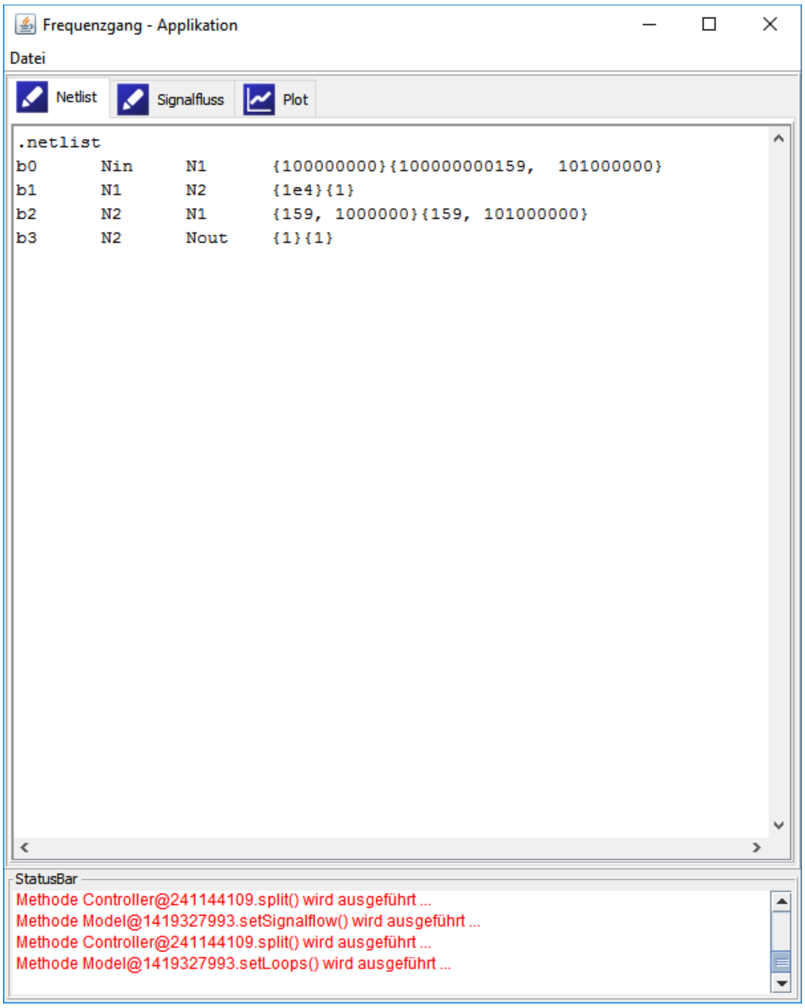
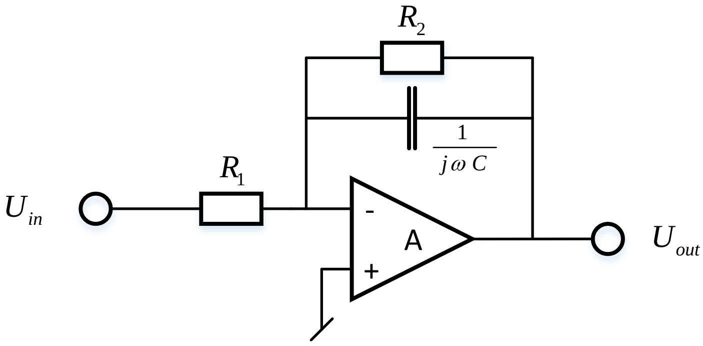
Übung Signalflussdiagramm Teil 1

Ziel dieser Übung ist es, eine Applikation zur Berechnung von Frequenzgängen, mittels Signalflussdiagrammen und der Regel von Mason, und deren Darstellung zu programmieren. Die zur Berechnung benötigten Polynome sollen in nummerischer Form vorliegen.

