Wydział Elektryczny
Zespół Automatyki (ZTMAiPC)
ZERiA

LABORATORIUM MODELOWANIA I SYMULACJI

Ćwiczenie 6

Wykorzystanie nakładki SIMULINK do budowy i symulacji modeli dynamicznych.

1. Cel ćwiczenia.

Celem ćwiczenia jest zapoznanie się z nakładką SIMULINK oraz zdobycie praktycznych umiejętności tworzenia i symulowania modeli z wykorzystaniem tej nakładki.

2. Wstęp teoretyczny.

Biblioteka *Simulink* dołączana do pakietu Matlab jest graficznie zorientowanym środowiskiem projektowym wyposażonym w funkcje:

- Konstrukcji modeli dynamicznych
- Analizy działania modeli dynamicznych przy różnych wymuszeniach
- Prezentacji wyników symulacji

W pełni interaktywne środowisko pracy *Simulink* umożliwia budowę modeli dynamicznych na bazie predefiniowanych bloków funkcjonalnych dołączanych wraz z pakietem. Funkcje edycyjne ułatwiają szybkie tworzenie modeli oraz ich modyfikację. W celu umożliwienia symulacji nakładkę *Simulink* wyposażono w zestaw bloków modelujących sygnały wejściowe. Podstawowe to: *step, const, ramp*. Możliwa jest też symulacja dla bardziej złożonych wymuszeń, w tym zdefiniowanych przez użytkownika. Symulacji układów sterowania można dokonywać dla różnych metod całkowania, zadanych parametrów (krok, rząd metody, czas symulacji, *solver* i in.). Prezentacja wyników symulacji w nakładce *Simulink* jest możliwa dzięki bogatej bibliotece bloków wyjściowych. Najprostsze z nich to: *display, scope, to workspace* i in.). Dzięki temu wyniki symulacji mogą być przesłane np. do przestrzeni roboczej Matlab'a i tam poddane dalszemu przetwarzaniu. Możliwości nakładki *Simulink* mogą zostać rozszerzone przez dodatkowe biblioteki bloków funkcjonalnych (*blocksetsi)*. Przykładowe biblioteki to: *Nonlinear Control Design Blockset* – wspomaganie projektowania nieliniowych układów sterowania, *Power System Blockset* – wspomaganie projektowania układów sterowania systemami maszyn i napędów dużych mocy. *DSP Blockset* – wspomaganie projektowania systemów wykorzystujących cyfrowe przetwarzanie sygnałów.

3. Program ćwiczenia.

3.1 Model zależności statycznej.

Przedmiotem modelowania będzie zależność umożliwiającą zamianę wartości temperatur wyrażonych w stopniach Celsjusza na wartości wyrażone w stopniach Fahrenheita:

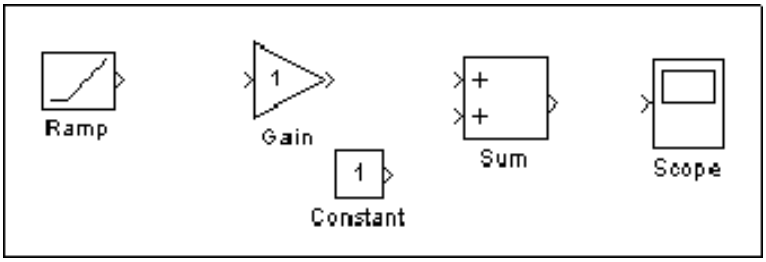
$$T_F = 9/5*(T_C) + 32, \quad (1)$$

Elementy nakładki SIMULINK wykorzystane do budowy modelu:

- Blok Ramp (wymuszenie prędkościowe) z biblioteki *Sources*, (wejście)
- Blok Const (wartość stała) z biblioteki *Sources*, (wartość stała równa 32)
- Blok Gain (wzmocnienie) z biblioteki *Math*, (mnożenie)
- Blok Sum (sumator) z biblioteki *Math*, (dodawanie)
- Blok Scope (oscyloskop) z biblioteki *Sinks*,) (wyniki symulacji)

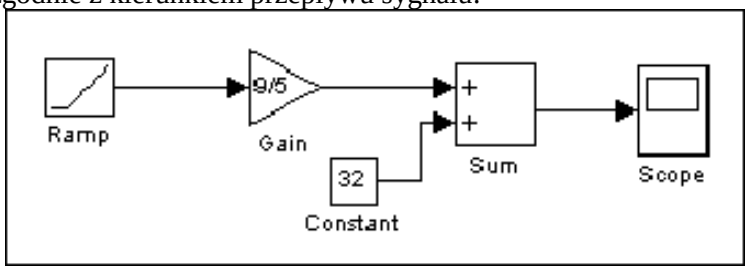
Zadania do wykonania:

a) Przeciągnąć wymienione elementy (bloki) do okna modelu:



