Politécnica UFRJ

# Instrumentação e Técnicas de Medidas
# Trabalho 2 - 2025.02 - Versão 3

Universidade Federal do Rio de Janeiro, 26 de novembro de 2025
Departamento de Engenharia Eletrônica e de Computação
Professor: João Victor da Fonseca Pinto

## Contrato de Prestação de Serviço

**CONTRATANTE:** Electronic Brazil Devices, inscrita no CNPJ sob o nº 11.234.567/0001-99, com sede na Rua da Inovação, nº 123, Bairro Tecnológico, Cidade Fictícia, representada neste ato por **Moreira Grandão**, Diretor fictício de Pesquisa.

**CONTRATADA:** Laboratório de Processamento e Sinais (LPS) da Universidade Federal do Rio de Janeiro (UFRJ), com sede na Av. Athos da Silveira Ramos, 149 Cidade Universitária da UFRJ, llha do Fundão, Centro de Tecnologia, CEP 21.941-914 – Rio de Janeiro – RJ , sala H220, representada neste ato por **João Victor da Fonseca Pinto**, Pesquisador Associado e professor adjunto do Departamento de Eletrônica (DEL) da Escola Politécnica da Universidade Federal do Rio de Janeiro (UFRJ).

As partes acima qualificadas, por meio do presente instrumento particular e na melhor forma de direito, celebram o presente Contrato de Prestação de Serviços, regido pelas cláusulas e condições a seguir:

### CLÁUSULA PRIMEIRA - OBJETO DO CONTRATO

O presente contrato tem como objeto a prestação de serviços técnicos-científicos pela **CONTRATADA** à **CONTRATANTE** para o desenvolvimento de um protótipo de simulador de circuitos elétricos.

### CLÁUSULA SEGUNDA - ESCOPO DO PROJETO

A **CONTRATADA** se compromete a desenvolver um protótipo de simulador de circuitos elétricos que atenda aos seguintes requisitos técnicos, conforme detalhado no **Anexo I – Especificações Técnicas e Requisitos de Projeto**.

### CLÁUSULA TERCEIRA - AVALIAÇÃO

A **CONTRATANTE** nomeia e outorga poderes de avaliação ao Professor **João Victor da Fonseca Pinto**, que atuará como auditor técnico e avaliador do projeto. Todas as decisões e pareceres técnicos por ele emitidos sobre o cumprimento dos requisitos e a qualidade do trabalho deverão ser integralmente acatados pela **CONTRATANTE**.

### CLÁUSULA QUARTA - EQUIPE DO PROJETO

A equipe de desenvolvimento será composta por até 3 (três) alunos da turma de **Instrumentação e Técnicas de Medidas – EEL710 do ano de 2025, segundo período**, sob a supervisão técnica de **João Victor da Fonseca Pinto**, professor do DEL/UFRJ.

## CLÁUSULA QUINTA - PRAZO DE EXECUÇÃO

O prazo de execução para a entrega do protótipo deverá ser maior que 2 (dois) meses, a contar da data de assinatura deste contrato. O protótipo deverá ser entregue até às **23 (vinte e três) horas, 59 (cinquenta e nove) minutos e 59 (cinquenta e nove) segundos** do prazo estipulado em contrato na plataforma ***Google ClassRoom*** na tarefa correspondente ao **trabalho 2**.

## CLÁUSULA SEXTA - APRESENTAÇÃO E AVALIAÇÃO

Cada equipe deverá realizar uma apresentação do projeto com duração máxima de 30 (trinta) minutos. Esta apresentação ocorrerá na semana imediatamente após à data de entrega, em dia e horário a serem definidos em comum acordo com o **outorgado**.

## CLÁUSULA SÉTIMA - VALOR DO PROJETO

O valor total do protótipo, em termos de pontuação, será de no máximo 10 (dez) pontos. A distribuição desses pontos será realizada pelo outorgado, **João Victor da Fonseca Pinto**, com base nos critérios estabelecidos no Anexo I - Requisitos Técnicos do Projeto. O cumprimento de cada requisito será auditado e pontuado de acordo com a sua complexidade e relevância para o projeto. Durante a apresentação, cada equipe terá a possibilidade de abordar cada um dos itens especificados no anexo.

## CLÁUSULA OITAVA - CONFORMIDADE

A **CONTRATANTE** se compromete a fornecer todos os arquivos de referência, curvas e *netlists* necessários para a execução dos testes no ato da assinatura deste contrato.

## CLÁUSULA NONA - VALOR JURÍDICO

Para todos os efeitos, esse contrato é fictício e tem como objetivo apresentar aos alunos da turma de Instrumentação e Técnicas de Medidas - EEL710, todas as etapas do desenvolvimento de um projeto.

### Disposições Gerais

O **CONTRATADO(A)** se compromete a desenvolver o projeto em conformidade com todos os requisitos técnicos, prazos e metodologias estabelecidos no Anexo I. Este contrato é regido pelas leis de Mordor. Qualquer controvérsia será resolvida amigavelmente em sala de aula.

