Srdce jako pumpa

Na cvičení jsme si udělali

- Zdroj tlaku (a)
- Rezistenci (b)
- Chlopeň (c)
- Elastický kompartment (d)
- Model a-b-c-d-c-b-a

Vaše mise (pro MCH: úkol)

Implementace bloku ***zdroj elasticity*** a bloku ***srdečních intervalů***.

Vyzkoušíte si:

- Tvorbu vlastního diskrétního konektoru
- Použití when
- Použití if
- Logicky přemýšlet ☺

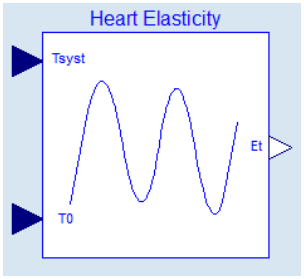
Heart elasticity

Výstup tohoto bločku *Et* bude záviset na vyhodnocení podmínek.

Pokud `time - T0 >= 0` a zároveň `time - T0 <= Tsyst` budou `true`, pak `Et = Edias + (Esys - Edias)/2*(1 - cos(Modelica.Constants.pi*(time - T0)/Tsyst))`

Pokud `time - T0 < (3/2)*Tsyst` bude `true`, pak `Et = Edias + (Esys - Edias)/2*(1 + cos(2*Modelica.Constants.pi*(time - T0 - Tsyst)/Tsyst))`

A pokud nebude podmínka ani jednou `true`, pak `Et = Edias`.



Obrázek 1. Ikona zdroje elasticity

Systolickou a diastolickou elasticitu parametrizujte a přednastavte na hodnoty 1/0,4 a 1/10. Rovněž nezapomeňte na jednotky (daná elasticita funguje pro [torr/ml]).

Heart intervals

Nyní se pokusíme zapojit zdroj skutečné elasticity do komory. Zdroj jej již hotov, nicméně chybí jeho ovládání – před každým beatem je nutné ho naplnit časem počátku následující systoly a její délkou.

Vytvořte si blok, který bude provádět výpočet srdečních intervalů. Bude opatřen jedním vstupem (*RealInput*) a dvěma výstupy (*DiscreteRealOutput*). Např. obrázek níže. HR je spojitá proměnná vyjadřující srdeční frekvenci. Na to musí bloček reagovat a neustále vypočítávat nové intervaly.

Obrázek 2: Ukázka možného vzhledu ikony HeartIntervals

- RealInput použijete ze standard Modelica Library
- DiscreteRealOutput vyrobíte z RealOutput z Modelica Library. Zkopírujte (*duplicate*) si ho do svého projektu a duplikát pojmenujte. V textovém režimu pak v něm doplňte klíčové slovo `discrete` před slovo `output`.
    - (To znamená, že si vyrobíme bloček, který je stejný jako RealOutput, akorát že jeho proměnná není *Real*, ale *discrete Real* a umístíme si ho do našeho package)
    - Proč není i RealInput na HeartElasticity discrete? Dobrá otázka – měl by totiž být, ale takhle to též funguje.
- Legenda:
    - HR "Heart rate (beats/min)"
    - TSyst "delka systoly v sec"
    - T0(start=0) "delka systoly v sec"

Dále je nutné provést deklaraci těchto proměnných:

```
discrete Real TPulsePrev "delka predchoziho srdecního cyklu v sec";
Boolean b;
XXX TPulse;
```

Jakého typu bude TPulse? Doplňte správně místo XXX

Nyní se dostáváme k sekci inicializačních rovnic, která bude mít 3 rovnice:

```
TPulse=60/HR;
TPulsePrev=TPulse;
TSyst=0.3*(TPulse^0.5);
```

Poznámka: nastavení počáteční hodnoty v rovnicích v sekci *initial equation* je ekvivalentní použití přímo v deklaraci, např. *discrete Real TPulsePrev(start = TPulse)*.

Než se pustíme do rovnic, tedy sekce *equation*, je důležité pochopit rozdíl mezi `if` a `when`. Opakování z minulé přednášky ☺

- If používáme k rozhodnutí, kterou ze dvou (i více) alternativ programu vykonat. Vždy se ale vykoná jen jedna alternativa či jedna cesta.

- When používáme k jednorázovém vykonání části programu nad jeho rámec. Pokud je tedy splněna podmínka, vykoná se „obsah“ v této části.

První rovnice pro booleovskou proměnnou b bude vyhodnocovat podmínku:

- `time-pre(T0) je větší než pre(TPulse)`

Její výsledek bude true nebo false. Proto jsme použili DiscreteRealOutput → abychom mohli získat hodnoty T0 v předchozím stavu (pre(T0)). Pomocí operátoru pre rozlišujeme minulou a budoucí hodnotu (stále to jsou rovnice).

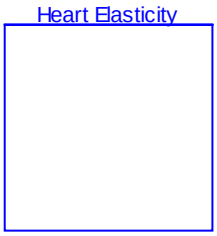
Co vrací pre(TPulse)? Odpověď uveďte do zprávy.