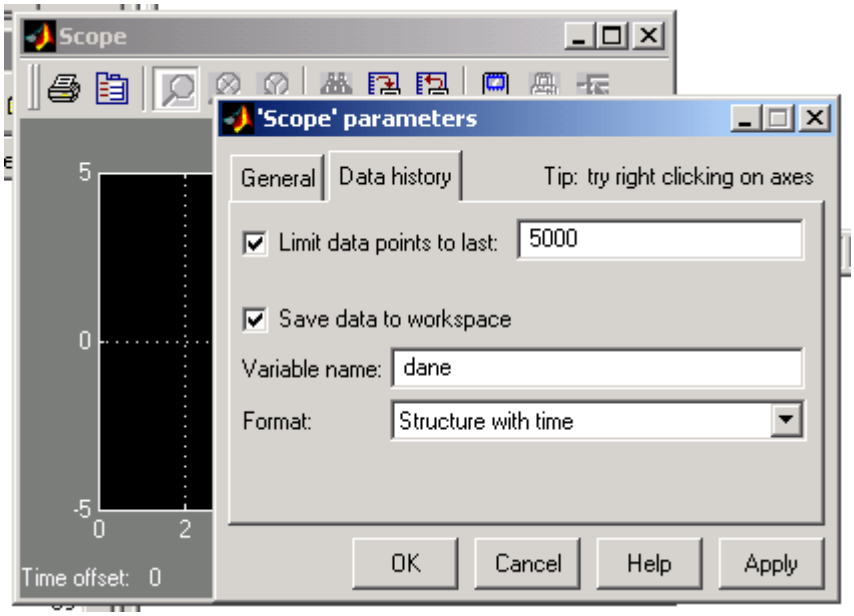
b) Przypisać wartości parametrom bloków. Dwukrotne kliknięcie – edycja wartości parametrów, wprowadzenie wartości, przycisk Close – zamknięcie okna edycji. Wartości parametrów:
Ramp: **Initial Output** = 0
Gain: **9/5**
Constant: **32**

c) Połączyć bloki zgodnie z kierunkiem przepływu sygnału:



d) Zapisać model w pliku C2F.mdl.

e) W bloku Scope ustawić następujące parametry



f) Zapoznać się z opcjami symulacji nakładki Simulink (menu **Simulation**|**Parameters**).

g) Zasymulować działanie modelu (polecenie **Start** z menu **Simulation)** dla czasu symulacji 10s.

h) Zasymulować działanie modelu (polecenie **Start** z menu **Simulation)** dla czasu symulacji 50s.

i) Z każdej symulacji skopiować charakterystykę otrzymaną w wyniku wykonania polecenia:

```
>> plot(dane.time, dane.signals.values)
>> title('Przelicznik stopni Celsjusza na Fahrenheita')
```

3.2 Model równania różniczkowego

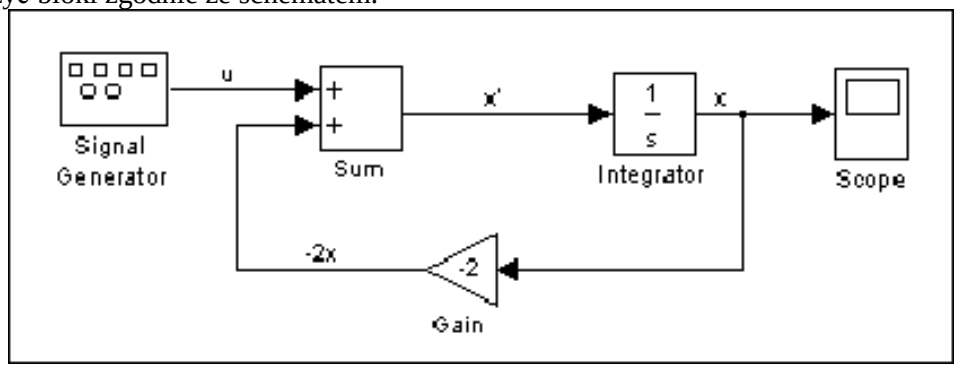
Przedmiotem modelowania będzie równanie różniczkowe postaci:

$$x'(t) = -2*x(t) + u(t), \tag{2}$$

gdzie u(t) jest falą prostokątną o amplitudzie równej 1 i częstotliwości równej 1 rad/sec.
W modelu równania do wyznaczenia x(t) na podstawie x'(t) wykorzystano blok integratora. Inne niezbędne bloki to Gain (mnożenie) oraz Sum (sumator). Dodatkowo zastosowano blok Signal Generator (biblioteka *Sources*) do wygenerowania zadanego przebiegu funkcji u(t).