# Anexo I - Especificações Técnicas e Requisitos de Projeto

Este documento, **Anexo I - Especificações técnicas e Requisitos de Projeto**, estabelece as especificações e exigências mandatórias para o desenvolvimento do *software*. As diretrizes aqui apresentadas visam assegurar a conformidade com as melhores práticas da indústria, a robustez do produto final e a transparência do processo de desenvolvimento.

## REQUISITO 1 - VERSIONAMENTO

É obrigatório o uso de um repositório público no GitHub. A gestão e o controle de versão do código devem ser realizados exclusivamente por meio desta plataforma.

### Especificações técnicas

- **Visibilidade:** O repositório deve ser configurado como público, permitindo que o acesso ao código-fonte, histórico de *commits*, *issues* e *pull requests* seja irrestrito.
- **Colaboração:** Todas as contribuições, incluindo o desenvolvimento de novas funcionalidades e correções de *bugs*, devem ser submetidas através de *pull requests*.
- **Comunicação:** O sistema de *issues* do GitHub deve ser utilizado para registrar e acompanhar tarefas, reportar *bugs* e discutir o desenvolvimento do projeto.

## REQUISITO 2 - IMPLEMENTAÇÃO

O protótipo do *software* deverá ser implementado na linguagem de programação Python. A arquitetura do *software* deve obrigatoriamente utilizar o paradigma de Programação Orientada a Objetos para a representação de cada tipo de elemento de circuito suportado.

### Especificações técnicas

- **Classes de Elementos:** Para cada elemento de circuito previsto, uma classe específica deverá ser criada. O nome da classe deve ser claro e refletir o elemento representado (por exemplo: Resistor, Capacitor, Indutor).
- **Herança:** As classes de elementos deverão herdar de uma classe base ou de uma hierarquia de classes que defina a interface e o comportamento comum a todos os elementos de circuito. Esta classe base deverá conter os atributos e métodos essenciais para a análise nodal modificada, como: métodos para adicionar as contribuições do elemento às matrizes do sistema considerando os diferentes tipos de integrações (por exemplo: *backward* ou *forward* de Euler e trapézio).
- **Polimorfismo:** A utilização de polimorfismo é mandatória. As classes filhas deverão sobrescrever (*override*) os métodos da classe base para implementar o comportamento específico de cada elemento no contexto da análise nodal modificada. Por exemplo, o método que adiciona as contribuições à matriz de condutância deverá ser implementado de forma diferente para um resistor, um capacitor e um indutor.

- **Instanciação e Gerenciamento:** O *software* principal deverá ser capaz de criar instâncias (objetos) dessas classes de elementos, ler seus atributos e chamar seus métodos de forma genérica, sem precisar conhecer a classe específica de cada objeto.

# REQUISITO 3 - ANÁLISE NO TEMPO

É mandatório que o protótipo de *software* possua a funcionalidade de simulação no domínio do tempo (análise transiente) para a verificação do comportamento dinâmico de circuitos. A implementação do motor de simulação deve obedecer às seguintes diretrizes:

- **Método de Integração Numérica:** O algoritmo principal para a integração no tempo deve ser o método *Backward* de Euler. A aplicação deste método é compulsória para garantir a estabilidade e a robustez da simulação.
- **Métodos Adicionais:** O *software* deve, por **design**, prever a possibilidade de alternar para outros métodos de integração numérica, como o método *Forward* de Euler e o método do Trapézio. Embora a implementação completa desses métodos não seja exigida neste estágio, a arquitetura do código deve ser modular o suficiente para permitir sua futura inclusão.
- **Resolução de Elementos Não Lineares:** A solução de equações não lineares, essenciais para a simulação de componentes como diodos, transistores e resistores não lineares, deve ser realizada por meio do método de *Newton-Raphson*. A precisão e a convergência do algoritmo serão auditadas.

## Especificações técnicas

- **Primeiro passo:** Inicialmente deverá ser feita uma análise usando um passo muito menor que o normal, para completar a solução em $t = 0$ a partir das condições iniciais sobre indutores e capacitores. Tipicamente, o passo da análise deve ser dividido por mil na primeira interação.
- **Componentes não lineares:** O programa deve iniciar com uma avaliação preliminar da natureza dos componentes presentes no circuito, e informar ao motor da análise se o circuito possui elementos não lineares.
- **Iteração e Teste de Convergência:** Caso o circuito possua elementos não lineares, o método de *Newton-Raphson* deve ser aplicado iterativamente. Se o sistema não convergir após $N$ tentativas, um novo valor aleatório de $x(t)$ deve ser gerado. Tipicamente, $N = [20, 50]$ tentativas. A convergência do método deve ser testada observando a variação entre $x(t)$ e $x(t + \Delta t)$. Se o erro entre essas duas soluções for maior que uma tolerância mínima, então o método ainda não convergiu. Tipicamente a tolerância pode ser definida como $0,001$.
- **Controle de Tentativas:** Este processo de geração de novos valores aleatórios é repetido até um limite máximo de $M$ tentativas. Tipicamente, $M = 100$ tentativas.
- **Falha de Convergência:** Se o número de valores aleatórios de $x(t)$ atingir o limite $M$ sem que o sistema convirja, o algoritmo interrompe a execução. Neste ponto, é emitida uma mensagem de erro, indicando que o sistema é impossível de ser solucionado sob as condições atuais.

