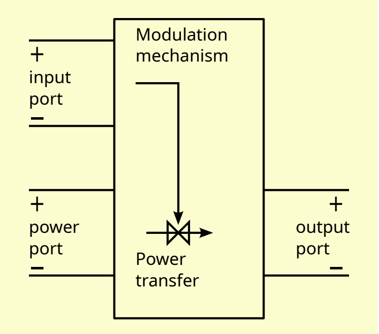
Amplifiers, concept, types and ideal behavioral models



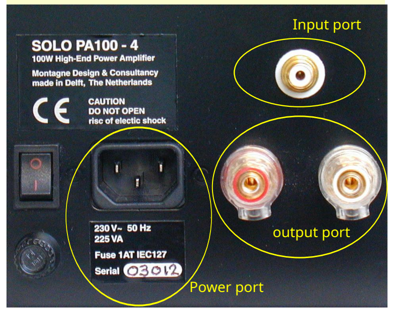
Definition

Amplifier object

Three electrical ports
- input port:
  connection to signal source
- output port:
  connection to load
- power port:
  connection to power supply

Amplification function
- provide load with accurate
  copy of source signal

Characteristic property
- Available power gain
  exceeds unity



Functional model

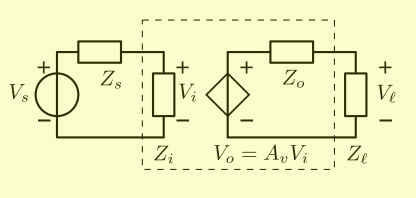
- Two-port
  input and output port only
- Active (delivers power)
- Linear, instantaneous,
  and time-invariant:

$$y(t) = A\,x(t)$$
$$A = \text{constant}$$

Example voltage amplifier

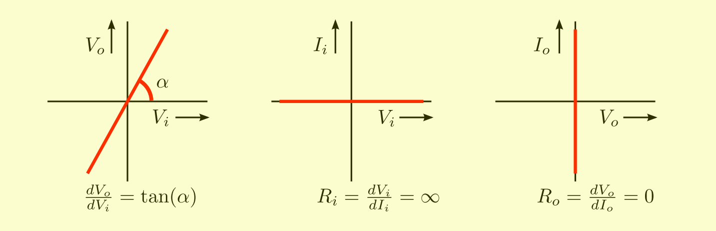
Source
- Information accurately related to open-circuit voltage
- Source impedance inaccurately known

Load
- Information accurately related to driving voltage
- Load impedance inaccurately known

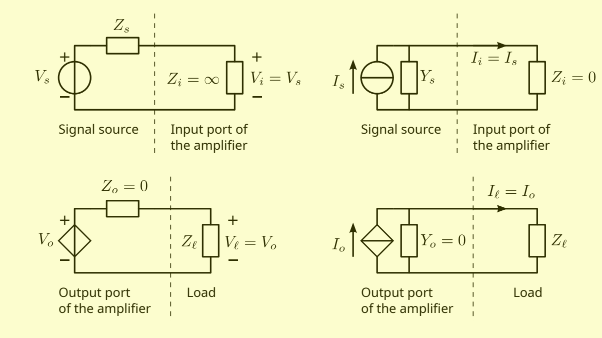


$$\frac{V_\ell}{V_s} = \frac{Z_i}{Z_i+Z_s} A_v \frac{Z_\ell}{Z_\ell+Z_o}$$

Ideal characteristics



Source termination impedance and load drive impedance



source
termination
impedance

load drive
impedance

Amplifier types

Follow from best source termination and load drive conditions for accurate signal transfer

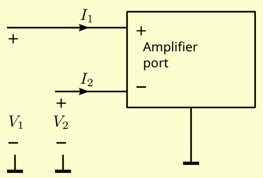
Type	Zi	Zo	A	B	C	D
voltage amplifier	infinite	0	A	0	0	0
transadmittance	infinite	infinite	0	B	0	0
transimpedance	0	0	0	0	C	0
current amplifier	0	infinite	0	0	0	D
voltage to voltage / current	infinite	Ro	A	B	0	0
current to voltage / current	0	Ro	0	0	C	D
voltage / curent to voltage	Ri	0	A	0	C	0
voltage / current to current	Ri	infinite	0	B	0	D
voltage / current to voltage /current	Ri	Ro	A	B	C	D

