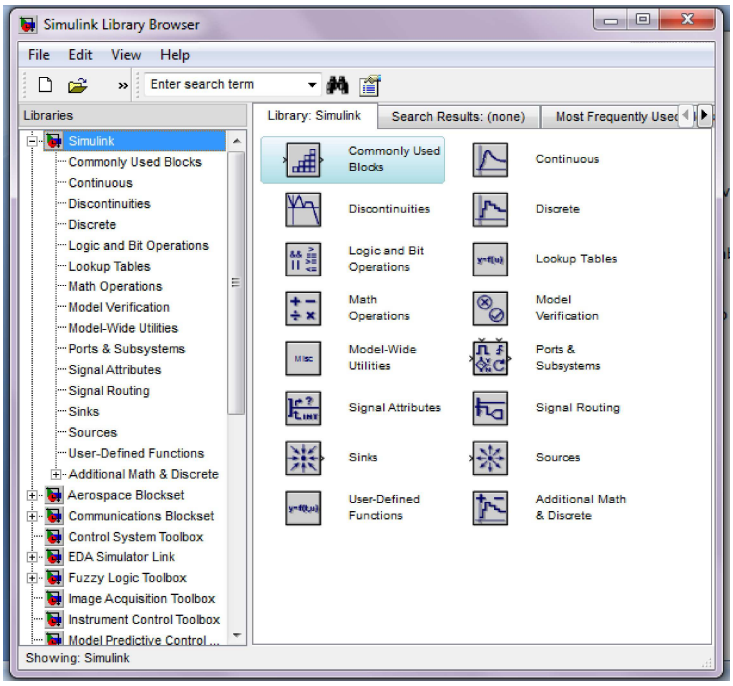
8 Programové prostredie Simulink

Simulink je nadstavba programového prostredia MATLAB, ktorá využíva :

- prácu s blokmi (z vopred definovaných knižníc)
- vyšetruje správanie DS -> je určený na riešenie (simuláciu prechodových dejov v dynamických lineárnych a nelineárnych systémoch)
- predpoklad znalosti matematického popisu dynamických systémov

Simulink používa pre prácu štandardné menu, pomocou ktorého vieme vytvoriť simulovaný model z blokov, ktoré sú vyberané z knižníc.
Simulink je možné otvoriť len ak máme otvorené programové prostredie MATLAB. Pre aktivovanie je potrebné kliknúť na ikonu  alebo zadať príkaz ***simulink*** do príkazového riadku MATLABu. Po tomto úkone sa nám otvorí samostatné okno (Simulink Library Browser).



Obrázok 8-1 Okno programového prostredia Simulink

(Na ľavom paneli sa nachádzajú knižnice, z ktorých základná sa nazýva Simulink, po rozkliknutí sa nám objavia jednotlivé podknižnice, ktoré sú k dispozícii.)
Programovanie v prostredí Simulink pozostáva:

- výber blokov z knižníc (libraries)
- pripájanie vstupov a výstupov odpovedajúcich blokov (signály modelu)
- zadávanie parametrov blokov
- vytváranie subsystémov

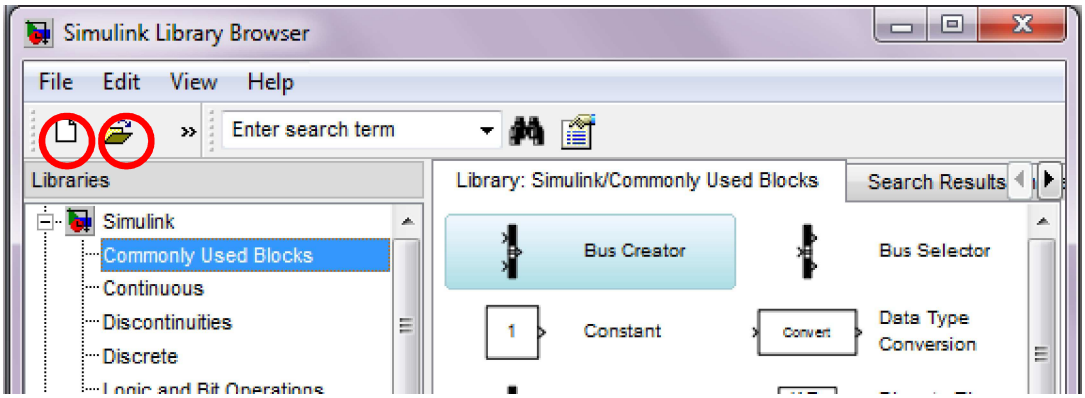
VSTUPNE SIGNÁLY vyberáme:

- z knižnice blokov generujúcich základné typy signálov,
- zo súboru,
- z matíc vopred pripravených v programovom prostredí MATLAB
- z merania v reálnom čase (meracia karta + Real Time Toolbox)

VÝSTUPNÉ SIGNÁLY získavame:

- z blokov typu osciloskop, xy graf ...
- do pracovného priestoru programového prostredia MATLAB
- do súboru

Modely vytvárame po kliknutí na ikonu NEW MODEL alebo OPEN MODEL, ak chceme pokračovať v už začatom modeli. **Modely** sú vytvárané (editované) pomocou myšou riadiacich príkazov -> pre kvalitné a rýchle vytváranie modelu je nutné orientačne poznať všetky typy blokov používaných pre danú triedu systémov.



Obrázok 8-2 Najpoužívanejšie bloky programového prostredia Simulink

Knižnice v Simulink-u:

Sources (zdroje) – generátor vstupov – obsahuje bloky, ktoré nemajú vstupy, pretože predstavujú vstupy vytváraného systému.


Step

blok generujúci skokovú funkciu (až po určitej dobe)

Constant

zdroj konštantnej hodnoty


Clock

zdroj času


Signal Generator

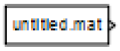
generátor rôznych funkcií, napr. sinus, obdĺžnik, píla, ...


Pulse Generator

blok simulujúci pulzný generátor


Sine Wave

generátor sínusového signálu



From File

blok pre načítanie údajov zo špecifického súboru *.mat

From Workspace

blok pre načítanie údajov z matice v pracovnom priestore

In1

blok pre tvorbu subsystému, vstupný blok

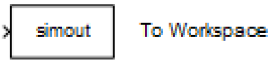

Band-Limited White Noise

aproximácia bieleho šumu (náhodný signál, ktorý ma rovnaký výkon na všetkých frekvenciách)

Sinks (bloky sledovania výstupov) – bloky, ktoré nemajú výstupy. Slúžia k sledovaniu a záznemu zvolených výstupov modelu pri simulačných experimentoch (ďalšie spracovanie)

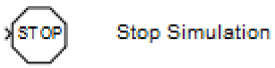
ekvivalent osciloskopu, zobrazenie signálu počas simulácie

numerické zobrazenie hodnôt signálu

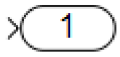


ukladanie simulovaných údajov do pracovného priestoru

ukladanie simulovaných údajov do súboru *.mat



ukončenie výpočtu modelu pri dosiahnutí zvolenej hodnoty



blok používaný pri tvorebe subsystému ako výstupný blok

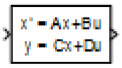
grafické znázornenie signálov t-parametrov

Bloky operácií - bloky, ktoré predstavujú V/V operácie

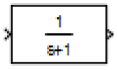
Continuous – obsahuje bloky pre vytváranie spojitých modelov z diferencialnych rovníc

derivačný blok

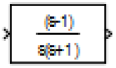
integračný blok



blok pre implementáciu stavového modelu systému



blok pre implementáciu prenosovej funkcie v polynomiálnom tvare



prenosová funkcia v tvare poly/nuly

spojité dopravné oneskorenie

premenlivé dopravné oneskorenie

Discrete – bloky pre vytvorenie diskretnych dynamických modelov

hodnota z minulého integračného kroku

Discontinuities – nespojité systémy, ktorých výstup je nespojitou funkciou vzhľadom na daný vstup (bloky typických nelinearít)

obmedzenie signálu

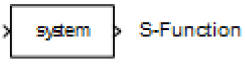
blok modelujúci relé

mrtvá zóna

blok modelujúci hysterézu

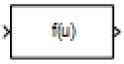
obmedzenie rýchlosti zmeny signálu

User-Defined Functions – používateľom definované funkcie

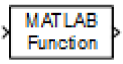


vytváranie vlastnej S-funkcie

príklady vlastných S-funkcií



blok používaný na vytvorenie vlastnej funkcie v programovacom jazyku C



blok odvolavajúci sa na matlabovské funkcie

Math Operations – zápis algoritmickej časti modelu

absolútna hodnota

zosilňovací blok, vynásobenie výstupného signálu konštantou

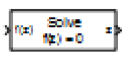
simulačný blok pre 1 ÷ n signálov

odmocnina

súčin vstupných signálov

preddefinovaná matematická funkcia

preddefinovaná trigonometrická funkcia



algoritmus slučky, hodnota signálu pre ktorý je f‘ = 0

Logic and Bit Operations

logická operácia AND

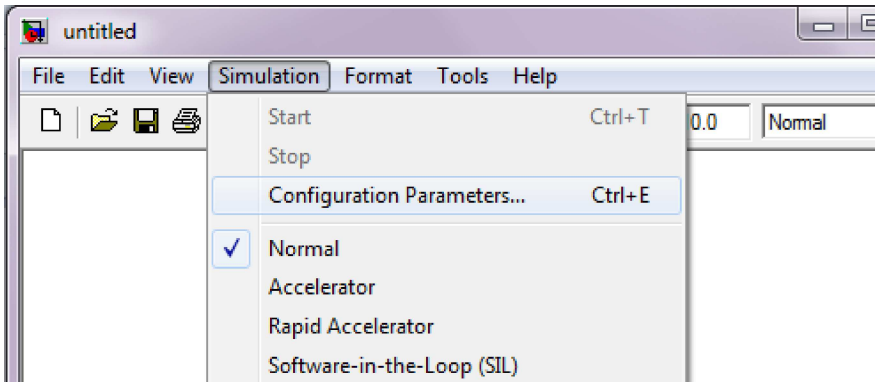
Signal Routing

blok spájajúci niekoľko skalárnych/vektorových signálov na jeden vektorový signál