O pseudo-algoritmo a seguir representa uma possível implementação para a análise no tempo considerando todos os itens descritos acima.

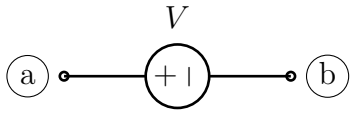
**Algorithm 1:** Análise no tempo

**Data:** $circuit, n\_variables, step\_time, step\_factor, max\_internal\_step, t_{max}, N, M$
**Result:** $answer, steps$
$answer \leftarrow []$; /* A list */
$steps \leftarrow []$; /* A list */
**while** $t \leq t_{max}$ **do**
  internal_step $\leftarrow 0$
  **if** $t = 0$ **then**
    $\Delta t \leftarrow step\_time/step\_factor$;
  **else**
    $\Delta t \leftarrow step\_time$;
  **while** $internal\_step \leq max\_internal\_step$ **do**
    $stop\_newton\_raphson \leftarrow False$;
    $n\_guesses \leftarrow 0$;
    $n\_newton\_raphson \leftarrow 0$;
    $x(t) \leftarrow zeros(number\_of\_variables)$ ; /* a vector */
    **while** *not* $stop\_newton\_raphson$ **do**
      **if** $n\_newton\_raphson = N$ **then**
        **if** $n\_guesses > M$ **then**
        $x(t) \leftarrow randomize(n\_variables)$;
        $n\_guesses \leftarrow n\_guesses + 1$;
        $n\_newton\_raphson \leftarrow n\_newton\_raphson + 1$;
      $A \leftarrow zeros(n\_variables, n\_variables)$ ; /* A square matrix */
      $b \leftarrow zeros(n\_variables)$ ; /* a vector */
      **for** *each element in circuit* **do**
        $A, b \leftarrow stamp(element, A, b, x(t))$ ; /* Add the stamp for each element */
      $x(t + \Delta t) \leftarrow solve(A, b, x(t))$ ; /* Solve the Ax=b linear system */
      $tolerance \leftarrow max(abs(x(t + \Delta t) - x(t)))$;
      **if** $is\_non\_linear$ *and* $(tolerance > max\_tolereance)$ **then**
        $x(t) \leftarrow x(t + \Delta t)$;
        $n\_newton\_raphson \leftarrow n\_newton\_raphson + 1$;
      **else**
        $stop\_newton\_rapshon \leftarrow True$;
    $circuit.update(x(t + \Delta t))$ ; /* Update all initial conditions */
    $internal\_step \leftarrow internal\_step + 1$;
  $internal\_step \leftarrow 0$;
  $answer.append(x(t + \Delta t))$;
  $steps.append(t)$;
  $t \leftarrow t + \Delta t$;

## REQUISITO 4 - ELEMENTOS SUPORTADOS

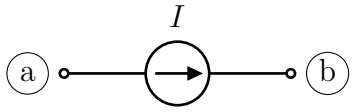
A funcionalidade do protótipo de software está condicionada ao seu suporte a um conjunto mínimo de elementos de circuito. É mandatório que o protótipo demonstre total capacidade de processar, simular e analisar circuitos que contenham, no mínimo, os componentes listados a seguir. A não conformidade com qualquer um dos itens desta lista será considerada uma falha crítica na avaliação técnica.

### Fonte de tensão



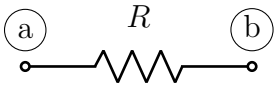

$$\begin{array}{c} \\ a \\ b \\ x \end{array}\!\!\begin{array}{c} \begin{array}{ccc} a & b & x \end{array} \\ \left[\begin{array}{ccc} . & . & +1 \\ . & . & -1 \\ -1 & +1 & . \end{array}\right] \end{array}\!\!\left[\begin{array}{c} v_a(t_0+\Delta t) \\ v_b(t_0+\Delta t) \\ j_x(t_0+\Delta t) \end{array}\right] = \left[\begin{array}{c} . \\ . \\ -V \end{array}\right]$$

### Fonte corrente




$$\begin{array}{c} \\ a \\ b \end{array}\!\!\begin{array}{c} \begin{array}{cc} a & b \end{array} \\ \left[\begin{array}{cc} . & . \\ . & . \end{array}\right] \end{array}\!\!\left[\begin{array}{c} v_a(t_0+\Delta t) \\ v_b(t_0+\Delta t) \end{array}\right] = \left[\begin{array}{c} -I \\ +I \end{array}\right]$$