Port isolation properties

input-output	input-power	output-power	configuration
non-isolated	non-isolated	non-isolated	common-ground
non-isolated	non-isolated	isolated	x
non-isolated	isolated	non-isolated	differential receiver
non-isolated	isolated	isolated	floating supply
isolated	non-isolated	non-isolated	x
isolated	non-isolated	isolated	differential driver
isolated	isolted	non-isolated	x
isolated	isolated	isolated	differential receiver / driver

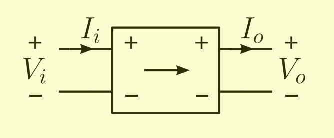
Floating port modeling and characterization

$$V_{cm} = \frac{V_1+V_2}{2}$$
$$V_{dm} = V_1 - V_2 \qquad Z_{cm} = \frac{V_{cm}}{I_{cm}}$$
$$I_{cm} = I_1 + I_2 \qquad Z_{dm} = \frac{V_{dm}}{I_{dm}}$$
$$I_{dm} = \frac{I_1-I_2}{2}$$



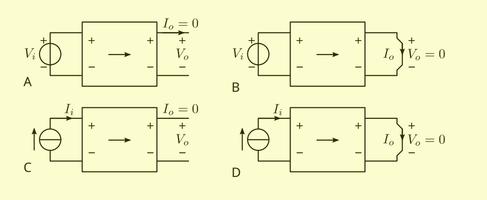
Modeling of ideal behavior (natural two-port)

Transmision-1 matrix representation



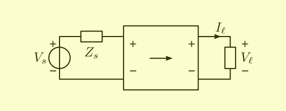
Anti-causal notation:

$$\begin{pmatrix} V_i \\ I_i \end{pmatrix} = \begin{pmatrix} A & B \\ C & D \end{pmatrix} \begin{pmatrix} V_o \\ I_o \end{pmatrix}$$

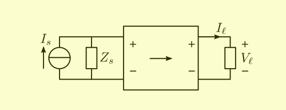


$$A = \frac{1}{\mu} = \left.\frac{V_i}{V_o}\right|_{I_o=0},$$
$$B = \frac{1}{\gamma} = \left.\frac{V_i}{I_o}\right|_{V_o=0},$$
$$C = \frac{1}{\zeta} = \left.\frac{I_i}{V_o}\right|_{I_o=0},$$
$$D = \frac{1}{\alpha} = \left.\frac{I_i}{I_o}\right|_{V_o=0}.$$

Source-to-load transfer



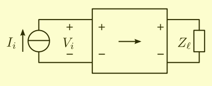
$$A_v = \frac{V_\ell}{V_s} = \frac{1}{A+B\frac{1}{Z_\ell}+CZ_s+D\frac{Z_s}{Z_\ell}}$$
$$A_y = \frac{I_\ell}{V_s} = \frac{1}{AZ_\ell+B+CZ_\ell Z_s+DZ_s}$$

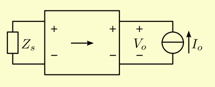


$$A_z = \frac{V_\ell}{I_s} = \frac{1}{A\frac{1}{Z_s}+B\frac{1}{Z_s Z_\ell}+C+D\frac{1}{Z_\ell}}$$
$$A_i = \frac{I_\ell}{I_s} = \frac{1}{A\frac{Z_\ell}{Z_s}+B\frac{1}{Z_s}+CZ_\ell+D}$$

Port impedances

$$Z_i = \frac{V_i}{I_i} = \frac{AZ_\ell+B}{CZ_\ell+D}$$





$$Z_o = \frac{V_i}{I_i} = \frac{DZ_s+B}{CZ_s+A}$$

For design purposes: source-to-load transfers and port impedances can be expressed in terms of the T1 matrix parameters (A,B,C,D) and the source and the load impedance.