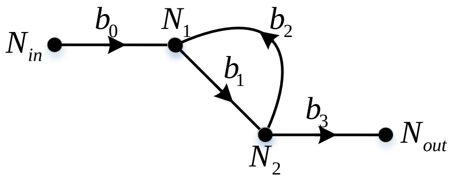


Signalflussdiagramme

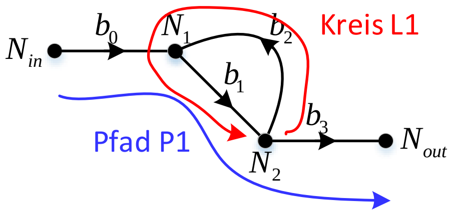
Das Signalflussdiagramm (SFD) ist die grafische Repräsentation eines linearen Gleichungssystems. Mit SFDs lassen sich Gleichungssysteme vergleichsweise einfach lösen. Wir betrachten dazu ein simples Beispiel einer Operationsverstärkerschaltung:



Das zugehörige Signalflussdiagramm hat folgende Form:



Ein Signalflussdiagramm ist ein gerichteter Graph und beinhaltet Knoten N_i und Zweige b_i. Die Knoten verkörpern Summationspunkt und sind durch die gerichteten Zweige verbunden. In einem SFD lassen sich Vorwärtspfade und Kreis einzeichnen:



Sowohl beim Pfad wie auch beim Kreis darf ein Knoten nur einmal durchlaufen werden. Die Berechnung der Zweige geschieht für obige OP-Schaltung sehr einfach:

- Für den Zweig b_0 wird die Übertragung von U_{in} zum Minuseingang des OPs für U_{out} gleich Null (Masse) ermittelt:

$$b_0(s) = \frac{R_2}{sCR_1R_2 + R_2 + R_1}$$

- Der Zweig b_1 entsprich der Verstärkung vom Minuseingang zum Ausgang des OPs:

$$b_1 = -A$$

- Für den Zweig b_2 wird die Übertragung von U_{out} zum Minuseingang des OPs für U_{in} gleich Null (Masse) ermittelt:

$$b_2(s) = \frac{sCR_1R_2 + R_1}{sCR_1R_2 + R_2 + R_1}$$

- Der Zweig b_3 ist 1:

$$b_3 = 1$$

Der entsprechende Maxima[1] – Code ist im Anhang zu finden

[1] http://maxima.cesga.es/

Regel von Mason

$$H(s) = \frac{U_{out}(s)}{U_{in}(s)} = \frac{\sum_{k=1}^{N} P_k(s) \cdot \Delta_k(s)}{\Delta(s)}$$

Wobei alle Funktionen gebrochenrationale Funktionen in s sind. Für die Berechnung gilt folgendes:

- $P_k(s)$ ist der k^{te} Vorwärtspfad und ergibt sich aus dem Produkt aller Zweige b_i entlang des Pfades P_k.

- $\Delta(\mathrm{s})$ ist die Determinante des Graphen und gegeben durch:

  $$\Delta(\mathrm{s}) = 1 - \sum L_i + \sum L_i\, L_j - \sum L_i\, L_j L_k + \cdots$$

  wobei nur Kreise, die sich gegenseitig nicht berühren, zu berücksichtigen sind.
- $\Delta_k(\mathrm{s})$ ist der Kofaktor des k^{ten} Vorwärtspfades und wird gleich wie die Determinante berechnet, wobei nur Kreise, die den k^{ten} Vorwärtspfad nicht berühren, zu berücksichtigen sind.