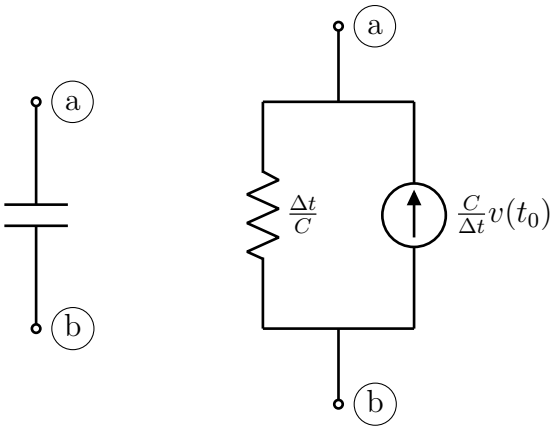
### Resistor




$$\begin{array}{c} \\ a \\ b \end{array}\!\!\begin{array}{c} \begin{array}{cc} a & b \end{array} \\ \left[\begin{array}{cc} +\frac{1}{R} & -\frac{1}{R} \\ -\frac{1}{R} & +\frac{1}{R} \end{array}\right] \end{array}\!\!\left[\begin{array}{c} v_a(t_0+\Delta t) \\ v_b(t_0+\Delta t) \end{array}\right] = \left[\begin{array}{c} . \\ . \end{array}\right]$$

### Capacitor (*backward* de Euler)

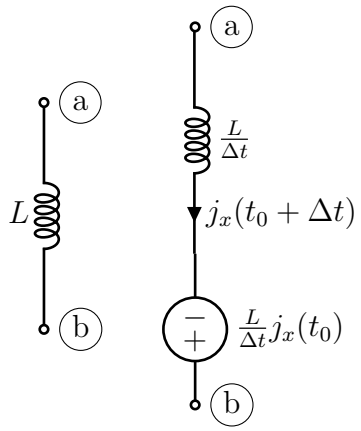
$$j(t_0+\Delta t) \approx \frac{C}{\Delta t} \cdot v(t_0+\Delta t) - \frac{C}{\Delta t} \cdot v(t_0)$$




$$\begin{array}{c} \\ a \\ b \end{array}\begin{array}{c} \begin{array}{cc} a & b \end{array} \\ \begin{bmatrix} +\frac{C}{\Delta t} & -\frac{C}{\Delta t} \\ -\frac{C}{\Delta t} & +\frac{C}{\Delta t} \end{bmatrix} \end{array}\begin{bmatrix} v_a(t_0+\Delta t) \\ v_b(t_0+\Delta t) \end{bmatrix} = \begin{bmatrix} +\frac{C}{\Delta t}v_{ab}(t_0) \\ -\frac{C}{\Delta t}v_{ab}(t_0) \end{bmatrix}$$

## Indutor (*backward* de Euler)

$$v_{ab}(t_0+\Delta t) \approx \frac{L}{\Delta t}j_x(t_0+\Delta t) + \frac{L}{\Delta t}j_x(t_0)$$




$$\begin{array}{c} \\ a \\ b \\ \hdashline x \end{array}\begin{array}{c} \begin{array}{ccc} a & b & x \end{array} \\ \left[\begin{array}{cc:c} . & . & +1 \\ . & . & -1 \\ \hdashline -1 & +1 & +\frac{L}{\Delta t} \end{array}\right] \end{array}\begin{bmatrix} v_a(t_0+\Delta t) \\ v_b(t_0+\Delta t) \\ \hdashline j_x(t_0+\Delta t) \end{bmatrix} = \begin{bmatrix} . \\ . \\ \hdashline +\frac{L}{\Delta t}j_x(t_0) \end{bmatrix}$$

**Fonte de tensão controlada por tensão:**

$$
\begin{array}{c} \\ a \\ b \\ c \\ d \\ x \end{array}
\begin{array}{c}
\begin{array}{cccc} a & b & c & d \end{array} \\
\left[\begin{array}{cccc:c}
. & . & . & . & +1 \\
. & . & . & . & -1 \\
. & . & . & . & . \\
. & . & . & . & . \\
\hdashline
-1 & +1 & A & -A & .
\end{array}\right]
\end{array}
\left[\begin{array}{c}
v_a(t_0+\Delta t) \\
v_b(t_0+\Delta t) \\
v_c(t_0+\Delta t) \\
v_d(t_0+\Delta t) \\
\hdashline
j_x(t_0+\Delta t)
\end{array}\right]
=
\left[\begin{array}{c}
. \\ . \\ . \\ . \\ \hdashline . \end{array}\right]
$$

**Fonte de corrente controlada por corrente**

$$
\begin{array}{c} \\ a \\ b \\ c \\ d \\ x \end{array}
\begin{array}{c}
\begin{array}{cccc} a & b & c & d \end{array} \\
\left[\begin{array}{cccc:c}
. & . & . & . & A_i \\
. & . & . & . & -A_i \\
. & . & . & . & +1 \\
. & . & . & . & -1 \\
\hdashline
. & . & -1 & +1 & .
\end{array}\right]
\end{array}
\left[\begin{array}{c}
v_a(t_0+\Delta t) \\
v_b(t_0+\Delta t) \\
v_c(t_0+\Delta t) \\
v_d(t_0+\Delta t) \\
\hdashline
j_x(t_0+\Delta t)
\end{array}\right]
=
\left[\begin{array}{c}
. \\ . \\ . \\ . \\ \hdashline . \end{array}\right]
$$