Zadania do wykonania:

a) Przeciągnąć bloki do okna modelu.
b) Połączyć bloki zgodnie ze schematem:



Węzeł zaczepowy tworzy się przez przeciąganie linii z wciśniętym prawym przyciskiem myszy. W celu odwrócenia bloku Gain należy wywołać menu kontekstowe (klikając prawym przyciskiem myszy na bloku) i wykonać polecenie **Flip Block**. Zapoznać się z opcja mi menu kontekstowego.

c) Zapisać model w pliku rr.mdl.
d) Przeprowadzić symulację działania stworzonego modelu dla dwóch różnych metod całkowania – jednokrokowej (Fixed-step) i wielokrokowej (Variable-step) (metody zmienia się w menu **Simulations**|**Parameters**|**Solver options**).
e) Z każdej symulacji skopiować otrzymaną charakterystykę (podobnie jak w pkt. 3.1i).

3.3 Postać transmitancyjna.

Zakładając zerowe warunki początkowe, do równania (2) stosuje się obustronne przekształcenie Laplace'a, co prowadzi do równania:

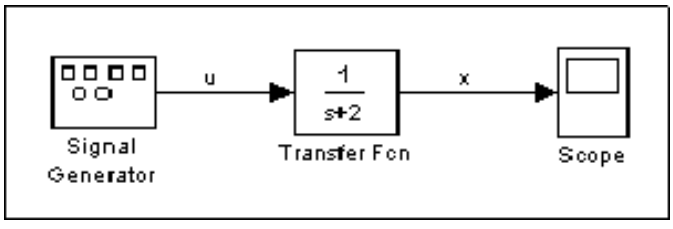
$$sX(s)= -2*X(s)+U(s) \tag{3}$$

Traktując U(s) jako transformatę wymuszenia, oraz X(s) jako transformatę odpowiedzi, po prostych przekształceniach uzyskuje się następującą transmitancję modelu:

$$G(s) = \frac{1}{s+2} \tag{4}$$

Zadania do wykonania:

a) Przeciągnąć do okna modelu bloki: Signal Generator (biblioteka *Sources*), Transfer Fcn, (biblioteka *Continous*)
b) Połączyć bloki zgodnie ze schematem:



c) Ustawić parametry bloku Signal Generator:

Wave Form: square

Amplitude: 1

Frequency: 1

Units: rad/sec

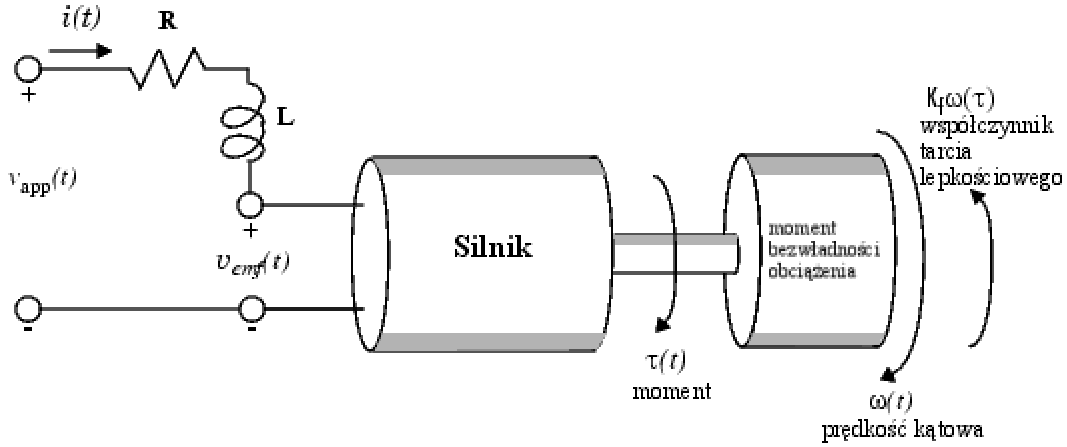
d) Zapisać model w pliku trans.mdl.

e) Przeprowadzić symulację działania stworzonego modelu dla dwóch różnych metod całkowania – jednokrokowej (Fixed-step) i wielokrokowej (Variable-step) (metody zmienia się w menu **Simulations|Parameters|Solver options**).

f) Z każdej symulacji skopiować otrzymaną charakterystykę (podobnie jak w pkt. 3.1i).

3.4 Model silnika prądu stałego.

Na rys. poniżej przedstawiono model silnika prądu stałego wraz z obciążeniem.



Wejściem modelu jest napięcie v_{app}, wyjściem – prędkość kątowa. Sterowanie modelem polega na takiej zmianie napięcia, która wywoła żądaną zmianę prędkości. Model opisują następujące zależności:

1. Moment τ, widziany od strony wału, jest proporcjonalny do prądu indukowanego przez przyłożone napięcie v_{app}:

$$\tau(t) = Km*i(t), \tag{5}$$

gdzie: K_m – stała uzwojenia silnika, związana z jego właściwościami fizycznymi.