Symbolische Berechnung

Für unser Beispiel ergibt sich symbolisch folgendes Bild:

$$P_1(s) = b_0(s) \cdot b_1(s) \cdot b_3(s) = \frac{R_2}{sCR_1R_2 + R_2 + R_1} \cdot (-A) \cdot 1$$

Für den Kreis $L_1(s)$ ergibt sich:

$$L_1(s) = b_1(s) \cdot b_2(s) = -\frac{sACR_1R_2 + AR_1}{sCR_1R_2 + R_2 + R_1}$$

Für die Determinante erhalten wir:

$$\Delta(s) = 1 - L_1(s) = 1 + \frac{sACR_1R_2 + AR_1}{sCR_1R_2 + R_2 + R_1}$$

Für den Kofaktor $\Delta_1(\mathrm{s})$ erhalten wir:

$$\Delta_1(\mathrm{s}) = 1$$

Für die gesamte Übertragungsfunktion ergibt sich somit

$$H(s) = \frac{P_1(s) \cdot \Delta_1(\mathrm{s})}{\Delta(s)} = \frac{-AR_2}{s(A+1)CR_1R_2 + R_2 + (A+1)R_1}$$

Lässt man die Verstärkung A des OPs gegen unendlich streben, ergibt sich:

$$H_\infty(s) = \lim_{A\to\infty} \frac{-AR_2}{s(A+1)CR_1R_2 + R_2 + (A+1)R_1} = \frac{-R_2}{sCR_1R_2 + R_1}$$

Es handelt sich bei der Schaltung um einen aktiven, invertierenden Tiefpass 1.Ordnung mit einer DC-Verstärkung von $\frac{R_2}{R_1}$!

Applikation

Ziel der Applikation ist wie erwähnt, die numerische Berechnung des Frequenzganges. Die Zweige und die Topologie des SFDs sind numerisch wie folgt gegeben:

.netlist

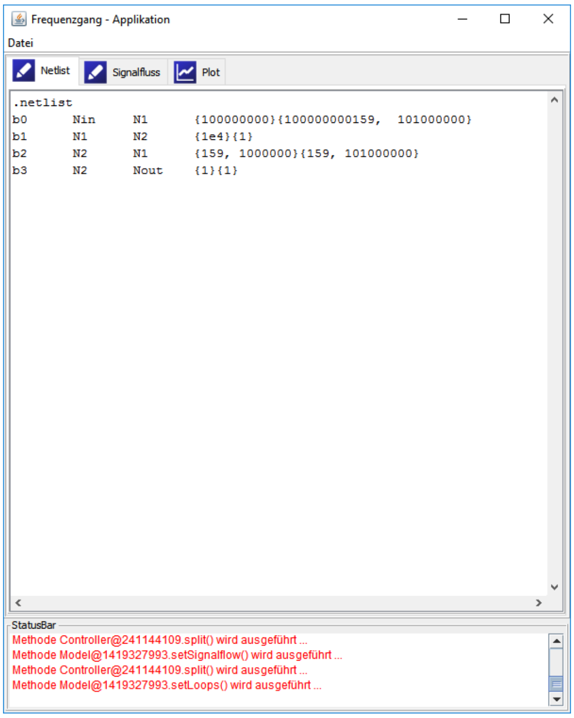
b0	Nin	N1	{100000000}{100000000159, 101000000}
b1	N1	N2	{1e4}{1}
b2	N2	N1	{159, 1000000}{159, 101000000}
b3	N2	Nout	{1}{1}

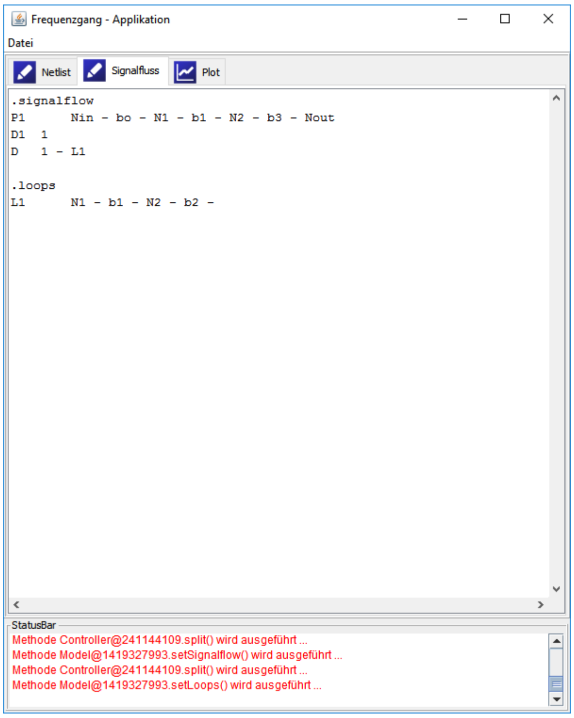
Die Kreise und der Signalfluss liegen als textliche Beschreibung in folgender Form vor:

.signalflow
P1 Nin - bo - N1 - b1 - N2 - b3 - Nout
D1 1
D 1 - L1

.loops
L1 N1 - b1 - N2 - b2 -

Die Applikation erlaubt obige Netzliste sowie die Beschreibung der Kreise und des Signalflusses aus einer Datei zu laden. Diese Daten werden in der Folge im Reiter *Netlist* und im Reiter *Signalfluss* dargestellt:





Aufgabe 1: Erstellen des Grundgerüstes

Als erstes wollen wir das Grundgerüst des Programms anhand des nachfolgenden Klassendiagramms erstellen. Ein paar Elemente der Klassen sind dabei bereits gegeben.

a) Erstellen Sie sämtliche Klassen-, Attribut- und Methodendeklarationen gemäss Klassendiagramm.
b) Implementieren Sie die fehlenden Interfaces mit zugehörigen Methoden.

Aufgabe 2: Erstellen der Klasse View

Die Klassen *View* erbt von *JTabbedPane* und hat drei Reiter, auf denen *JPanel* platziert werden.

a) Erstellen Sie das User-Interface gemäss diesen Angaben.
b) Testen Sie ihren Code.

Aufgabe 3: Erstellen der Klasse MeineMenuBar

Die Klasse *MeineMenuBar* verfügt über die drei *MenuItem* «Datei laden», «Datei speichern» und «Exit».

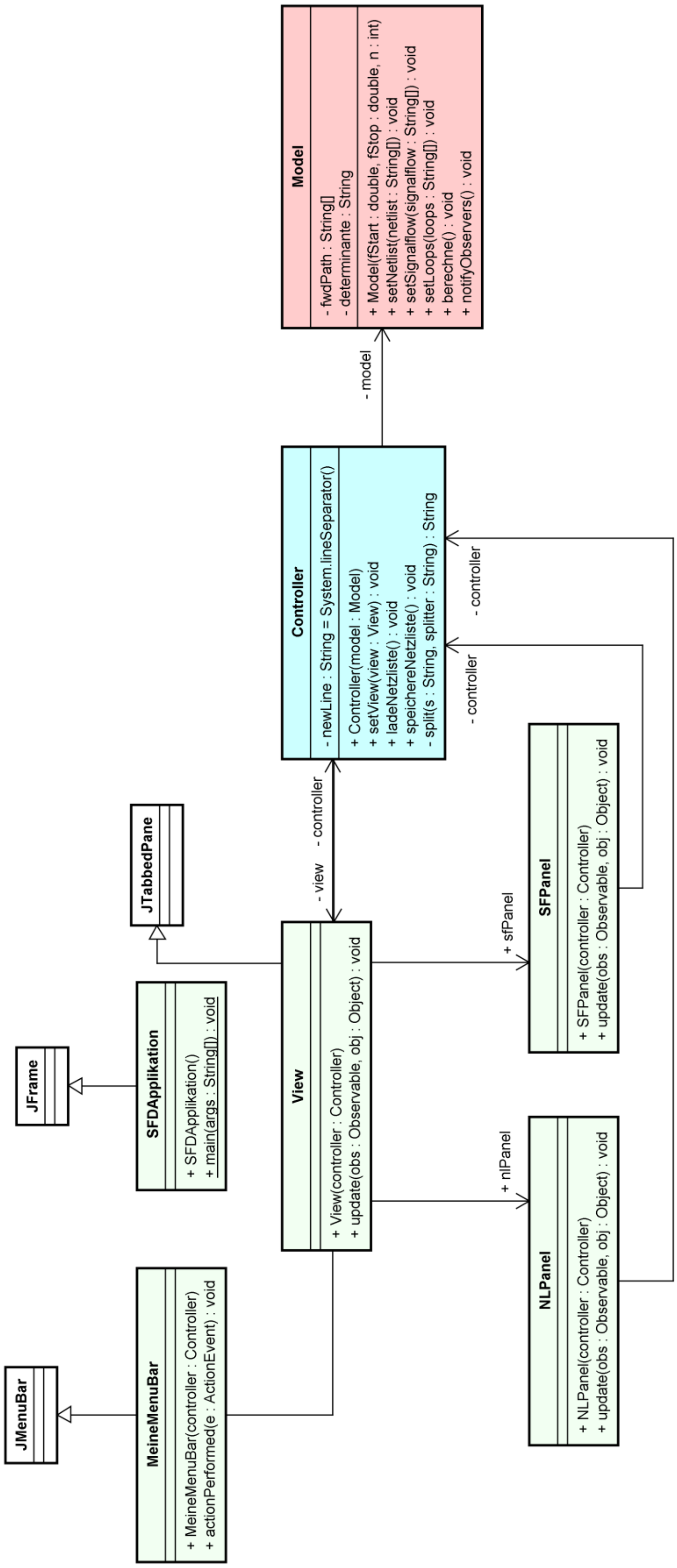
a) Implementieren Sie die Klasse *MeineMenuBar*.