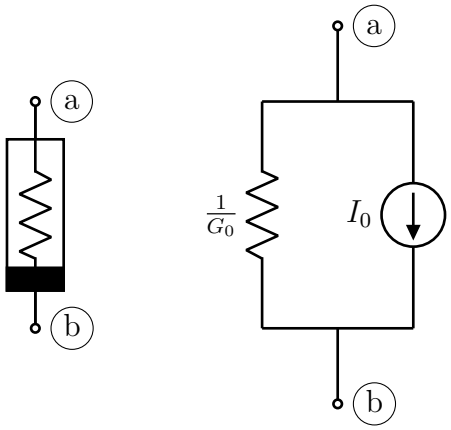
**Fonte de corrente controlada por tensão (transcondutância)**

$$
\begin{array}{c} \\ a \\ b \\ c \\ d \end{array}
\begin{array}{c}
\begin{array}{cccc} a & b & c & d \end{array} \\
\left[\begin{array}{cccc}
. & . & +G_m & -G_m \\
. & . & -G_m & +G_m \\
. & . & . & . \\
. & . & . & .
\end{array}\right]
\end{array}
\left[\begin{array}{c}
v_a(t_0+\Delta t) \\
v_b(t_0+\Delta t) \\
v_c(t_0+\Delta t) \\
v_d(t_0+\Delta t)
\end{array}\right]
=
\left[\begin{array}{c}
. \\ . \\ . \\ . \end{array}\right]
$$

**Fonte de tensão controlada por corrente (transresistência)**

$$
\begin{array}{c} \\ a \\ b \\ c \\ d \\ x \\ y \end{array}
\begin{array}{c}
\begin{array}{cccccc} a & b & c & d & x & y \end{array} \\
\left[\begin{array}{cccc:cc}
. & . & . & . & 1 & . \\
. & . & . & . & -1 & . \\
. & . & . & . & . & +1 \\
. & . & . & . & . & -1 \\
\hdashline
. & . & -1 & +1 & . & . \\
. & . & -1 & +1 & . & R_m
\end{array}\right]
\end{array}
\left[\begin{array}{c}
v_a(t_0+\Delta t) \\
v_b(t_0+\Delta t) \\
v_c(t_0+\Delta t) \\
v_d(t_0+\Delta t) \\
\hdashline
j_x(t_0+\Delta t) \\
j_y(t_0+\Delta t)
\end{array}\right]
=
\left[\begin{array}{c}
. \\ . \\ . \\ . \\ \hdashline . \\ . \end{array}\right]
$$

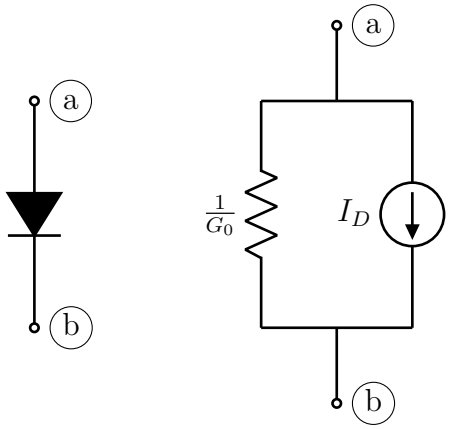
## Resistor não linear (com 3 segmentos)




$$\begin{cases} G_0 = (I_4 - I_3)/(V_4 - V_3)\ ,\ I_0 = I_4 - V_4, & \text{Se } v_{ab} > V_3, \\ G_0 = (I_3 - I_2)/(V_3 - V_2)\ ,\ I_0 = I_3 - V_3, & \text{Se } V_2 < v_{ab} \leq V_3, \\ G_0 = (I_2 - I_1)/(V_2 - V_1)\ ,\ I_0 = I_2 - V_2, & \text{Se } v_{ab} \leq V_2. \end{cases}$$

## Diodo

O modelo simplificado do diodo é dado por:



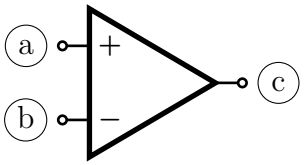

Inicialmente, calcula-se a diferença de potencial entre o nós do diodo através do vetor de tensões e correntes $x(t)$ da seguinte forma: $v_{ab} = x_a(t) - x_b(t)$. Se $v_{ab} > 0,9$ então $v_{ab} = 0,9$. A condutância $(G_0)$e a corrente $(I_D)$ do modelo são dadas por:

$$G_0 = \frac{I_S \cdot e^{v_{ab}/V_T}}{V_T}$$

$$I_D = I_S \cdot (e^{v_{ab}/V_T} - 1) - G_0 \cdot v_{ab}$$

onde $I_S = 3.7751345 \cdot 10^{-14}$ (corrente de saturação) e $V_T = 25 \cdot 10^{-3}$ (tensão térmica).