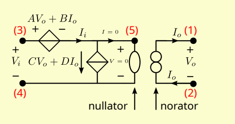
Unilateral amplifier types

Zero reverse transfer

$$AC = BD$$

Nullor and six non-unilateral types not listed in the table

Network model



MNA matrix stamp

$$\begin{pmatrix} 0\\0\\0\\0\\0\\0\\0 \end{pmatrix} = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ C & -C & 0 & 0 & 0 & D & 1 \\ -C & C & 0 & 0 & 0 & -D & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & -1 \\ 0 & 0 & 0 & 1 & -1 & 0 & 0 \\ -A & A & 1 & 0 & -1 & -B & 0 \end{pmatrix} \begin{pmatrix} V_1\\V_2\\V_3\\V_4\\V_5\\I_o\\I_i \end{pmatrix}$$

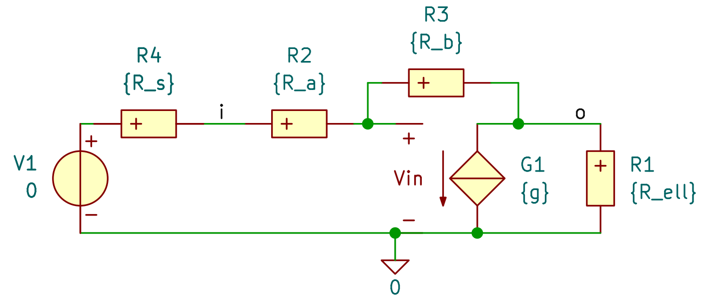
SLiCAP test bench for determination of the T1 matrix parameters

— Variable
— SLiCAP or Python keyword
— Comment
— String variable

```
#!/usr/bin/env python3
# -*- coding: utf-8 -*-

import SLiCAP as sl
import sympy as sp

sl.initProject('Circuit T1 matrix solutions')

files = ['cir-1']

# In all circuit files, use:

# Signal source: V1
# Load resistor: R1; model = 'r', value = 'R_ell'
# The name of the input node of the circuit: i'
# The name of the output node of the circuit: 'o'

# Define the symbolic variables

# The load resistance:

R_s, R_ell = sp.symbols('R_s, R_ell')

# The input vector:

ViIi = sp.Matrix([[sp.Symbol('V_i')], [sp.Symbol('I_i')]])

# The output vector:

VoIo = sp.Matrix([[sp.Symbol('V_o')], [sp.Symbol('I_o')]])

# Calculate and display the T1 matrix equation of the circuit(s):

for fileName in files:
    kicadFile = 'kicad/' + fileName + '/' + fileName + '.kicad_sch'
    cir = sl.makeCircuit(kicadFile)

    V_i =  sl.doLaplace(cir, source='V1', detector='V_i').laplace
    I_i = -sl.doLaplace(cir, source='V1', detector='I_V1').laplace
    V_o =  sl.doLaplace(cir, source='V1', detector='V_o').laplace
    I_o =  sl.doLaplace(cir, source='V1', detector='I_R1').laplace

    # Calculate the T1 parameters

    A = sp.simplify(sp.limit(V_i/V_o, R_ell, 'oo'))
    B = sp.simplify(sp.limit(V_i/I_o, R_ell, 0))
    C = sp.simplify(sp.limit(I_i/V_o, R_ell, 'oo'))
    D = sp.simplify(sp.limit(I_i/I_o, R_ell, 0))

    # Create the T1 matrix

    T = sp.Matrix([[A, B], [C, D]])

    # Store the matrices in a result object for displaying on an HTML page

    result    = sl.doMatrix(cir)
    result.M  = T
    result.Iv = ViIi
    result.Dv = VoIo

    # Calculate all six transfers

    Z_i = sp.simplify((A*R_ell+B)/(C*R_ell+D))          # Input impedance
    Z_o = sp.simplify((D*R_s+B)/(C*R_s+A))              # Output impedance
    A_v = sp.simplify(1/(A+B/R_ell+C*R_s+D*R_s/R_ell)) # Voltage gain
    A_y = sp.simplify(1/(A*R_ell+B+C*R_s*R_ell+D*R_s)) # Transadmittance
    A_z = sp.simplify(1/(A/R_s+B/R_ell/R_s+C+D/R_ell)) # Transimpedance
    A_i = sp.simplify(1/(A*R_ell/R_s+B/R_s+C*R_ell+D)) # Current gain

    # HTML Report ###############################################################

    sl.htmlPage('Determination of T1 matrix parameters')
    sl.head2html('Circuit diagram')
    sl.img2html(fileName + '.svg', 500)
    sl.head2html('T1 matrix ')
    sl.text2html('The T1 matrix of the device under test is found as:')
    sl.eqn2html('T', T)
    sl.text2html('The matrix equation for the two-port (DUT) is found as:')
    sl.matrices2html(result)
    sl.head2html("Input and output impedances")
    sl.eqn2html("Z_i", Z_i)
    sl.eqn2html("Z_o", Z_o)
    sl.head2html("Transfers")
    sl.eqn2html("A_v", A_v)
    sl.eqn2html("A_y", A_y)
    sl.text2html("For $A_z$ and $A_i$, the voltage source V1 in series with " +
                 "R4 is replaced with a current source in parallel with R4.")
    sl.eqn2html("A_z", A_z)
    sl.eqn2html("A_i", A_i)
```