blok rozkladajúci vektorový signál na niekoľko skalárnych/ vektorových Signálov

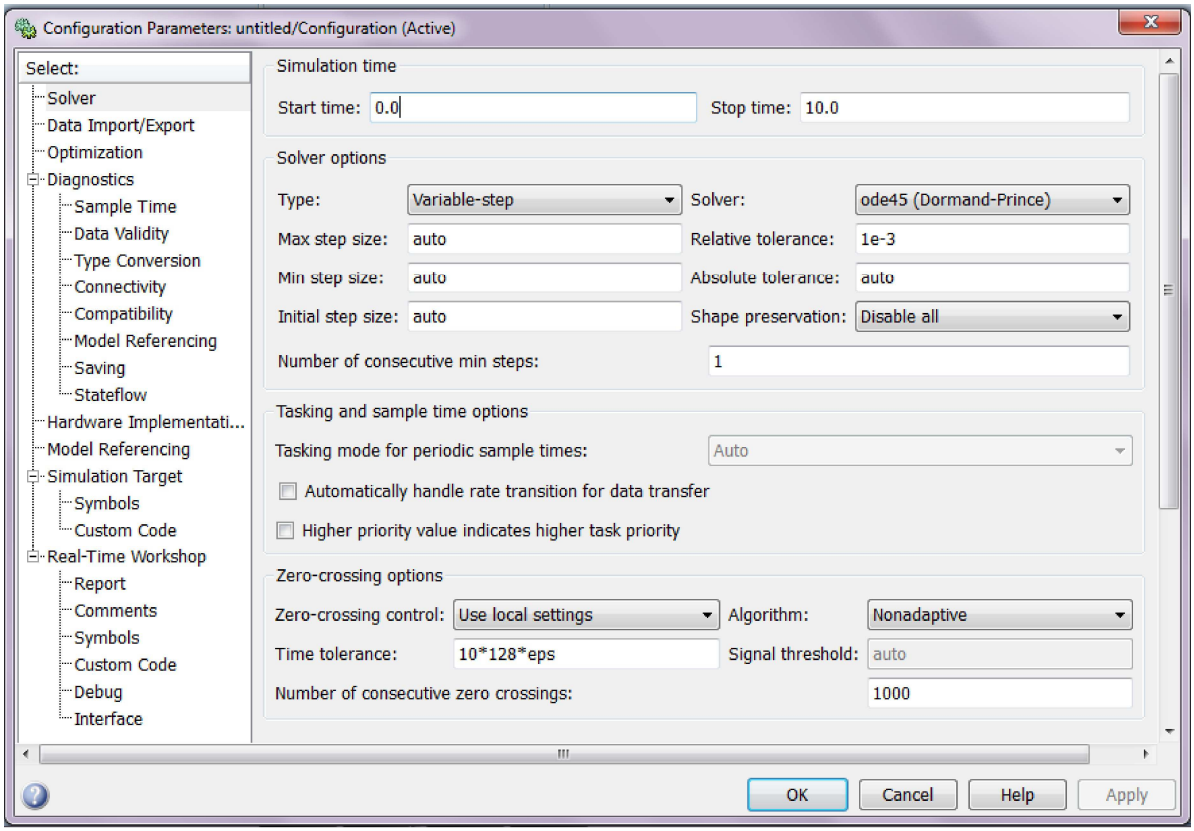
Commonly Used Blocks – najčastejsie používateľom používané bloky
Subsystems – samostatne vyčlenená knižnica. Obsahuje bloky súvisiace s tvorbou subsystému a dovoľujú do simulovaného modelu zahŕňať štandardné programové vybavenie

Pre nastavenie parametrov simulácie vyberieme záložku SIMULATION -> COMFIGURATION PARAMETERS



Obrázok 8-3 Nastavenie parametrov simulácie

alebo použijeme klávesovú skratku ctrl+E, ktorou otvoríme okno pre nastavenie simulácie



Obrázok 8-4 Okno pre nastavenie parametrov simulácie

V položke Solver môžeme nastaviť čas simulácie, voľba metódy riešenia a pod. ...

8.1 Simulácia riešenia LDR v prostredí SIMULINK

Majme LDR 2 rádu : $2y'' + 4y' + 2y = 1(t)$

1. Normovanie LDR

$1y'' + 2y' + 1y = 0{,}5(t)$

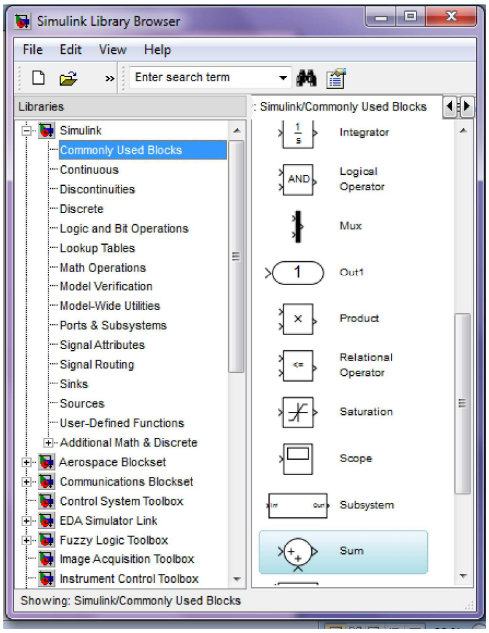
2. Prepis do substitučného kanonického tvaru:

$y' = x_1' = x_2$

$y'' = x_2' = (0{,}5 - 2x_2 - x_1)$

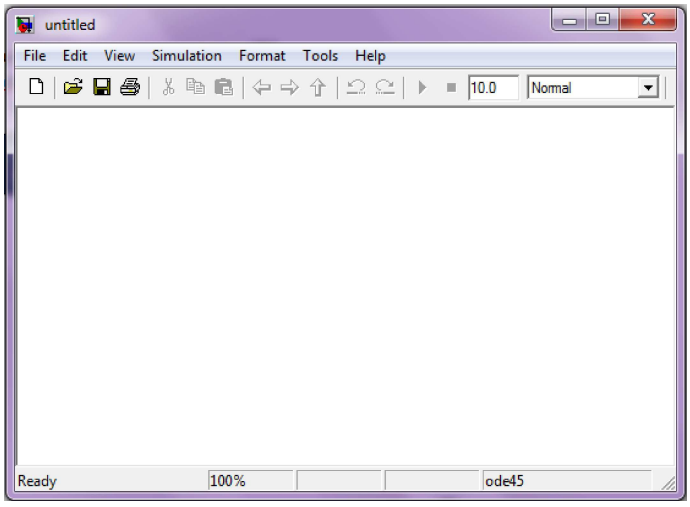
V Simulink-u budeme pracovať iba s $y'' = x_2' = (0{,}5 - 2x_2 - x_1)$

3. Spustíme si grafickú nadstavbu Simulink pomocou ikony  alebo pomocou príkazu ***simulink*** v command window. Otvorí sa následujúce okno:



Obrázok 8-5 Okno programového prostredia Simulink

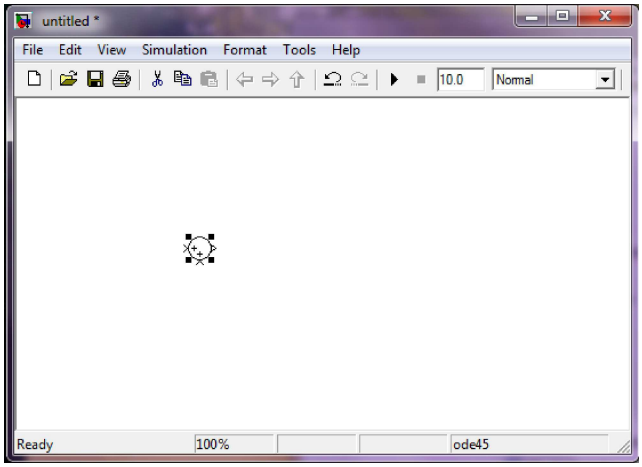
4. Vytvorenie nového podsystému: File -> New -> Model.



Obrázok 8-6 Okno pre vytvorenie modelu v Simulinku

Do tohto okna budeme postupne presúvať a spájať jednotlivé bloky, a následne nastavovať ich parametre.

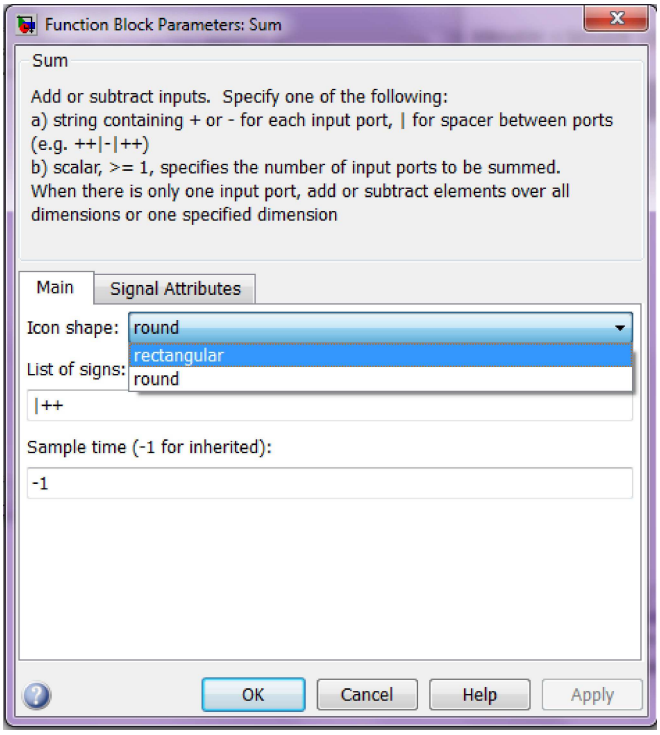
5. Vložíme blok **sumátor** jednoduchým kliknutím v ***Simulink Library Browser*** na konkrétny blok a potiahneme ho do nového okna pre model.



Obrázok 8-7 Blok sumátor

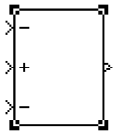
6. Blok **sumátor** je možné zmeniť na štvoruholníkový kliknutím na **sumátor** pravým tlačidlom myši a následným vybratím položky ***Sum Parameters***, alebo dvojklikom na blok.

Následne sa zobrazí :



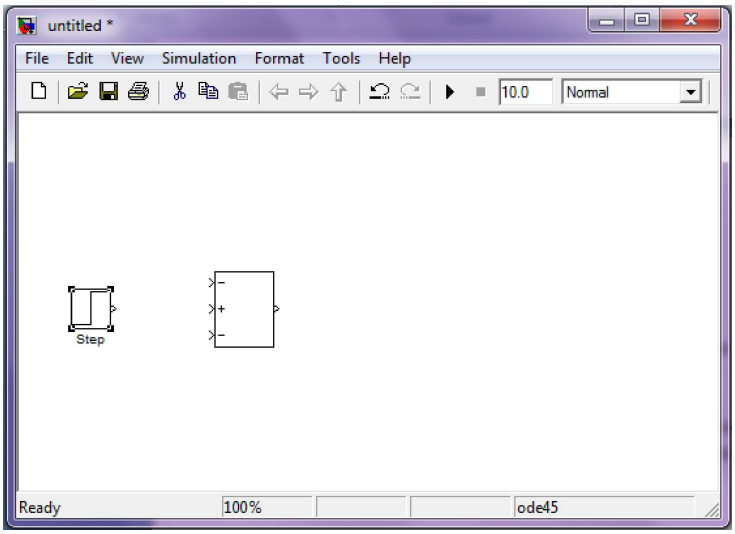
Obrázok 8-8 Definovanie vstupov do sumátora

V tomto prípade vyberieme **rectangular** a vstupy budú **- + -** a dostaneme



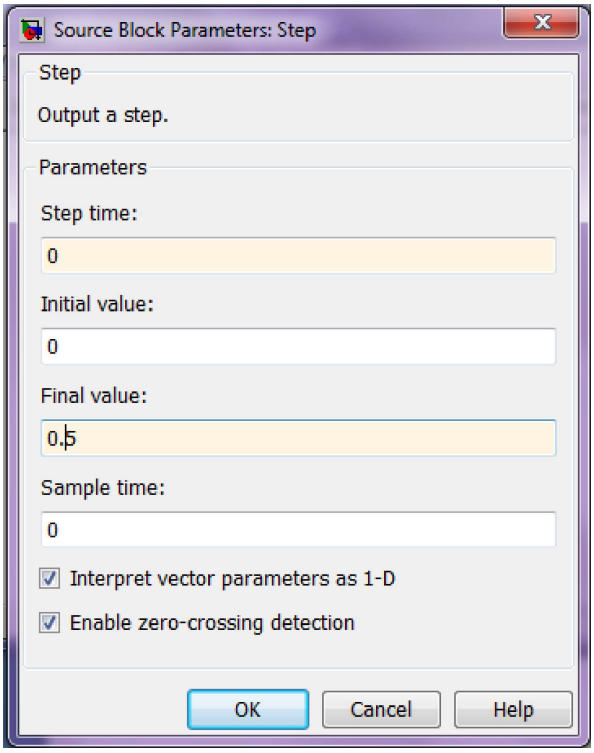
Výber potvrdíme tlačidlom OK.