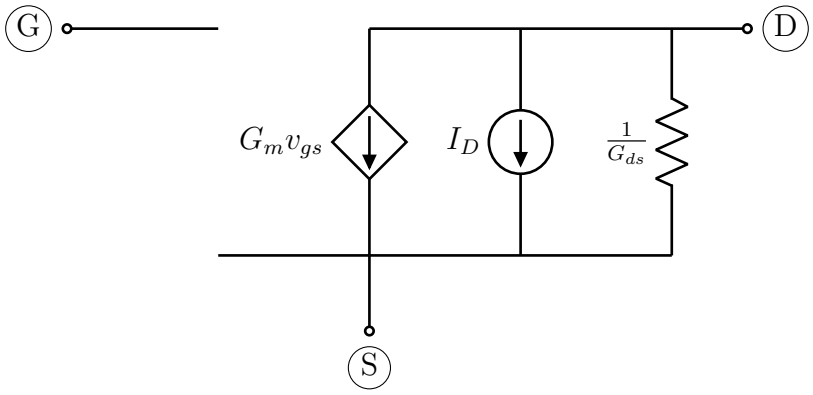
## Amplificador Operacional Ideal




$$
\begin{array}{c}
\\ 0 \\ a \\ b \\ c \\ x
\end{array}
\begin{array}{c}
\begin{array}{cccc} 0 & a & b & c \end{array} \\
\left[\begin{array}{cccc:c}
. & . & . & . & -1 \\
. & . & . & . & . \\
. & . & . & . & . \\
. & . & . & . & +1 \\
\hdashline
. & -1 & +1 & . & .
\end{array}\right]
\end{array}
\left[\begin{array}{c}
v_a(t_0+\Delta t) \\
v_b(t_0+\Delta t) \\
v_c(t_0+\Delta t) \\
v_d(t_0+\Delta t) \\
\hdashline
j_x(t_0+\Delta t)
\end{array}\right]
=
\left[\begin{array}{c}
. \\ . \\ . \\ . \\ \hdashline . \end{array}\right]
$$

## MOSFET

O modelo simplificado do MOSFET é dado por:




Inicialmente, deve-se armazenar os nós do dreno, porta e fonte em variáveis temporárias, por exemplo, $d$, $g$ e $s$ respectivamente. Em seguida:

- **Canal tipo N:** Verificar se a tensão no dreno é menor que na fonte. Se isso acontecer, deve-se inverter as posições entre dreno e fonte nas respectivas variáveis temporárias ($d$ = fonte e $s$ = dreno). Se for a primeira interação, deve-se forçar $v_{gs} = 2$, caso contrário $v_{gs}$ será a diferença de potencial entre o dreno e a fonte. Os potenciais serão calculados através do vetor de tensões e correntes $x(t)$ da seguinte forma: $v_{gs} = x_g(t) - x_s(t)$ e $v_{ds} = x_d(t) - x_s(t)$.
- **Canal tipo P:** Verificar se a tensão no dreno é maior que na fonte. Se isso acontecer, deve-se inverter as posições entre dreno e fonte nas respectivas variáveis temporárias. Se for a primeira interação, deve-se forçar $v_{gs} = -2$, caso contrário $v_{gs}$ será o negativo da diferença de potencial entre o dreno e a fonte. Os potenciais serão calculados através do vetor de tensões e correntes $x_t$, da seguinte forma: $v_{gs} = -1 \cdot (x_g(t) - x_s(t))$ e $v_{ds} = -1 \cdot (x_d(t) - x_s(t))$.

Após encontrar os valores de $v_{gs}$ e $v_{ds}$, deve-se calcular a corrente de dreno e as condutâncias do modelo utilizando as equações do modo de operação correto:

- **Saturação ( $v_{ds} > v_{gs} - V_{th}$ e $v_{gs} > V_{th}$):**

$$G_m = K\left(\frac{W}{L}\right) \cdot (2 \cdot (v_{gs} - V_{th})(1 + \lambda v_{ds}))$$

$$i_D = K\left(\frac{W}{L}\right)(v_{gs} - V_{th})^2(1 + \lambda v_{ds})$$

$$G_{ds} = K\left(\frac{W}{L}\right)(v_{gs} - V_{th})^2\lambda$$

$$I_D = i_D - G_m v_{gs} - G_{ds} v_{ds}$$

- **Triodo ( $v_{ds} \leq v_{gs} - V_{th}$ e $v_{gs} > V_{th}$):**

$$i_D = K\left(\frac{W}{L}\right)[2(v_{gs} - V_{th})v_{ds} - v_{ds}^2](1 + \lambda v_{ds})$$

$$G_m = K\left(\frac{W}{L}\right)[2v_{ds}(1 + \lambda v_{ds})]$$

$$G_{ds} = K\left(\frac{W}{L}\right)\left[2(v_{gs} - V_{th}) - 2v_{ds} + 4\lambda(v_{gs} - V_{th}) - 3\lambda v_{ds}^2\right]$$

$$I_D = i_D - G_m v_{gs} - G_{ds} v_{ds}$$

- **Corte ($v_{gs} \leq V_{th}$):**

$$I_D = i_d = G_m = G_{ds} = 0$$

## REQUISITO 5 - SUPORTE A NETLISTS

O protótipo do *software* deve oferecer duas interfaces distintas e funcionais para a construção e simulação de circuitos elétricos: uma interface programática baseada em Python e uma interface de arquivo, que lê a descrição do circuito de um arquivo de texto formatado (*netlist*). O sistema deve ser capaz de processar circuitos complexos e executar a simulação no tempo com base em ambas as abordagens.