Determination of T1 matrix parameters

Circuit diagram



T1 matrix

The T1 matrix of the device under test is found as:

$$T = \begin{bmatrix} \frac{-R_a g-1}{R_b g-1} & \frac{-R_a-R_b}{R_b g-1} \\ -\frac{g}{R_b g-1} & -\frac{1}{R_b g-1} \end{bmatrix} \tag{1}$$

The matrix equation for the two-port (DUT) is found as:

Matrix equation:

$$\begin{bmatrix} V_i \\ I_i \end{bmatrix} = \begin{bmatrix} \frac{-R_a g-1}{R_b g-1} & \frac{-R_a-R_b}{R_b g-1} \\ -\frac{g}{R_b g-1} & -\frac{1}{R_b g-1} \end{bmatrix} \cdot \begin{bmatrix} V_o \\ I_o \end{bmatrix} \tag{2}$$

Input and output impedances

$$Z_i = \frac{R_a + R_b + R_\ell\left(R_a g + 1\right)}{R_\ell g + 1} \tag{3}$$

$$Z_o = \frac{R_a + R_b + R_s}{R_a g + R_s g + 1} \tag{4}$$

Transfers

$$A_v = -\frac{R_\ell\left(R_b g - 1\right)}{R_a + R_b + R_\ell\left(R_a g + R_s g + 1\right) + R_s} \tag{5}$$

$$A_y = \frac{-R_b g + 1}{R_a + R_b + R_\ell R_s g + R_\ell\left(R_a g + 1\right) + R_s} \tag{6}$$

For A_z and A_i, the voltage source V1 in series with R4 is replaced with a current source in parallel with R4.

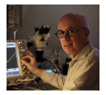
$$A_z = \frac{R_\ell R_s\left(-R_b g + 1\right)}{R_a R_\ell g + R_a + R_b + R_\ell R_s g + R_\ell + R_s} \tag{7}$$

$$A_i = \frac{R_s\left(-R_b g + 1\right)}{R_a R_\ell g + R_a + R_b + R_\ell R_s g + R_\ell + R_s} \tag{8}$$

Go to cir-1_index

For documentation, examples, support, updates and courses please visit: analog-electronics.tudelft.nl

Last project update: 2024-10-22 11:36:22



(c) 2021 Anton Montagne
anton@montagne.nl
https://www.analog-electronics.eu