7. Druhý blok, ktorý si pridáme je **vstup do systému**, ktorý nájdeme pod knižnicou **Sources** a vyberieme blok simulujúci jednotkový skok (step).



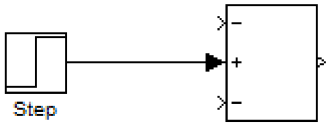
Obrázok 8-9 Blok pre vstup do systému (step)

8. Dvojklikom na blok **step** alebo kliknutím pravým tlačidlom na blok a výber ***Step Parameters*** otvoríme nastavenia parametrov tohto bloku.



Obrázok 8-10 Inicializácia parametrov pre funkciu step

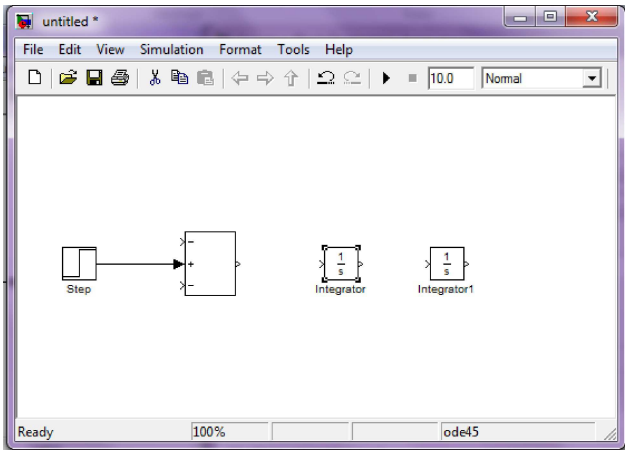
9. Bloky spojíme kliknutím na výstup bloku step a vstup sumátora a spojíme čiarou.



10. Ďalšie bloky, ktoré pridáme, budú dva integrátory. Integrátory nájdeme v položke ***Continuous*** alebo v položke ***Commonly Used Blocks.***

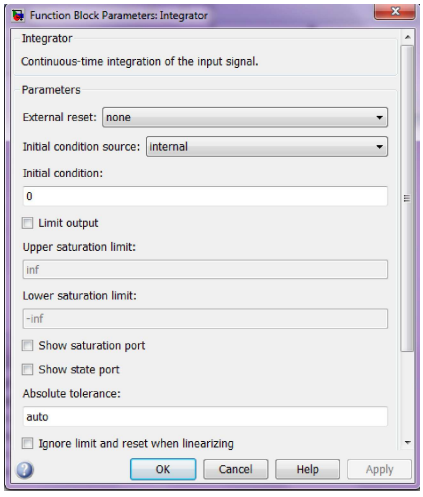
⇨ Prvý integrátor použijeme pre integráciu x_2' na $x_2 = x_1'$

⇨ Druhý na integráciu získaného x_2 na x_1



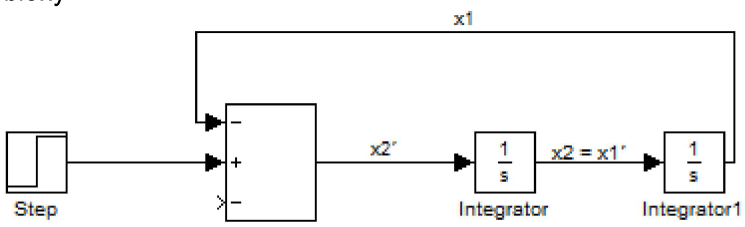
Obrázok 8-11 Integrátory pre zostavenie modelu v Simulinku

11. V jednotlivých integrátoroch buď dvojklikom na integrátor alebo kliknutím pravým tlačidlom myši na integrátor a následným výberom položky ***Block Properties*** sa otvorí okno, kde môžeme nastaviť napríklad počiatočné podmienky pre integráciu.

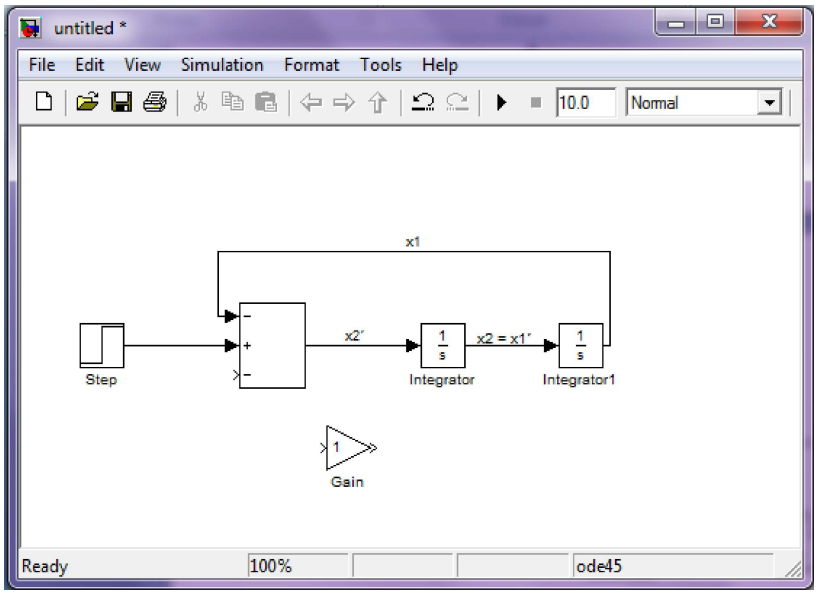


Obrázok 8-12 Nastavenie počiatočných podmienok pre integrátory

12. Pospájame bloky



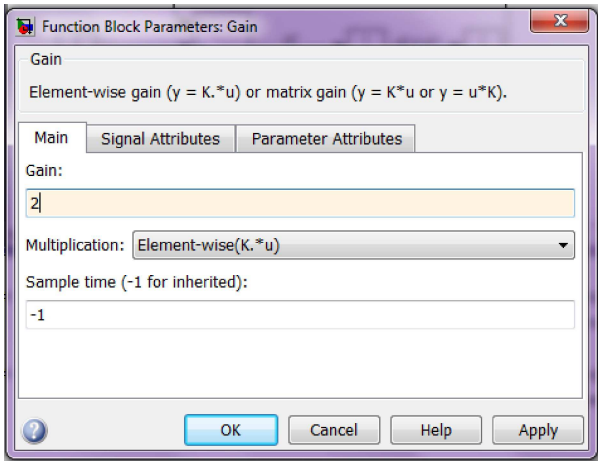
13. Potrebujeme ešte zosilniť signál x_2 a preto pridáme blok **Gain** z položky ***Math Operations*** alebo ***Commonly Used Blocks.***



Obrázok 8-13 Pospájanie blokov v modeli pre riešenie LDR

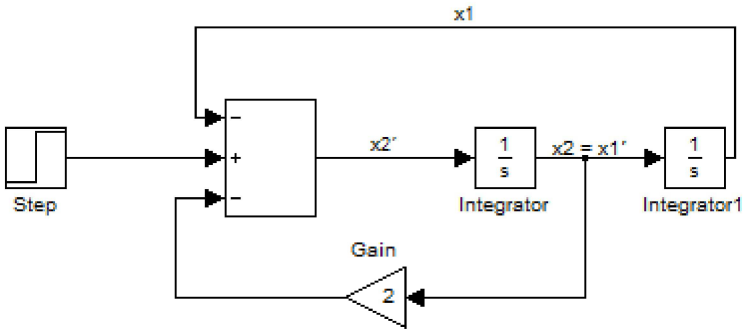
⇨ Pootočenie jednotlivých blokov je možné po kliknutí na blok ktorý chceme otočiť a stlačení kláves **CTRL+R** alebo kliknutím pravým tlačidlom na objekt a výberom položky **Format -> Rotate Block -> Clockwise**

⇨ Dvojklikom na zosilnenie **gain**, resp. kliknutím pravým tlačidlom myši na objekt a výberom položky ***Gain Properties*** sa otvorí okno, kde je možné napríklad zmeniť zosilnenie.

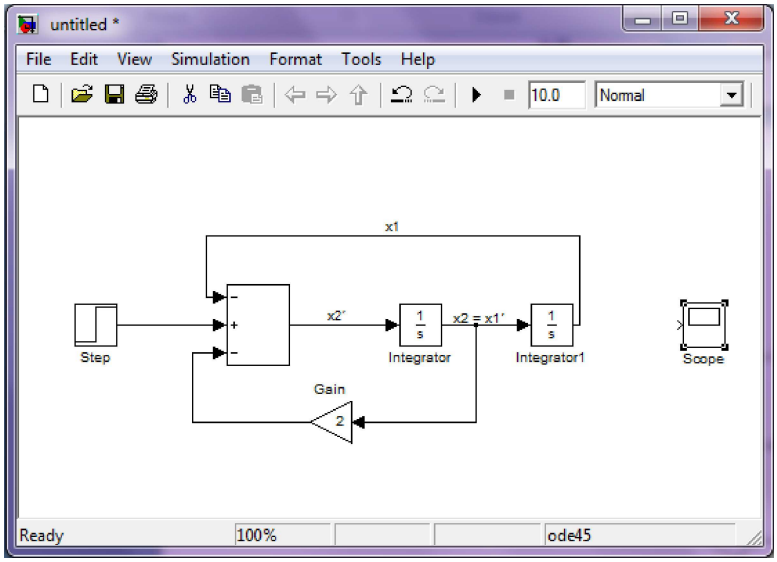


Obrázok 8-14 Nastavenie parametrov bloku gain

Po pospájaní jednotlivých blokov dostávame model v Simulinku na riešenie LDR:

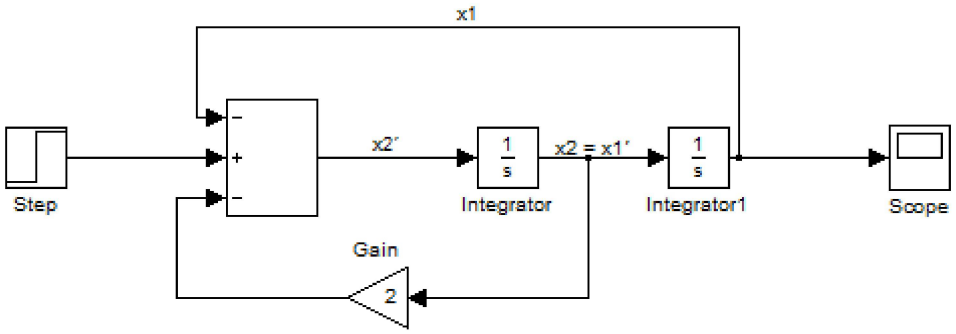


14. Nakoniec ak chceme vykresliť priebeh riešenia zadanej diferenciánej rovnice musíme pridať blok **Scope**. Blok Scope nájdeme v položke ***Sinks*** alebo ***Commonly Used Blocks.***

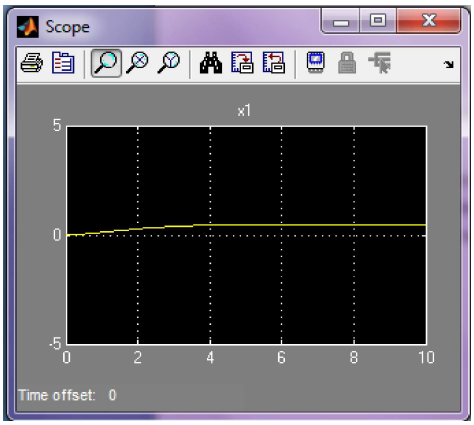


Obrázok 8-15 Model pre riešenie LDR s pridaním bloku Scope

⇨ Pripojenie výstupu integrátora integrator1 na vstup bloku **Scope** s cieľom vykresliť priebeh riešenia LDR:

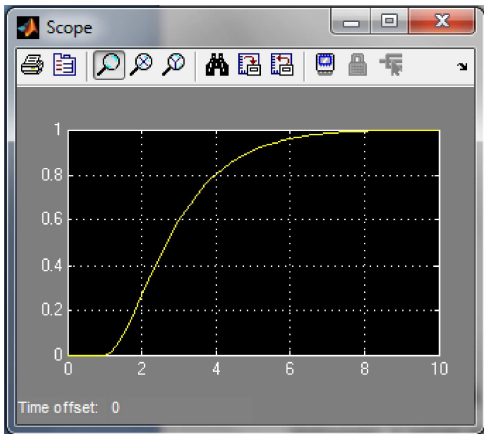


15. Pre vykreslenie priebehu riešenia musíme spustiť simuláciu a to pomocou ikony  .
16. Následne dvojklikom na Scope sa otvorí okno **figure** a v ňom priebeh simulovaného systému (lineárnej diferenciálnej rovnice)



Obrázok 8-16 Časový priebeh riešenia LDR

17. V bloku Scope kliknutím na ikonu  nám automatický priblíži nábeh vykreslenej simulácie (automatické škálovanie). V našom prípade to bude vyzerať:



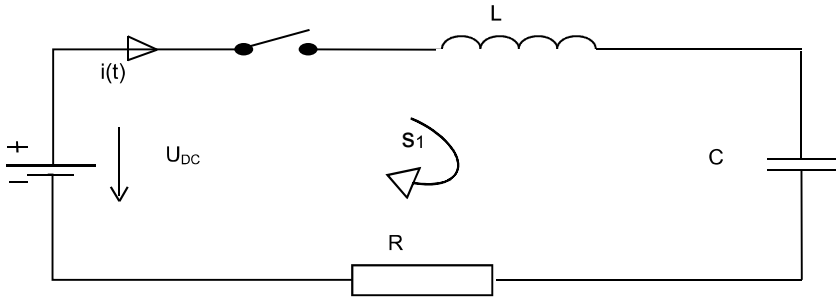
Obrázok 8-17 Škálovanie časového priebehu riešenia LDR v Simulinku

Ak chceme toto škálovanie použiť aj pre ďalšie simulácie klikneme na ikonu  , ktorá nám uloží aktuálne nastavenie osí. Teraz pri nasledujúcom spustení simulácie nám automaticky otvorí s uloženými nastaveniami osi. Opakom, teda na zrušenie uloženia osí slúži ikona  .

8.2 Simulácia modelov fyzikálnych systémov v prostredí SIMULINK

PRÍKLAD 1 – NABÍJANIE KONDENZÁTORA

Zostavte na základe matematického modelu RLC obvodu simulačný model v programovom prostredí Simulink a vypočítajte časový priebeh prúdu i(t) a časový priebeh napätia $u_c(t)$.



$$u_R(t) = Ri(t)$$

$$u_L(t) = L\frac{di(t)}{dt}$$

$$u_C(t) = \frac{1}{C}\int_0^t i(\tau)d\tau$$

Na základe vyššie uvedených vzťahov pre napätia na jednotlivých elektronických prvkoch obvodu vieme matematický model podľa 2.Kirchhoffovho zákona, ktorý hovorí, že súčet úbytkov napätí v uzavretej slučke obvodu je rovný nule. Znázornený RLC obvod obsahuje práve jednu slučku s_1, pre ktorú platí:

$$L\frac{di(t)}{dt} + Ri(t) + \frac{1}{C}\int i(t)dt = u_{DC}(t)$$

Zavedieme substitúciu pre prúd pretekajúci obvodom:

$$i(t) = C * \frac{du_c(t)}{dt}$$

Pomocou tejto substitúcie eliminujene integrál na pravej strane a získame výslednú diferenciálnu rovnicu 2. rádu popisujúcu nabíjanie kondenzátora cez technickú cievku:

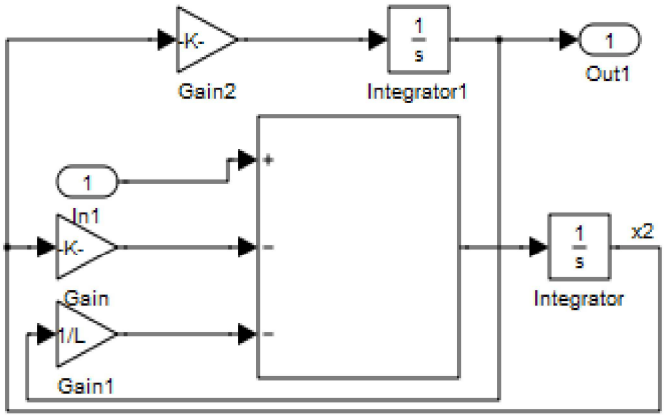
$$CL * \frac{d^2u_C(t)}{dt^2} + CR\frac{du_C(t)}{dt} + u_C(t) = u_{DC}(t)$$

Túto diferenciálnu rovnicu zapíšeme do substitučného kanonického tvaru a následne zostavíme model v Simulinku:

$$x_1'(t) = x_2(t)$$

$$x_2'(t) = \frac{1}{CL}u_{DC}(t) - \frac{1}{CL}x_1(t) - \frac{R}{L}x_2(t)$$

Vytvorený model pre nabíjanie kondenzátora cez technickú cievku v prostredí Simulink



PRÍKLAD 2 - HYDRAULIKA

Z nádoby o priereze $S = 2\,m^2$a výške $h = 2\,m$, vyteká voda otvorom na dne nádoby o priereze $S_0 = 0{,}001\ m^2$. Hydraulický súčiniteľ je $\alpha = 0{,}94$. Výška hladiny na začiatku sledovania je $h = 0{,}1\,m$. Zistite, ako sa bude meniť v čase výška hladiny $h(t)$ a vytekajúce množstvo Q_0, keď nebude vždy 1 minútu nič pritekať $Q = 0$ a následne 2 minúty bude pritekať množstvo $0{,}017\ m^3 s^{-1}$.

Hmotnostá bilancia systému: $\rho Q = \rho Q_0 = \frac{d(\rho S_N h(t))}{dt}$

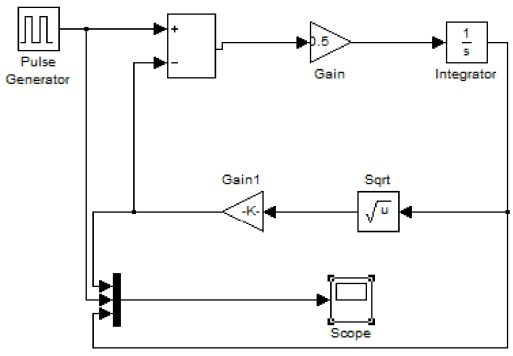
Torriceliho vzťah : $v_0 = \alpha\sqrt{2gh} \rightarrow \rho Q_0 = \rho v_0 S_0$

Výsledná nelineárna diferenciálna rovnica - závislosť h(t) na prítokového množstva Q:

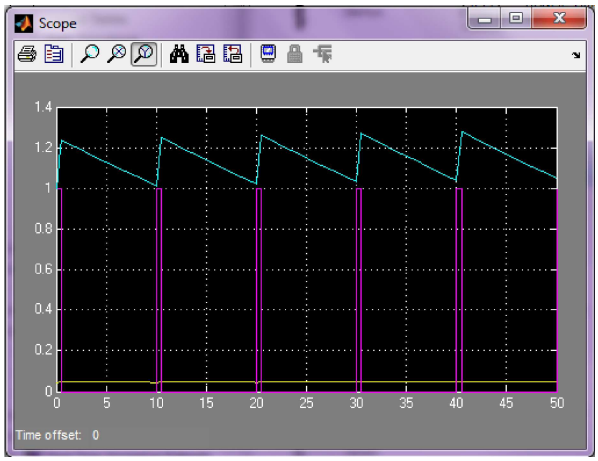
$$S_N \frac{dh(t)}{dt} + \alpha S_0\sqrt{2gh} = Q \qquad \text{h(t-0) = 1; h } \in <0,2)$$

$$h(t) = \int_0^t \frac{1}{s}(Q - \underbrace{\alpha S_0\sqrt{2g}}_{konšt.}\sqrt{h(t)})dt$$

Riešenie modelu hydrauliky – jednej nádoby v prostredí Simulink:



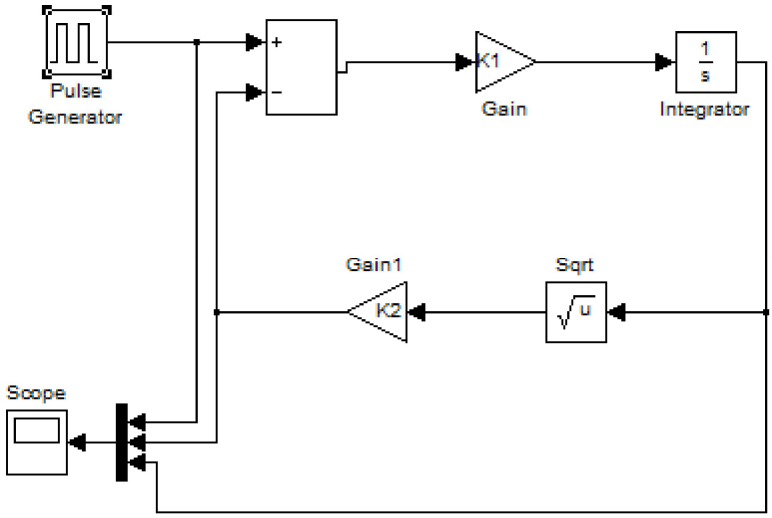
Obrázok 8-18 Model hydraulického systému



Obrázok 8-19 Časový priebeh výšky hladiny *h₁(t)*

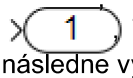
8.3 Tvorba subsystémov a maskovanie

Uvažujme simulačný model vytvorený pomocou blokov v Simulink-u. V tomto prípade je to naprogramovaný matematický model nádoby, z ktorej vyteká prerušovane kvapalina otvorom na dne (vždy 1 minúty nič nepriteká a následne 2 minúty kvapalina priteká).

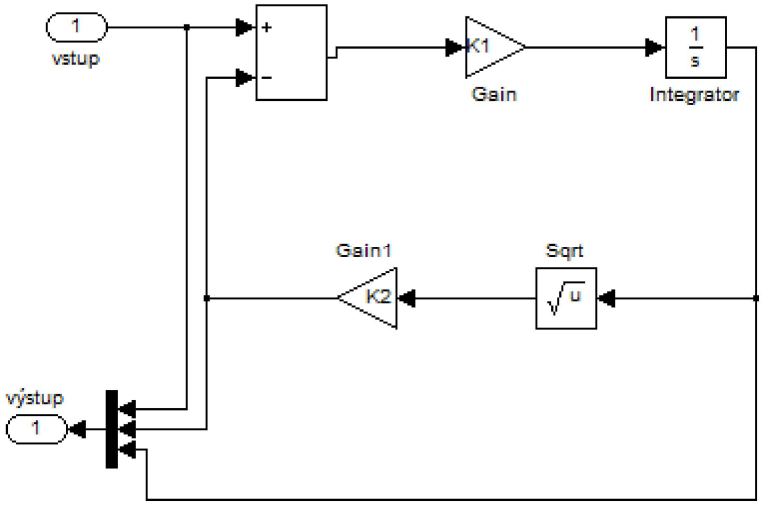


Úloha: Vytvorte subsystém z daného simulačného modelu v Simulink-u.