### Especificações técnicas para a interface programática:

- O software deve expor um conjunto de classes e funções públicas que permitam ao usuário construir um circuito diretamente via código Python.
- Deve existir uma classe principal, Circuito, que atue como um contêiner para os elementos do circuito.
- A classe Circuito deve fornecer métodos para adicionar elementos, conectar nós e configurar parâmetros de simulação.

- A sintaxe para a criação do circuito deve ser clara e intuitiva, permitindo que o usuário defina os nós, o tipo de elemento e seus valores de forma encadeada.
- O software deve incluir uma função de simulação que receba o objeto Circuito e execute a análise, retornando os resultados.
- O software deve permitir salvar o circuito construído através da interface programática em um arquivo *netlist*.

## Especificações técnicas para a interface de arquivo (*netlist*):

A primeira linha deve ser o número de nós do circuito seguida pela descrição do circuito, com um elemento por linha. A primeira letra determina o tipo de elemento. Os nós podem ser inteiros ou *labels* (*strings*). Para cada elemento suportado, o seguinte formato deve ser implementado:

- **Comentário:**
  - Primeiro carácter: *
  - Primeira palavra (excluindo primeiro carácter): comentário
- **Resistor:**
  - Primeiro carácter: R
  - Primeira palavra (excluindo primeiro carácter): nome do elemento.
  - Segunda palavra: nó positivo.
  - Terceira palavra: nó negativo.
  - Quarta palavra: resistência.
- **Indutor:**
  - Primeiro carácter: L
  - Primeira palavra (excluindo primeiro carácter): nome do elemento.
  - Segunda palavra: nó positivo.
  - Terceira palavra: nó negativo.
  - Quarta palavra: indutância.
  - Quinta palavra (opcional): valor da corrente inicial do indutor. Quando houver essa opção, essa palavra deve começar com 'IC=' seguida pelo valor da condição inicial.
- **Capacitor:**
  - Primeiro carácter: C
  - Primeira palavra (excluindo primeiro carácter): nome do elemento.
  - Segunda palavra: nó positivo.
  - Terceira palavra: nó negativo.

  - Quarta palavra: capacitância.
  - Quinta palavra (opcional): valor da tensão inicial do capacitor. Quando houver essa opção, essa palavra deve começar com 'IC=' seguida pelo valor da condição inicial.

- **Resistor não linear:**
  - Primeiro carácter: N
  - Primeira palavra (excluindo primeiro carácter): nome do elemento.
  - Segunda palavra: nó positivo.
  - Terceira palavra: nó negativo.
  - Quarta palavra: $V_1$.
  - Quinta palavra: $I_1$.
  - Sexta palavra: $V_2$.
  - Sétima palavra: $I_2$.
  - Oitava palavra: $V_3$
  - Nona palavra: $I_3$
  - Décima palavra: $V_4$
  - Décima primeira palavra: $I_4$

- **Fonte de tensão controlada por tensão:**
  - Primeiro carácter: E
  - Primeira palavra (excluindo primeiro carácter): nome do elemento.
  - Segunda palavra: nó de tensão positivo.
  - Terceira palavra: nó de tensão negativo.
  - Quarta palavra: nó de controle positivo.
  - Quinta palavra: nó de controle negativo.
  - Sexta palavra: ganho de tensão.

- **Fonte de corrente controlada por corrente:**
  - Primeiro carácter: F
  - Primeira palavra (excluindo primeiro carácter): nome do elemento.
  - Segunda palavra: nó de corrente positivo.
  - Terceira palavra: nó de corrente negativo.
  - Quarta palavra: nó de controle positivo .
  - Quinta palavra: nó de controle negativo.
  - Sexta palavra: ganho de corrente.

- **Fonte de corrente controlada por tensão (transcondutância):**

  - Primeiro carácter: G
  - Primeira palavra (excluindo primeiro carácter): nome do elemento.
  - Segunda palavra: nó de corrente positivo.
  - Terceira palavra: nó de corrente negativo.
  - Quarta palavra: nó de controle positivo.
  - Quinta palavra: nó de controle negativo.
  - Sexta palavra: transcondutância.

- **Fonte de tensão controlada por corrente (transresistência):**
  - Primeiro carácter: H
  - Primeira palavra (excluindo primeiro carácter): nome do elemento.
  - Segunda palavra: nó de tensão positivo.
  - Terceira palavra: nó de tensão negativo.
  - Quarta palavra: nó de controle positivo.
  - Quinta palavra: nó de controle negativo.
  - Sexta palavra: transresistência.

