina irá efetuar as operações de esterilização, lavagem e secagem sobre cada um dos novos produtos criados, vamos representar estas atividades no seguinte subsistema:



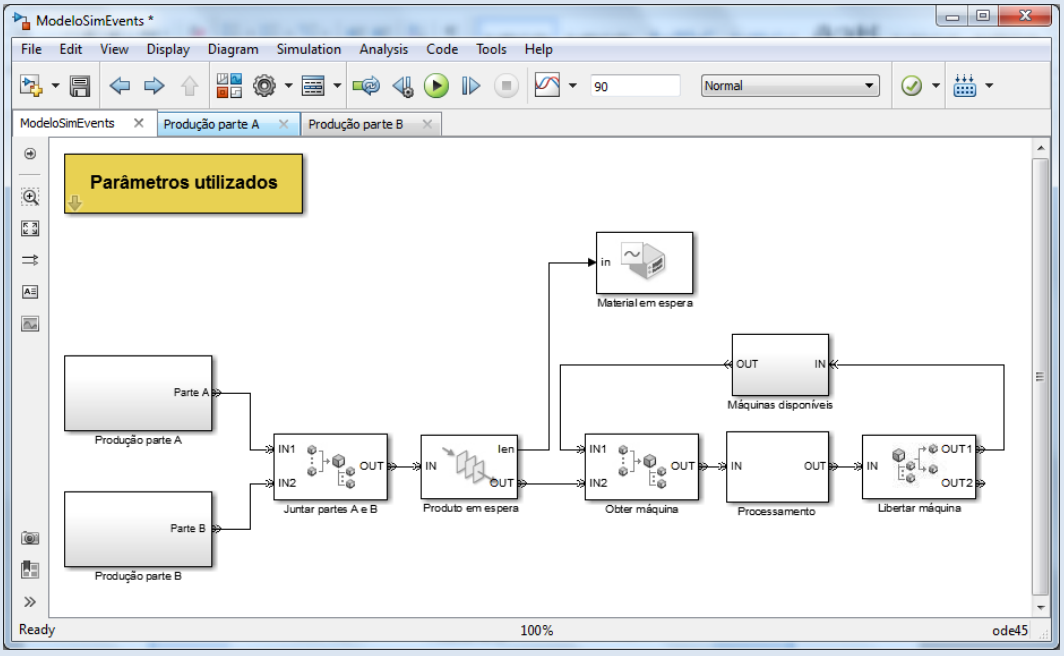
Para simplificar vamos assumir que cada um dos "Infinite Servers" tem um tempo de serviço igual ao parâmetro "TempoProces" definido inicialmente.

Assim sendo, para cada novo produto criado vamos ter que associar uma máquina, executar o processamento e depois libertar a máquina de modo a esta ficar disponível para processar novos produtos criados. Para tal, vamos adicionar os blocos "Entity Combiner" e "Entity Splitter" aos dois subsistemas criados anteriormente:

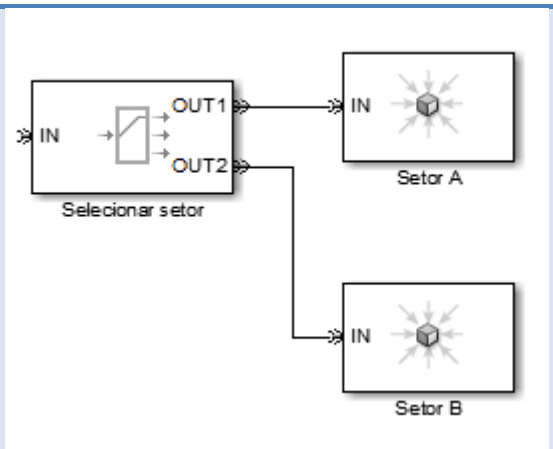


Conforme foi referido anteriormente, uma vez que queremos recuperar a entidade após o processamento com o bloco "Entity Splitter" é fundamental que no bloco "Entity Combiner" a opção "Retain structure in departing entity" esteja assinalada. Caso contrário irá aparecer um erro aquando da simulação.