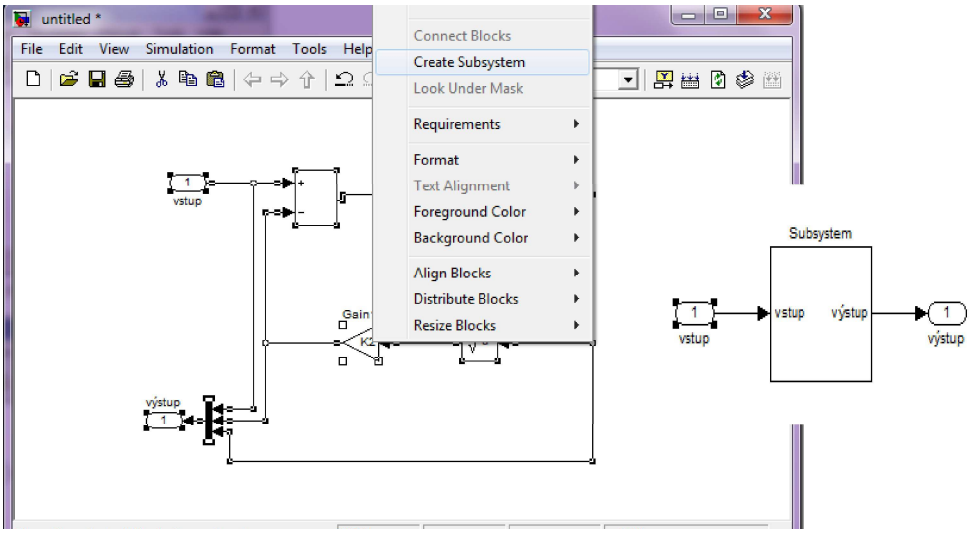
1. Všetky vstupy zameníme blokmi In 

a všetky výstupy zameníme blokmi Out 

Tieto bloky môžeme pomenovať a názvy sa následne prenesú aj do subsystému, ktorý následne vytvoríme .



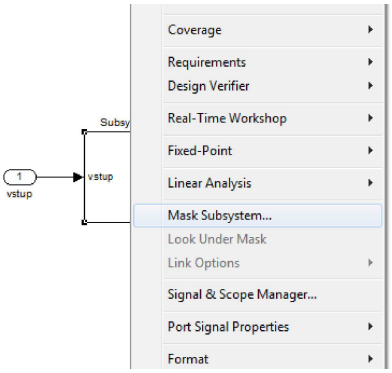
2. Teraz označíme všetky bloky systému, stlačíme pravé tlačidlo myši a vyberieme položku **Create Subsystem**

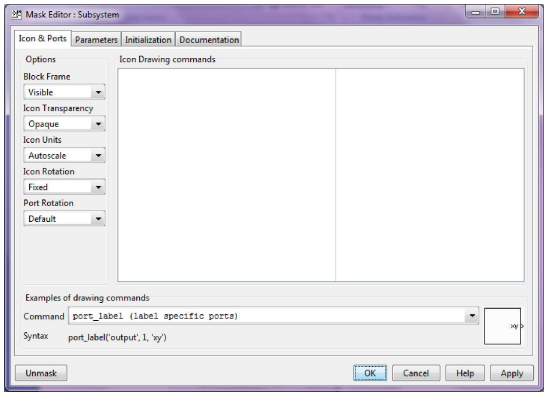


Obrázok 8-20 Vytvorenie subsystému

Úloha: Vytvorte masku daného subsystému.

1. Kliknutím pravým tlačidlom myši na **subsystém** a následným výberom položky **Mask podsystem** sa otvorí okno:

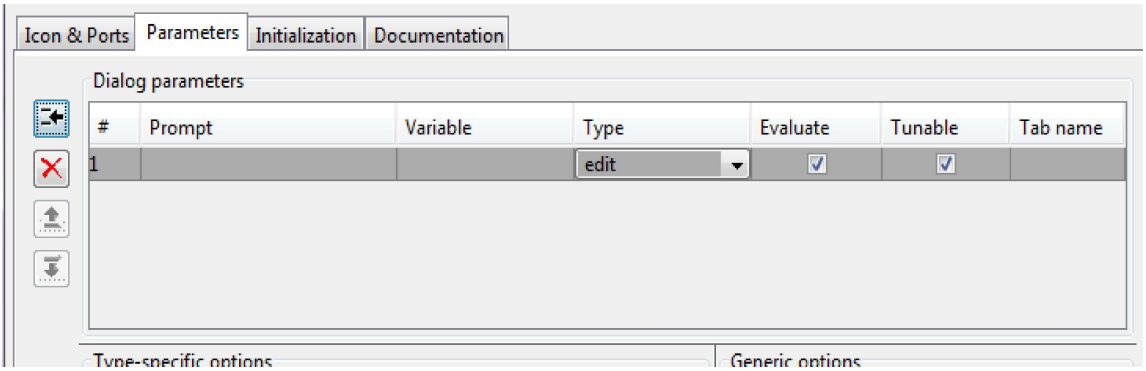




2. V záložke ***parameters*** je možné parametre, ktoré chceme meniť zapísať, a následne pri kliknutí na **subsystém** sa otvorí okno, kde si tieto parametre budeme môcť nastavovať.

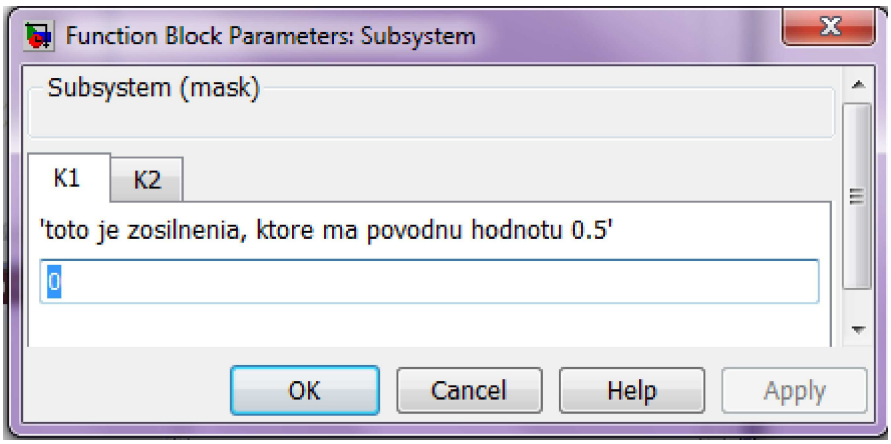
Napríklad v našom systéme si môžeme meniť **zosilnenia**, ak pri tvorení **subsystému** sme nekonkretizovali **hodnotu zosilnenia** ale zadali napríklad len **K**, potom postupujeme nasledovné:

a) V záložke *parameters* klikneme na ikonu  , a vytvorí sa nám 1 riadok v tabuľke.



Prompt – obsahuje popis tejto hodnoty, pozor musí byť zapísaný v apostrofoch, pretože je to reťazec
Variable – do tejto položky zapíšeme názov z systému, v našom prípade K
type – vyberieme zobrazenie
Tab name – vytvorí záložky s názvami pre jednotlivé premenné

Následne klikneme na *Apply*, ak chceme pridať ďalšiu položku zopakujeme postup, inak klikneme na *OK*.
Ak sme správne vytvorili premenné po kliknutí na subsystém sa nám zobrazí okno



V tomto okne si nastavíme požadované hodnoty a s nimi vykonáme simuláciu.

8.4 Zadanie č.5 : Simulácia LDR/NDR a modelu fyzikálneho systému v prostredí SIMULINK

ZADANIE: *Naprogramujte simulačnú schému (model) v prostredí Simulink na riešenie:*

a) *lineárnej diferenciálnej rovnice (zadanie č. 2) s uvažovaním definovaného budiaceho signálu,*
b) *nelineárnej diferenciálnej rovnice (zadanie č. 3) s uvažovaním definovaného budiaceho signálu,*
c) *odozvy fyzikálneho modelu (zadanie č. 4) na definovaný budiaci signál. So simulačným modelom pracujte ako so subsystémom a parametre nech sú zadávané v maske.*

- **Majme zadanú lineárnu diferenciálnu rovnicu tvaru:**

$$y''(t) + 2y'(t) + y(t) = t$$

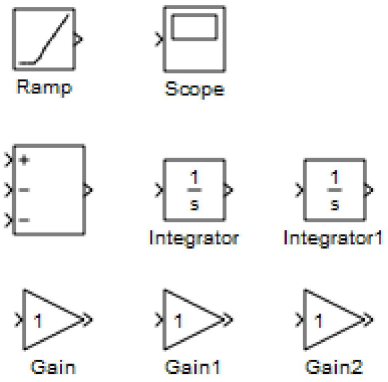
Počiatočné podmienky pre riešenie tejto lineárnej diferenciálnej rovnice sú $y(0) = y'(0) = 0$.

1. Zadanú diferenciálnu rovnicu 2. rádu rozdelíme na dve diferenciálne rovnice 1. rádu – teda prevedieme diferenciálnu rovnicu 2. rádu do substitučného kanonického tvaru:

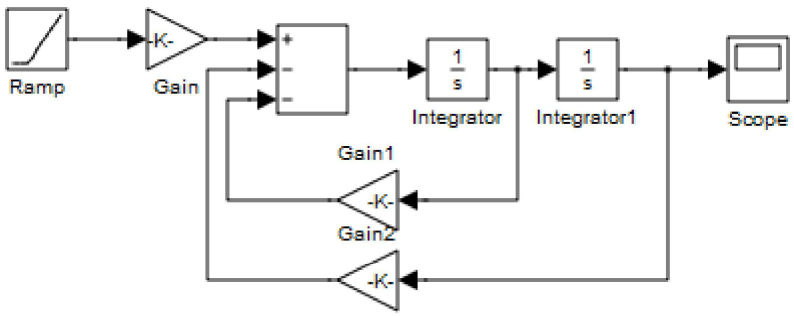
$$x_2 = x_1' = y'(t)$$
$$x_2' = y''(t) = t - 2 * x_2 - x_1$$

Substitúcia: $x_1 = y(t)$

2. Ako je vidieť zo substitučného kanonického tvaru, pre vytvorenie tejto diferenciálnej rovnice potrebujeme 2xintegrátor,1x zosilnenie, 1x súčtový člen , vstupný signál, v tomto prípade použijeme RAMP a pre vykreslenie získaného priebehu potrebujeme osciloskop. Ak však chceme vytvoriť všeobecné riešenie, kde používateľ má možnosť zadať vlastné hodnoty koeficientov a_0, a_1 a a_2 potom použijeme blok zosilnenia 3x.

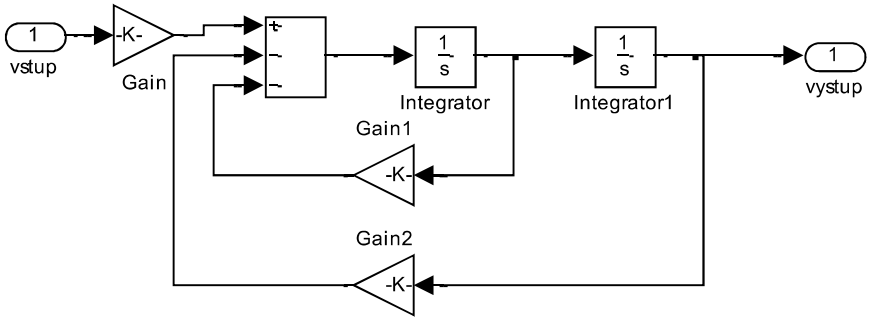


3. Pospájaním týchto blokov podľa vytvoreného substitučného kanonického tvaru získame riešenie zadanej lineárnej diferenciálnej rovnice v grafickom prostredí Simulink.