Aufgabe 4: Erstellen der Klasse Controller

Der Kontroller agiert als Schaltzentrale bezüglich User-Interaktion. Er erlaubt die Dateien zu laden und zu speichern.

a) Implementieren Sie die Methoden der Klasse *Controller*.
b) Testen Sie das Laden und das Speichern der Datei *Test.net*.

Klassendiagramm Teil 1



Anhang

Maxima: http://maxima.cesga.es/

Maxima-Code zur symbolischen Berechnung der Übertragungsfunktion:

```
Z2: ratsimp((R2*(1/(s*C)))/(R2+(1/(s*C))), s);      /* Parallelschaltung R2 // C */
b0: ratsimp(Z2/(R1+Z2), s);                         /* Spannungsteiler ... */
b1: -A;
b2: ratsimp(R1/(R1+Z2), s);                         /* Spannungsteiler ... */
b3: 1;
L1: ratsimp(b1*b2, s);                              /* Kreis */
P1: ratsimp(b0*b1*b3, s);                           /* Vorwärtspfad 1 */
D1: 1;                                              /* Kofaktor 1 */
D: ratsimp(1-L1, s);                                /* Graph-Determinate */
H: ratsimp(P1*D1/D, s);                             /* Übertragungsfunktion */
Hi: limit(H, A, inf);                               /* Grenzwert A -> unendlich */
```

Maxima-Code zur numerischen Berechnung der Übertragungsfunktion:

```
R1: 1e3$
R2: 1e5$
C: 1.59e-9$

Z2: ratsimp((R2*(1/(s*C)))/(R2+(1/(s*C))), s);      /* Parallelschaltung R2 // C */
b0: ratsimp(Z2/(R1+Z2), s);                         /* Spannungsteiler ... */
b1: -A;
b2: ratsimp(R1/(R1+Z2), s);                         /* Spannungsteiler ... */
b3: 1;
L1: ratsimp(b1*b2, s);                              /* Kreis */
P1: ratsimp(b0*b1*b3, s);                           /* Vorwärtspfad 1 */
D1: 1;                                              /* Kofaktor 1 */
D: ratsimp(1-L1, s);                                /* Graph- Determinate */
H: ratsimp(P1*D1/D, s);                             /* Übertragungsfunktion */
Hi: limit(H, A, inf);                               /* Grenzwert A -> unendlich */
```