Até esta fase, o nosso sistema tem a seguinte configuração:



Por fim, sabemos que o produto após processado irá ser distribuído de forma aleatória por dois setores. Para tal vamos utilizar um "Output Switch" para efetuar a decisão de qual o setor para que será enviado o produto e dois "Sinks" que representam os Setores A e B:



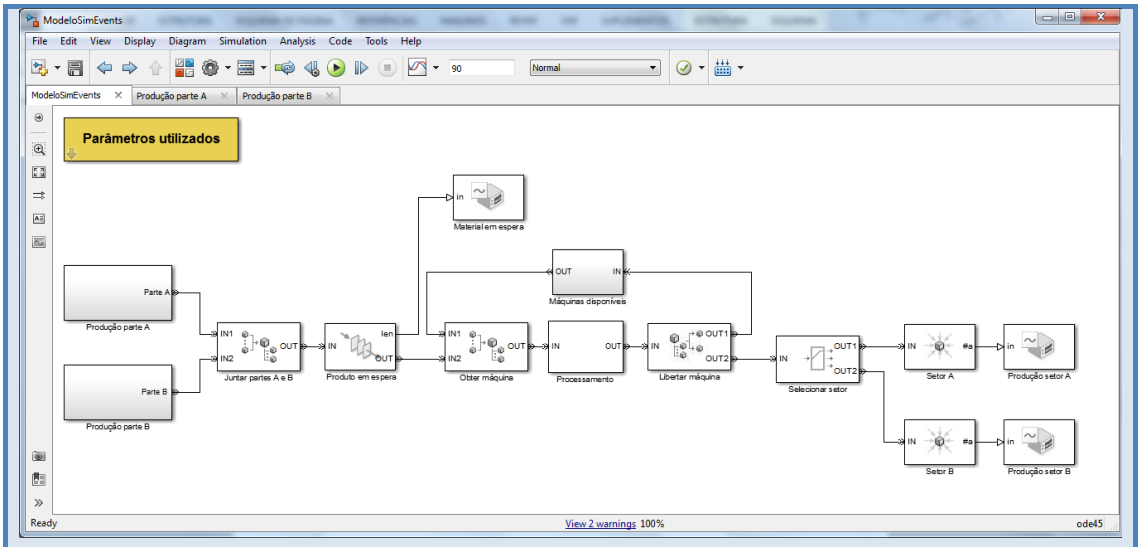
De forma a selecionar aleatoriamente o setor, no "Output Switch" o critério de seleção deverá ser o "Equiprobable".

Exercício Flash...

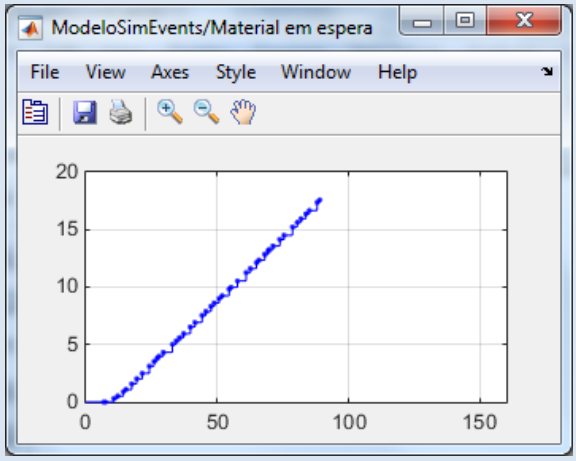
Efetue as alterações necessárias para que a seleção do bloco "Output Switch" seja efetuada com base num atributo associado à entidade que representa o novo produto criado.

Para sabermos quantos produtos chegaram a cada um dos setores, nos blocos "Sink" vamos assinalar a opção "Report number of entities arrived, #a" e ligar esta nova saída do bloco a um "Signal Scope".