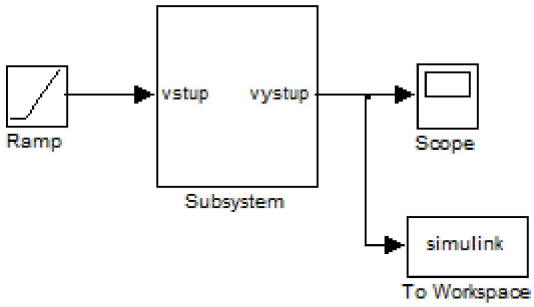


V tejto schéme Gain 1 predstavuje hodnotu a1/ a2 a Gain 2 hodnotu a0/a2 a Gain 1/a2.

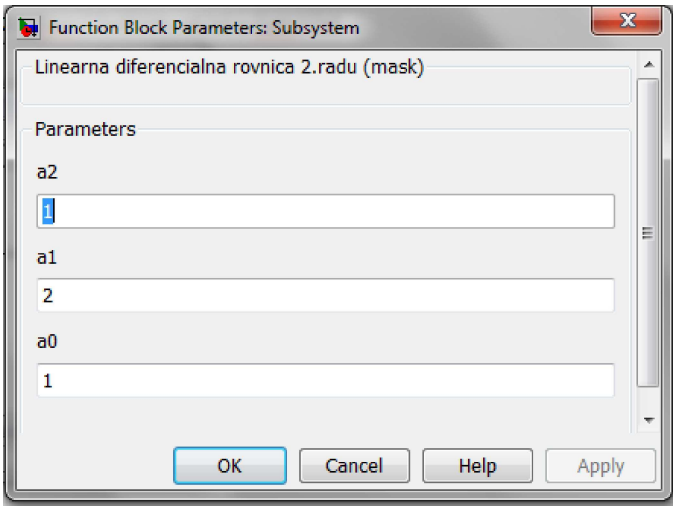
4. Zo získaného modelu vytvoríme subsystém tak, že za vstup (ramp) a výstup (scope) použijeme bloky IN a OUT.



5. Z tejto schémy modelu si následne vytvoríme subsystém a bloky IN a OUT zmeníme späť na Ramp a Scope, prípadne môžeme pridať aj blok To Workspace, ktorý vypočítané hodnoty zapíše do Workspace, kde s nimi môžeme ďalej pracovať .

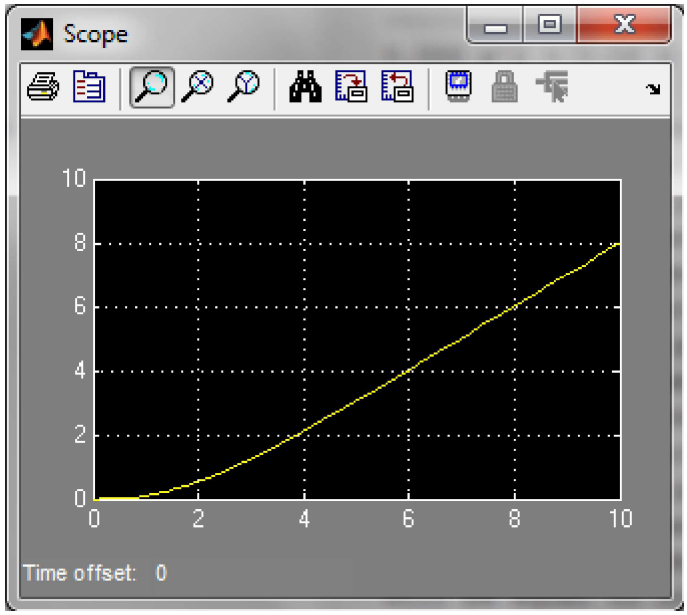


6. Parametre vytvoreného subsystému zamaskujeme, čím pri kliknutí na subsystém sa nám otvorí okno, v ktorom je možné nastavovať parametre a0, a1,



Obrázok 8-21 Nastavenie príslušných parametrov pre LDR v Simulinku

7. Výsledné grafické riešenie bude vyzerať:



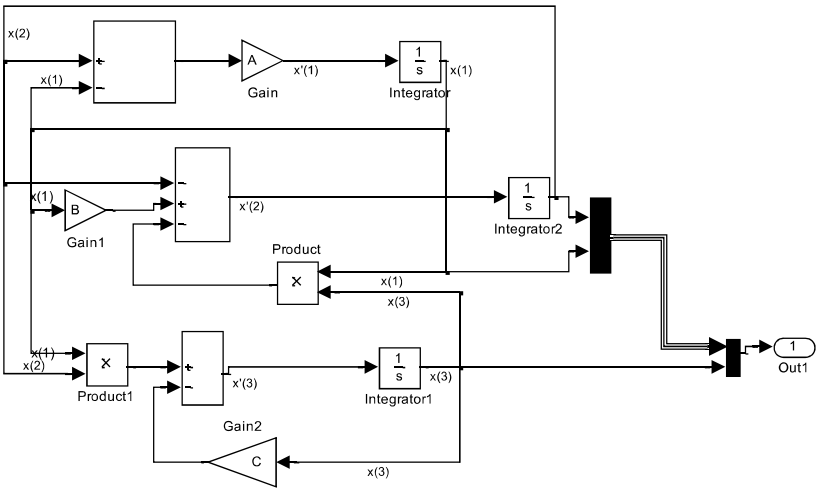
Obrázok 8-22 Časový priebeh riešenia $x_1(t)$ ako záznam z bloku Scope

- **Majme sústavu 3 diferenciálnych rovníc 1. rádu, ktoré predstavujú turbulencie v kvapalinách:**

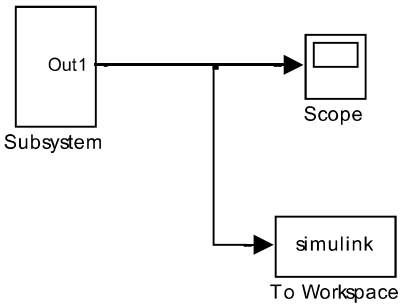
$$x_1'(t) = A(x_2(t) - x_1(t))$$
$$x_2'(t) = Bx_1(t) - x_2(t) - x_1(t) * x_3(t)$$
$$x_3'(t) = x_1(t)x_2(t) - Cx_3(t)$$

1. Pretože všetky rovnice sú 1. rádu, nepotrebujeme vytvárať substitučný kanonický tvar a môžeme začať rovno vytvárať model v prostredí Simulink:

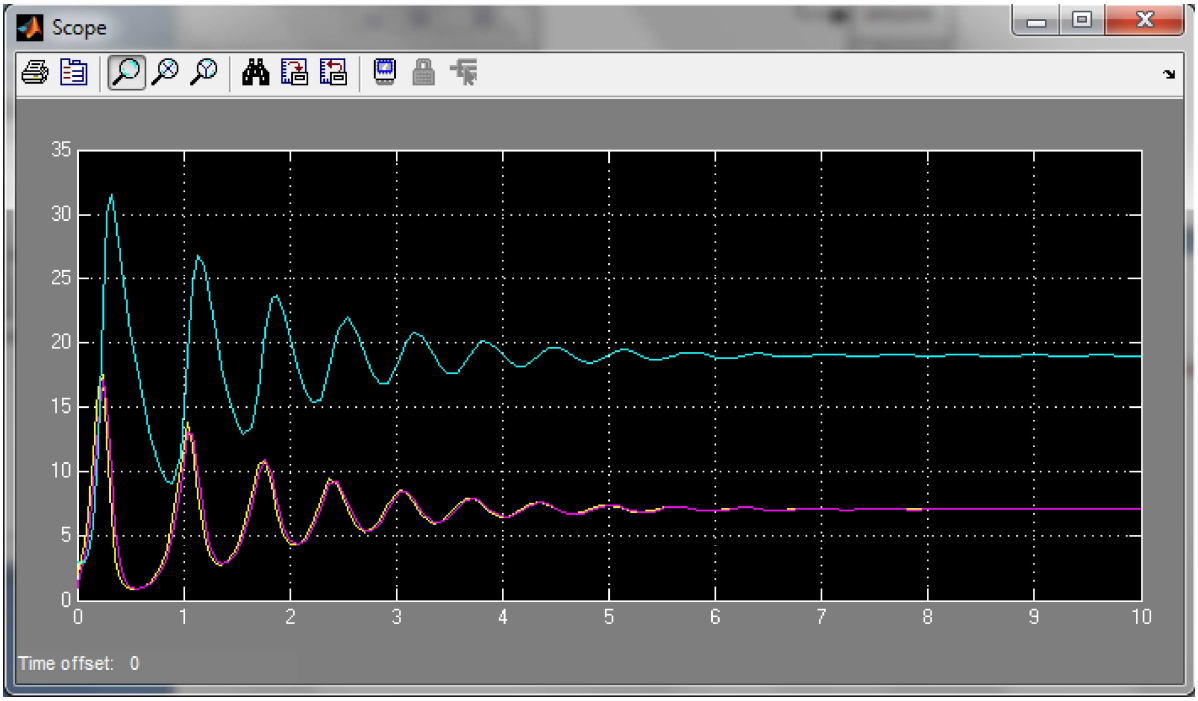
Postup pri vytváraní modelu bude zhodný z postupom vytvárania modelu lineárnej diferenciálnej rovnice. Pre tvorbu subsystému si zameníme iba blok výstupu (Scope) za blok Out, vstup do systému v tomto prípade neexistuje, pretože do týchto diferenciálnych rovníc nevstupuje žiadna budiaca sila:



2. Z takto vytvoreného modelu vytvoríme subsystém a ten následne zamaskujeme:



3. Spustením simulácie a nahliadnutím na oscilátor (scope) uvidíme:



Obrázok 8-23 Riešenie NDR z modelu v Simulinku

- **Majme matematický model otáčok motora. Našou úlohou je vytvoriť simulačný model otáčok motora pomocou programového nástroja Simulink.**

1. Tento model je popísaný diferenciálnou rovnicou 2. rádu tvaru:

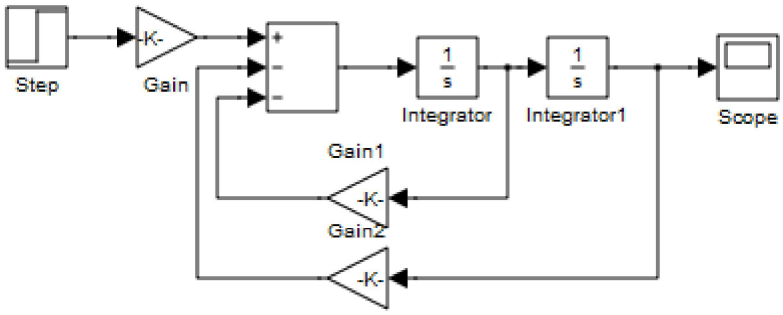
$$L * J * \frac{d^2\omega}{dt^2} + (L * B + R * J) * \frac{d\omega}{dt} + \left(R * B + C_u{}^2\right) * \omega = U(t) * C_u$$

Pre riešenie tohto modelu potrebujeme diferenciálnu rovnicu prepísať do substitučného kanonického tvaru, ktorý bude mať nasledovný tvar:

$$x_1{}' = \omega' = x_2$$
$$x_2{}' = \omega'' = \frac{1}{LJ}\left(C_u U(t) - (L * B + R * J)x_2 - \left(RB + C_u{}^2\right)x_1\right)$$

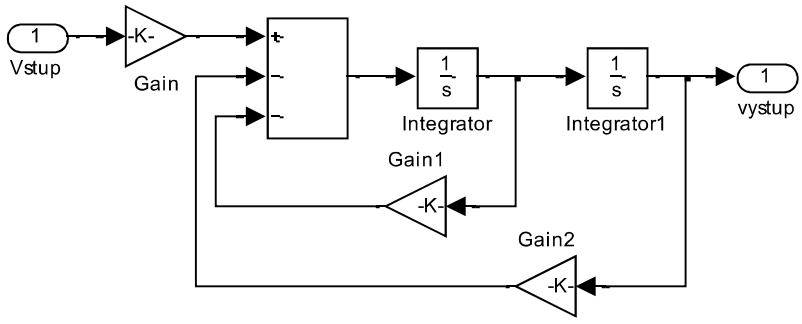
Substitúcia:
$x_1 = \omega$

2. Následne vytvoríme schému modelu, ktorá bude mať nasledovný tvar:

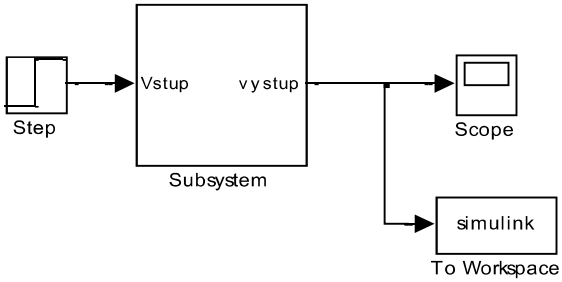


V tejto schéme Gain predstavuje hodnotu Cu/(J*L), Gain 1 hodnotu (B/J)+(R/L) a hodnota Gain 2 je daná ako ((Cu*Cu)/(J*L))+((B*R)/(J*L)).