2. Indukowana w silniku siła elektromotoryczna v_{emf}, jest proporcjonalna do prędkości kątowej $\omega(t)$. K_b, jest pewną stała zależną również od własności silnika.

$$V_{emf}(t) = K_b*\omega(t) \tag{6}$$

Równania (4,5) są związane z przepływem prądu w uzwojeniach silnika. Stosując dodatkowo prawa mechaniki Newtona, można zapisać równanie równowagi momentów (J – moment bezwładności obciążenia):

$$J\frac{d\omega}{dt} = \sum_i \tau_i = -K_f\,\omega(t) + K_m i(t) \tag{7}$$

Uwzględniając równania (5) i (6) część elektryczną modelu silnika można opisać równaniem:

$$v_{app}(t) - v_{emf}(t) = L\frac{di(t)}{dt} + Ri(t) \tag{8}$$

Uwzględniając (6) równanie przekształca się do postaci:

$$\frac{di(t)}{dt} = -\frac{R}{L}i(t) - \frac{K_b}{L}\omega(t) + \frac{1}{L}v_{app}(t) \tag{9}$$

Przekształcając (7) można wyprowadzić 2 równanie opisujące silnik:

$$\frac{d\omega(t)}{dt} = -\frac{1}{J}K_f\omega(t) + \frac{1}{J}K_m i(t) \tag{10}$$

Równania (9, 10) można przedstawić w postaci macierzowej, jako równania stanu:

$$\frac{d}{dt}\begin{bmatrix} i \\ \omega \end{bmatrix} = \begin{bmatrix} -\frac{R}{L} & -\frac{K_b}{L} \\ \frac{K_m}{J} & -\frac{K_f}{J} \end{bmatrix} \cdot \begin{bmatrix} i \\ \omega \end{bmatrix} + \begin{bmatrix} \frac{1}{L} \\ 0 \end{bmatrix} \cdot v_{app}, \tag{11}$$

oraz równanie wyjścia:

$$y(t) = \begin{bmatrix} 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} i \\ \omega \end{bmatrix} + [0] \cdot v_{app}(t) \tag{12}$$

Zadania do wykonania:

a) W środowisku Matlab utworzyć skrypt ustalający wartości parametrów silnika oraz tworzący jego model w postaci przestrzeni stanów:

```
% Inicjacja wartości parametrów
R= 2.0 % Omów
L= 0.5 % Henrów
Km = .015 % stała momentu
Kb = .015 % stała siły elektromotorycznej
Kf = 0.2 % Nms
J= 0.02 % kg.m^2/s^2

% Konstrukcja modelu w postaci równań stanu
% poprzez zbudowanie macierzy i wywołanie funkcji ss()
A = [-R/L -Kb/L; Km/J -Kf/J]
B = [1/L; 0];
C = [0 1];
D = [0];
dc_ss = ss(A,B,C,D)
```

Skrypt należy zapisać w pliku model_dc.m.

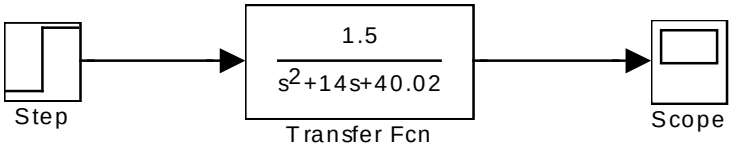
b) Uruchomić utworzony skrypt poleceniem model_dc.

c) Zmienić postać modelu z równań stanu na transmitancję (wejście – $v_{app}(t)$, wyjście - $\omega(t)$):
`dc_tf = tf(dc_ss).`

d) Wyznaczyć charakterystyki częstotliwościowe (Bodego) modelu:
`bode(dc_tf).`

e) Przejść do nakładki SIMULINK i utworzyć nowy model poleceniem **File|New Model**

f) Korzystając z bloku Transfer Fcn (biblioteka *Continous*), Step (biblioteka *Sources*), oraz Scope (*Sinks*), wyznaczyć odpowiedź czasową modelu transmitancyjnego. Postać transmitancji w bloku Transfer Fcn odpowiada transmitancji wyznaczonej w punkcie a. Układ zamodelować jak na schemacie:



g) Przeciągnąć do okna modelu blok wymuszenia liniowo narastającego Ramp (biblioteka *Sources*).
h) Podłączyć blok Ramp do wejścia modelu.
i) Zapisać utworzony model w pliku dc.mdl
j) Symulację przeprowadzić należy dla czasu 5s.
k) Przeprowadzić symulację dla następujących parametrów bloku Ramp:
   prędkość narastania napięcia (slope = 10 V/s.),
   start time = 0.
   Czas symulacji równy 10s.
l) Z symulacji skopiować otrzymaną charakterystykę (podobnie jak w pkt. 3.1i).