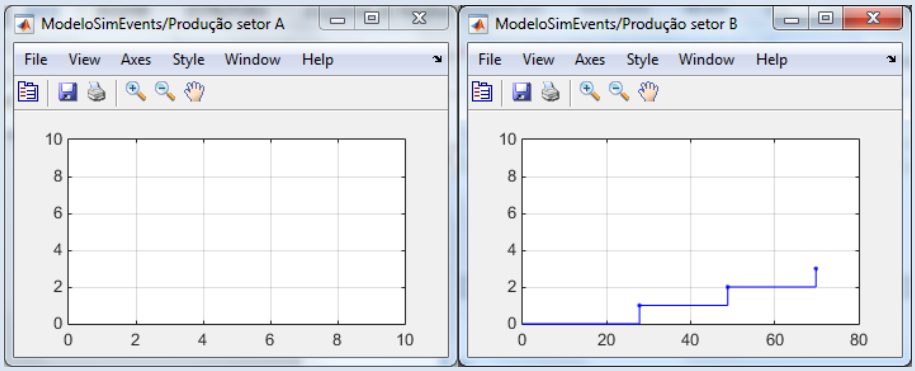
A configuração completa do nosso modelo é a seguinte:



Aumente o tempo de simulação para 90 e simule o modelo. Para a configuração inicial dos parâmetros utilizados, o resultado referente ao material em espera para processamento deverá ter o seguinte aspeto:

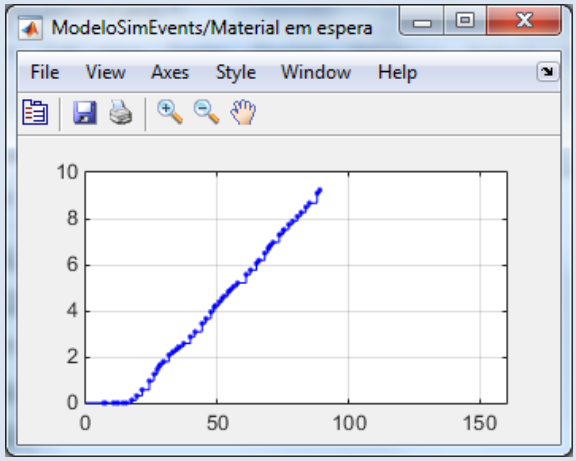


E a produção nos setores A e B poderá ser nula ou muito baixa:



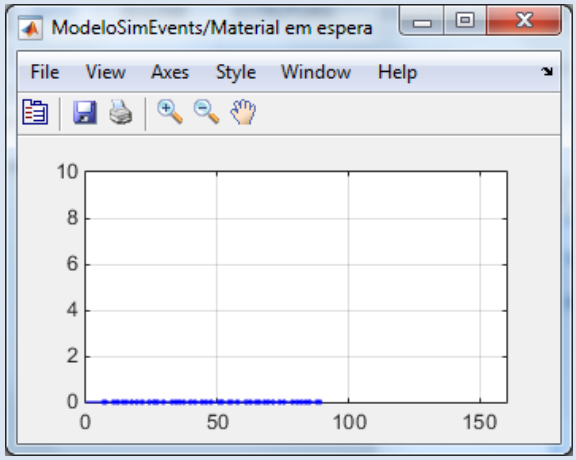
Isto significa que apenas com uma máquina de processamento o novo produto formado fica, em média, um tempo significativo em espera para ser processado e poucos produtos chegam aos setores A e B ao fim de 90 minutos.

Vamos experimentar aumentar o número de máquinas para 5 e observar o resultado. Para este caso, o tamanho médio da fila de espera tem o seguinte aspeto:



Embora o tamanho médio da fila de espera seja já mais baixo, sabemos que o objetivo da empresa é que este valor seja nulo. Se experimentar com mais configurações, irá verificar que são necessárias cerca de 12 máquinas para que tal aconteça.

Relembrando os parâmetros inicialmente definidos, outro dos aspetos que podemos modificar e realizar alguns testes é o tempo de processamento. Vamos agora definir o tempo de processamento igual a 4 e o número de máquinas igual a 7. Neste caso, o tamanho médio da fila de espera do material para o processamento é nulo:



Isto significa que utilizando máquinas um pouco mais rápidas o número necessário é muito inferior.

Importa aqui salientar que neste exemplo foram feitas muitas simplificações e que os sistemas reais são mais complexos. Ainda assim, com este tipo de análise o fabricante poderá fazer várias simulações de modo a averiguar qual a configuração que mais de convém antes de efetivamente realizar a compra.

Outros dos pontos interessantes do SimEvents é que é possível interligá-lo com outras *Toolboxes* do MATLAB. Neste exemplo em específico poderia ser utilizada uma *Toolbox* de otimização para encontrar a melhor configuração para a empresa segundo os objetivos definidos inicialmente.