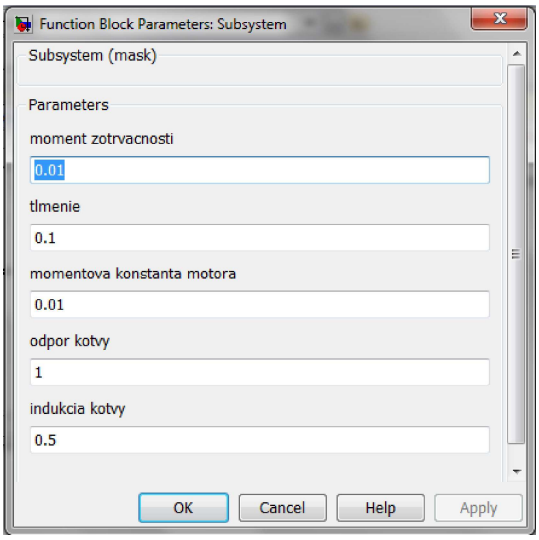
3. Pre vytvorenie subsystému potrebujeme dosadiť za vstup (Step) a výstup (Scope) bloky In a Out.



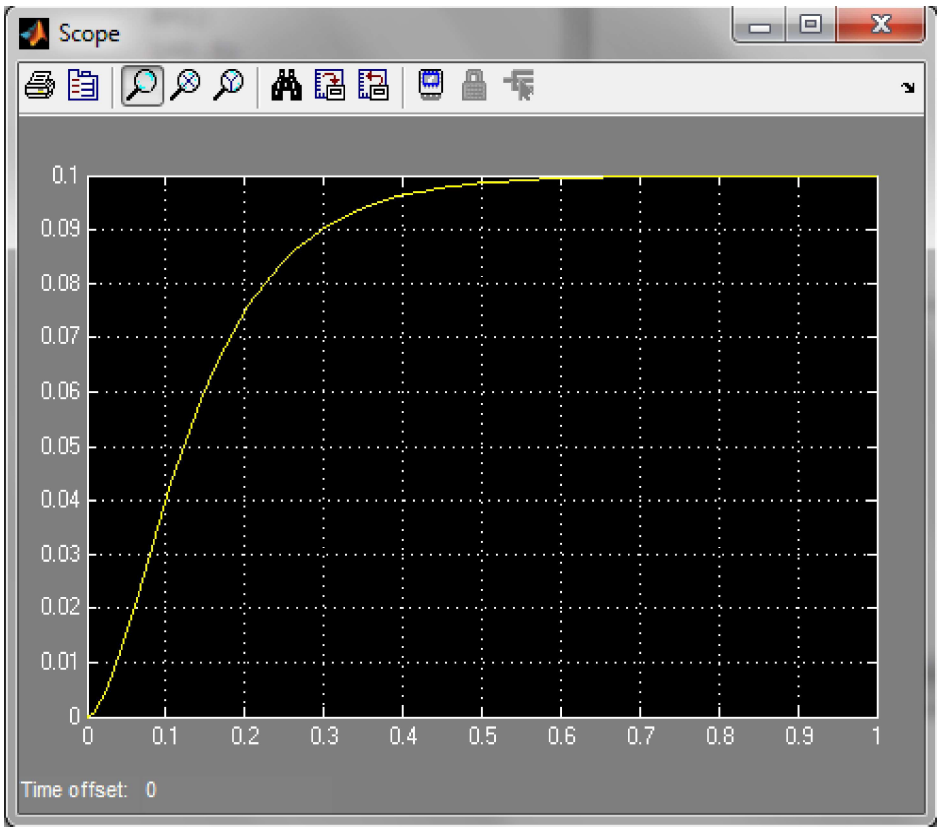
4. Takto vytvorený subsystém zamaskujeme a máme výsledný model otáčok motora:



Po dvojkliku na subsystém sa objaví blok pre nastavenie jednotlivých parametrov:



5. výstup získaný z oscilátora (scope)



Obrázok 8-24 Časový priebeh otáčok motora získaný zo simulačného modelu v Simulinku pri nastavených parametroch modelu a simulácie

8.5 Zadanie č. 6: Návrh a simulačné overenie algoritmu PID pre riadenie otáčok jednosmerného motora

ZADANIE: Navrhnite algoritmus riadenia (PI, PD, PID) dvoma metódami syntézy pre Vami zvolený fyzikálny model dynamického systému (viď zadanie č. 4), pričom rád dynamického systému je $n \leq 3$.

Navrhnutý algoritmus riadenia overte v spätnoväzobnej štruktúre uzavretého regulačného obvodu (URO) pomocou blokov knižníc Simulink, kde model vyskladajte z blokov namiesto použitia bloku Transfer Fanction pri skokovej zmene vstupov URO: $w(t) = 1(t) pre\ t = 0(s)$ a $z(t) = 0.39(t)\ pre\ t = 0.5(s)$.

OBSAH ZADANIA:

1. Analytický výpočet parametrov PID algoritmu zvolenými metódami syntézy pre simulačný model dynamického systému.
2. Overenie PID algoritmu v spätnoväzobnej štruktúre URO v Simulinku.
3. Grafické priebehy regulovanej veličiny:

$y(t)\ pre\ w = 1, z = 0$

$y(t)\ pre\ w = 0, z = 1(t)$

$y(t)\ pre\ w(t) = 1(t), z = 0.3(t) pre\ t = 0.5s$

4. Graf odozvy systému na jednotkový skok.

Majme zadanú prenosovú funkciu otáčok motora
:

$$F(s) = \frac{C_u}{s^2(JL) + s(BL + JR) + (C_u{}^2 + BR)}$$

Ak dosadíme hodnoty $J = 0{,}01\ kg.m^2$, $B = 0{,}1\ N.m.s$, $R = 1\ \Omega$, $L = 0{,}5H$, $C_u = 0{,}01\,{}^{N.m}/_{Amp}$

Získame obrazový prenos sústavy:

$$F_s(s) = \frac{1}{0{,}5s^2 + 6s + 10{,}0{,}1}$$

Pre riadenie zadanej sústavy si zvolíme PI regulátor, ktorého obrazový prenos vyzerá nasledovne:

$$F_R(s) = K * \left(1 + \frac{1}{T_i s}\right)$$

<u>Zvolíme si riešenie metódou Graham-Lathrop</u>

1. Potrebujeme získať charakteristickú rovnicu, ktorá má tvar $1+F_R(s)F_s(s)$:

$$1 + \frac{T_i K s + K}{T_i s} * \frac{1}{0{,}5s^2 + 6s + 10{,}0{,}1} = 0$$

Po roznásobení a úprave získame charakteristickú rovnicu v tvare:

$$\frac{0{,}5T_i s^3 + 6T_i s^2 + 10{,}01T_i s + KT_i s + K}{0{,}5T_i s^3 + 6T_i s^2 + 10{,}01T_i s} = 0$$

Menovateľ je rovný nule. Zavedieme substitúciu:

$$\frac{K}{T_i} = I$$

A následne ešte čitateľa charakteristickej rovnice upravíme do tvaru:

$$s^3 + 12s^2 + 2 * (10{,}01 + K)s + 2I = 0$$

Z tabuľky štandardných tvarov charakteristických rovníc podľa Graham-Lathropa vyčítame odpovedajúcu charakteristickú rovnicu, ktorá musí obsahovať tak ako naša charakteristická rovnica 3 mocninu pri najvyššom laplaceovom operátorovi :

$$s^3 + 1{,}75s^2\omega_0 + 2{,}15s{\omega_0}^2 + {\omega_0}^3$$

Z týchto dvoch rovníc potrebujeme získať neznáme hodnoty ω_0, K, I

$$s^3 : 1 = 1$$

$$s^2 : 12 = 1{,}75\omega_0$$

$$\omega_0 = 5{,}857$$

$$s : 20{,}02 + 2K = 2{,}15{\omega_0}^2$$

$$K = 40{,}535$$

$$s^0 : \ 2I = {\omega_0}^3$$

$$I = 100{,}46$$

Ďalšou možnosťou pre získanie hodnou K a I systému je použiť Naslinovú metódu

Opäť začíname získaním charakteristickej rovnice, ktorá bude teda vyzerať $1+F_R(s)F_s(s)$. Výsledná charakteristická rovnica teda bude vyzerať:

$$0{,}5s^3 + 6s^2 + (10{,}01 + K)s + \frac{K}{T_i} = 0$$

Zavedieme substitúciu:
$$\frac{K}{T_i} = I$$

Podľa vzťahu ${a_j}^2 \geq \alpha a_{j-1}a_{j+1}$ pričom α získame z tabuľky:

α	1.75	1.8	1.9	2.0	2.2	2.4
$\sigma(\%)$	16	12	8	5	3	1

Pre náš výpočet si určíme maximálne preregulovanie 5%.
A teda získame dve nerovnice, ktorý úpravou získame hodnoty K a I

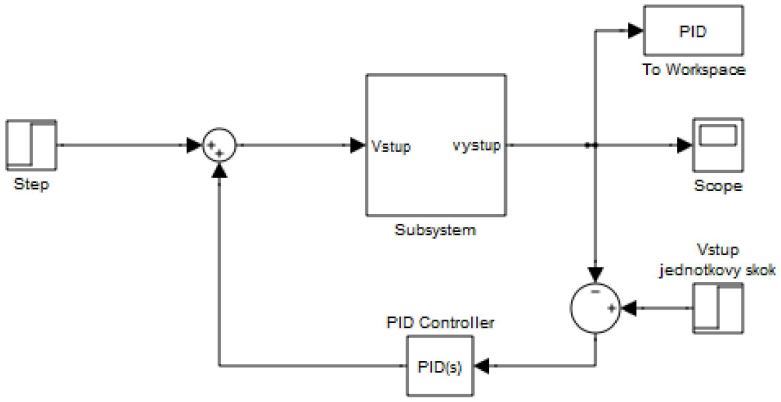
$$j = 1: \quad (10{,}01 + K)^2 \geq 2 * I * 6$$

$$j = 2: \quad 6^2 \geq 2 * 0{,}5 * (10{,}01 + K)$$

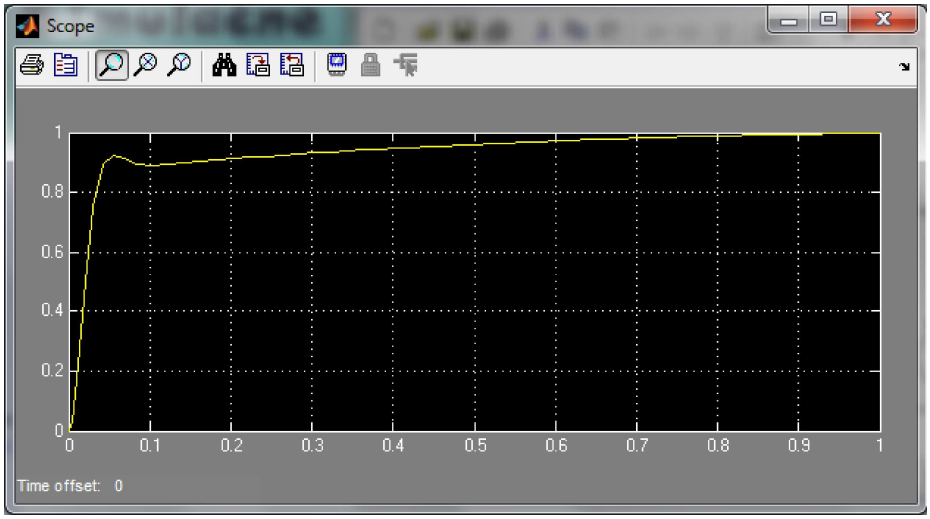
$$K \leq 25{,}99$$
$$I \leq 108$$

RIEŠENIE:

1.