Nyní potřebujeme reagovat na událost, kdy dojde k přechodu proměnné b z false na true (z 0 na 1). Bude tedy **reagovat** pouze **na náběžnou hranu** proměnné **b** (čili použijete if, nebo when?). Viz číslicová technika a honění jedniček a nul… A pouze při této změně se vykonají tyto rovnice:

- `T0=time;`
- `TPulse = 60/HR;`
- `TPulsePrev=pre(TPulse);`
- `TSyst = 0.3*(pre(TPulse)^0.5);`

Použijeme if nebo when? Na co reaguje when? Jak bude vypadat podmínka? Dokážete popsat smysl operátorů pre()?

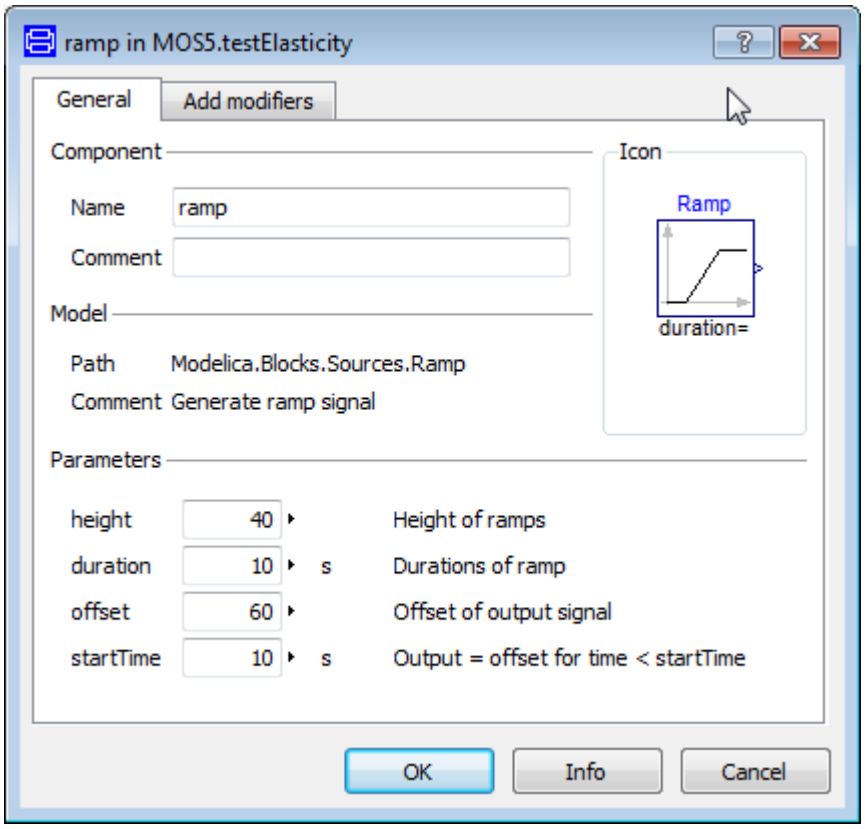
Blok je hotový a nyní jej zapojíme do systému.



Obrázek 3: TestHeartElasticity – před a po

Nezapomeňte na vstup vyrobeného bloku (HR) připojit zdroj rychlosti srdečního tepu. Budeme simulovat klidný tep 60 bpm, který se v 10 vteřině začne zvedat a ve 20 vteřině dosáhne 100 bpm, kde setrvá až do 30s.

Pro otestování můžete zpočátku použít konstantu, nebo rampu z Modelica.Library, pak bychom ale rádi viděli vlastní implementaci (zde použijete if, nebo when?) – pro upřesnění povinně.



Obrázek 4: Parametry

Blok HR

Pomocí *if* a *time* si rozdělte čas do intervalů 0, 10 a 20. V některých blocích bude hodnota konstantní, jinde bude určitou rychlostí (tu si spočtěte, ať už předem, nebo dynamicky) stoupat (trik: místo derivace můžete použít znovu *time*, tentokrát ale jako hodnotu). Poté je opět fixována na určitou hodnotu.

Zapojení do obvodu

Nyní to všechno spojte dohromady, jako je na obrázku 4.

Obrázek 4: Finální diagram

Simulace

- Pozorujte průběhy v Elastický kompartment (d).
    - konkrétně elastanci. Čím je tento průběh omezen?
    - Objem. Okomentujte jeho průběh. Co když nastavím pExt na 1?
- Jak se mění proměnné ve zdrojích tlaku - Zdroj tlaku (a)?
- Jakým způsobem dochází k uzavírání a otevírání chlopní?
- Pozorujte vygenerovaný průtok a srovnejte tlaky a průtoky s realitou

Pokud to není jasné..

ČTĚTE chybové hlášky – většinou se vám snaží něco sdělit!

Případnou potřebnou nápovědu můžete žádat na tomas.krocek@gmail.com, pokud myslíte, že to bude lépe probrat na konzultaci, obracejte se také na tomas.krocek@gmail.com.

Bonus

- `Tentokrát není, čímž zvyšujeme váhu těch minulých.`