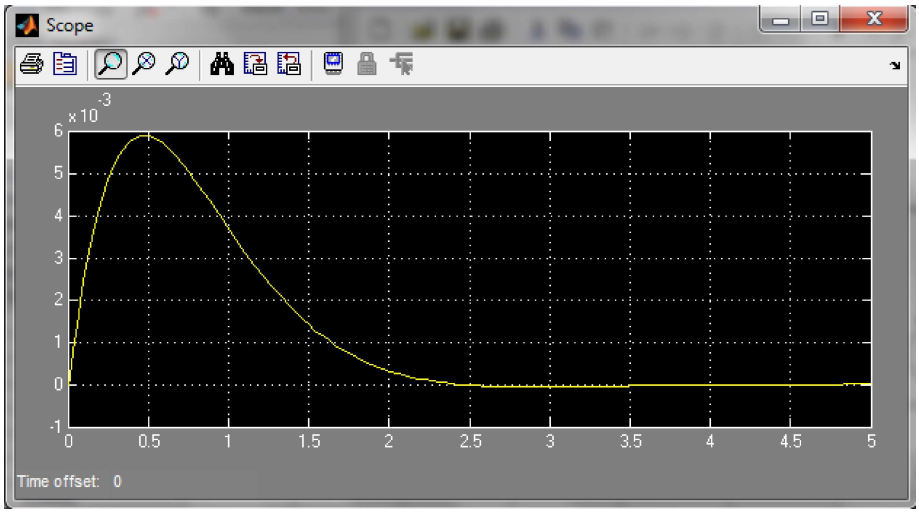


2.



Obrázok 8-25 Časový priebeh regulácie otáčok pri zmene vstupného signálu riadiacej veličiny w(t)

3.



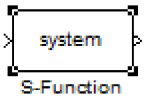
Obrázok 8-26 Časový priebeh regulácie otáčok pri pôsobení poruchového signálu z(t)

8.6 Postup pri tvorbe s-funkcií

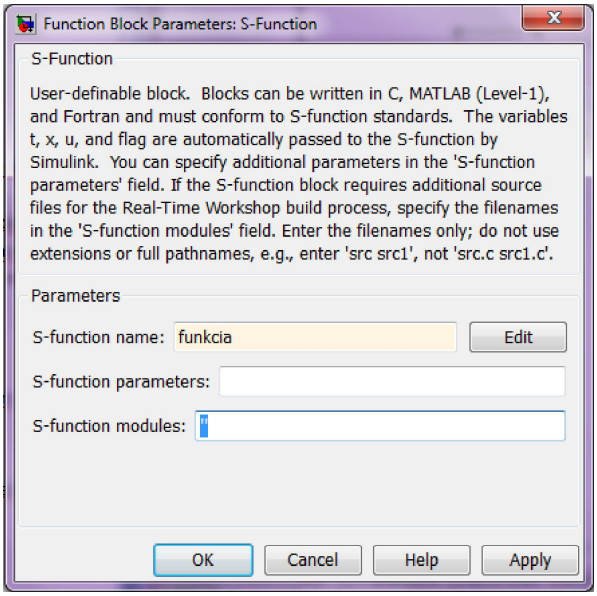
- ⇨ S-funkcia je dynamicky blok v Simulinku, ktorého “opis” je definovaný vo funkcií-súbore , ktorý je vytvorený v simulačnom jazyku MATLAB, v jazykoch C, C++, Ada alebo Fortran.
- ⇨ S-funkcie vytvorené v jazykoch C, C++, Ada a Fortran sú kompilované na MEX-súbory
- ⇨ S-funkcia umožňuje užívateľovi vytvárať vlastné bloky do modelu v Simulink-u. V m-súbore s-funkcie môžu byť definované vlastné diferenciálne rovnice, rovnice diskrétneho systému a/alebo ľubovoľný typ algoritmu.

Použitie S-funkcie v modeloch

Blok S-funkcion sa nachádza v knižnici Simulink → User- Defined Function →S-Function



Po presunutí do bloku modelu sa po dvojkliku na systém otvorí okno Block Parameters: S-Function.



- ⇨ Parametre sú oddeľované čiarkou a môžu to byť konštanty (vektory, matice), názvy premenných definovaných v pracovnom priestore programového prostredia MATLAB alebo výrazy v programovom prostredí MATLAB.
- ⇨ Parametre t, x, u (čas, stavy a vstupy) sú Simulink-om automaticky prenášané do S-funkcie.

Význam S-funkcií

- ⇨ S-funkcia reprezentuje dynamický prvok (u → [x] → y) , kde výstupy sú funkciou periódy vzorkovania, vstupov a stavov.
- ⇨ Matematické vzťahy medzi vstupmi, výstupmi a stavmi môžu byť vyjadrené nasledujúcimi rovnicami:

$y = f_y(t, x, u)$ (výstup)

$x_s = f_s(t, x, u)$ (derivácia – spojitý systém)

$x_d(t+1) = f_d(t, x, u)$ (diferencia – diskrétny systém)

$x = \{x_s, x_d\}$

⇨ Simulácia S-funkcie prebieha v niekoľkých etapách:

1. *flag = 0* - inicializácia (funkcia *mdlInitializeSizes*)
2. *flag = 4* - výpočet periódy vzorkovania (funkcia *mdlGetTimeOfNextVarHit*)
3. *flag = 3* - výpočet výstupu y(funkcia *mdlOutputs*)
4. *flag = 2* - výpočet diskrétnych stavov x_d (funkcia *mdlUpdate*)
5. *flag = 1* - výpočet spojitých stavov x_s (funkcia *mdlDerivatives*)
6. *flag = 9* - koniec (funkcia *mdlTerminate*)

S-funkcia v tvare m-súboru je definovaná ako obyčajná funkcia v simulačnom jazyku MATLAB:

$$\boldsymbol{function\ [sys,x0,str,ts] = meno(t,x,u,flag,p1,p2,\ldots)}$$

kde ***meno*** je názov S-funkcie, ***t*** je aktuálny čas, ***x*** je vektor stavov daného bloku (S-funkcie), ***u*** sú vstupy do bloku, ***flag*** udáva vykonávanú úlohu a ***p1,p2, a ďalšie*** sú parametre bloku.

⇨ Počas simulácie modelu, Simulink opakovane vyvoláva S-funkciu ***meno*** a na základe ***flag***-ov sa vykonávajú jednotlivé etapy S-funkcie.

PRÍKLAD 1
Vytvorte s-funkciu pre model Van der Pólovho oscilátora, ktorý je popísaný diferenciálnou rovnicou:

$$y'' - A(1-y^2)y' + y = u$$

Po prepise do substitučného kanonického tvaru, za podmienky, že stavová premenná $x_1 = y$, získame dve diferenciálne rovnice 1.rádu:

$$x_1' = x_2$$
$$x_2' = u - x_1 + x_2 * A * (1 - x_1^2)$$

Postup:

1. Otvoríme si Simulink a z knižnice User-Defined Functions vyberieme blok S-Function a vložíme ho do vytváraného systému alebo do nového súboru.
2. Po dvojkliku na tento blok sa otvorí okno v ktorom vieme zmeniť názov tohto bloku a pridať parametre.
3. Otvoríme si nový m-file, ktorý bude predstavovať funkciu a jej názov bude rovnaký ako názov nami vytvorenej s-funkcie.
4. Ak máme vytvorenú funkciu začneme s definovaním príznakov, ktoré budeme využívať. V našom prípade bude táto časť vyzerať nasledovne:

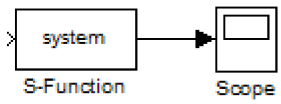
```
switch flag,
  case 0 % inicializácia
    [sys,x0,str,ts]=mdlInitializeSizes;
  case 1 % výpočet spojitých stavov
    sys=mdlDerivatives(t,x,u);
  case 3 % príznak výstupu
    sys=mdlOutputs(t,x,u);
  case {2, 4, 9}
    sys = [];
  otherwise
    DAStudio.error('Simulink:blocks:unhandledFlag',    num2str(flag));    %
chybový oznam, v pprípade ak sa vyskytné iný príznak ako sú definované.
end
```

pokračujeme definovaním konkrétnych príznakov:

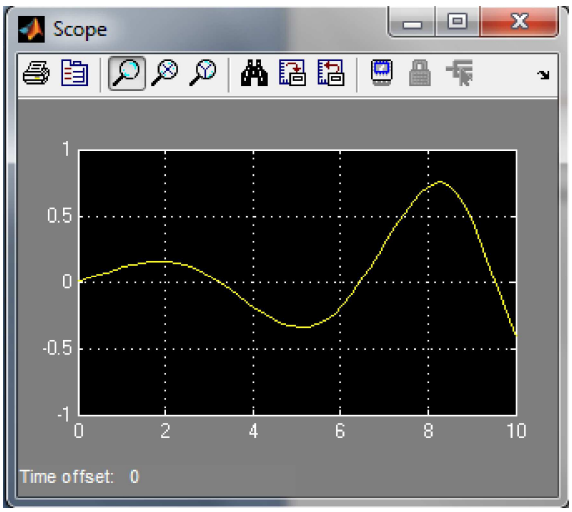
```
%=============================================================================
% mdlInitializeSizes
%=============================================================================
function [sys,x0,str,ts]=mdlInitializeSizes
```

```
sizes = simsizes;


sizes.NumContStates  = 2; %počiatočné stavy
sizes.NumDiscStates  = 0;
sizes.NumOutputs     = 1; %výstup
sizes.NumInputs      = 1; %vstup
sizes.DirFeedthrough = 0;
sizes.NumSampleTimes = 1;

sys = simsizes(sizes);
x0  = [0; 0]; % počiatočné podmienky
str = [];
ts  = [0 0]; % spojitý systém

%=============================================================================
% mdlDerivatives
%=============================================================================
function sys=mdlDerivatives(t,x,u)
 A=0.5; u=0.1    % definovanie vnútorných premenných
sys = [x(2); u-x(1)+x(2)*A*(1-x(1)*x(1))];

%=============================================================================
% mdlOutputs
%=============================================================================
function sys=mdlOutputs(t,x,u)
sys = x(2); % Výstupom je druhý stav
```

5. Do schémy pridáme ostatné bloky ktoré sú potrebné pre spustenie simulácie a systém je vytvorený.



Získaná simulácia vyzerá nasledovne:



Obrázok 8-27 Časový priebeh riešenia NDR (Van-der-Polov oscilátor) získaný z S-funkcie