- **Amplificador Operacional Ideal:**
  - Primeiro carácter: O
  - Primeira palavra (excluindo primeiro carácter): nome do elemento.
  - Segunda palavra: nó de controle positivo.
  - Terceira palavra: nó de controle negativo.
  - Quarta palavra: nó de saída.

- **Diodo:**
  - Primeiro carácter: D
  - Primeira palavra (excluindo primeiro carácter): nome do elemento.
  - Segunda palavra: nó positivo.
  - Terceira palavra: nó negativo.

- **MOSFET:**
  - Primeiro carácter: M
  - Primeira palavra (excluindo primeiro carácter): nome do elemento.
  - Segunda palavra: dreno.
  - Terceira palavra: porta.
  - Quarta palavra: fonte.
  - Quinta palavra: Tipo do canal (N ou P).

  - Sexta palavra: $W$.
  - Sétima palavra: $L$.
  - Oitava palavra: $\lambda$
  - Nona palavra: $K$
  - Décima palavra: $V_{th}$

- **Fonte de corrente:**
  - Primeiro carácter: I
  - Primeira palavra (excluindo primeiro carácter): nome do elemento.
  - Segunda palavra: nó de corrente positivo.
  - Terceira palavra: nó de corrente negativo.
  - Palavras seguintes: parâmetros da fonte de corrente conforme o tipo de fonte utilizada.

- **Fonte de tensão:**
  - Primeiro carácter: V
  - Primeira palavra (excluindo primeiro carácter): nome do elemento.
  - Segunda palavra: nó de tensão positivo.
  - Terceira palavra: nó de tensão negativo.
  - Palavras seguintes: parâmetros da fonte de corrente conforme o tipo de fonte utilizada.

Os parâmetros para as fontes de tensão e corrente independentes (a partir da terceira palavra) devem seguir o formato SPICE, com a capacidade de suportar fontes contínuas, senoidais e em pulso. A configuração de cada uma das fontes é detalhada a seguir:

- **Fonte contínua:**
  - Quarta palavra: DC
  - Quinta palavra: valor do nível contínuo do sinal.

- **Fonte senoidal:**
  - Quarta palavra: SIN
  - Quinta palavra: valor do nível contínuo do sinal.
  - Sexta palavra: amplitude do sinal.
  - Sétima palavra: frequência em Hertz.
  - Oitava palavra: atraso do sinal.
  - Nona palavra: coeficiente de amortecimento do sinal.
  - Décima palavra: defasagem do sinal em graus.
  - Décima primeira palavra: número de ciclos.

A fonte senoidal tem o seguinte formato:

$$x(t) = A_0 + A \cdot e^{-\alpha(t-t_a)} sen\left(2\pi f(t-t_a) + \frac{\pi}{180}\phi\right)$$

onde $A_0$ é o nível contínuo, $A$ é a amplitude do sinal, $f$ é a frequência em Hertz, $t_a$ o atraso em segundos, $\alpha$ o amortecimento e $\phi$ a defasagem em graus. Antes de $t = t_a$ ou após o número de ciclos, $x(t)$ tem o valor inicial ou final respectivamente, de forma a não criar descontinuidades, dado por:

$$x(t) = A_0 + A \cdot sin\left(\frac{\pi\phi}{180}\right)$$

- **Fonte pulsada:**
  - Quarta palavra: PULSE
  - Quinta palavra: valor do nível de amplitude 1.
  - Sexta palavra: valor do nível de amplitude 2.
  - Sétima palavra: atraso do sinal.
  - Oitava palavra: tempo de subida.
  - Nona palavra: tempo de descida.
  - Décima palavra: tempo de pulso ligado.
  - Décima primeira palavra: período do sinal.
  - Décima segunda palavra: número de ciclos.

  A fonte de pulso inicia sua operação em amplitude 1, mantendo-se neste nível durante o tempo de atraso. Após este período, a amplitude é elevada para o nível 2 por meio de uma variação linear, que ocorre ao longo do tempo de subida. A fonte mantém-se na amplitude 2 durante o tempo ligado, para então retornar à amplitude 1 por meio de uma variação linear no tempo de descida. O ciclo de operação se repete de acordo com o período e o número de ciclos predefinidos. A operação é finalizada na amplitude 1. É importante observar que os tempos de subida e de descida podem ser zero. Nesse caso, o programa deve utilizar o tempo do passo.

A última linha do arquivo é reservada para os parâmetros da simulação. Essa linha deve ter o seguinte formato:

- **Parâmetros de simulação (última linha):**
  - Primeira palavra: .TRAN para análise de transiente.
  - Segunda palavra: tempo total de simulação.
  - Terceira palavra: passo de simulação.
  - Quarta palavra: Método de integração utilizado na simulação com as seguintes opções: *backward* de Euler (BE), *forward* de Euler (FE) ou trapezoidal (TRAP).
  - Quinta palavra: número de passos internos.
  - Sexta palavra (opcional): 'UIC' que significa usar as condições inicias durante a simulação. Tipicamente, esse modo de simulação é sempre utilizado.

### Especificações técnicas de saída

- Independentemente da interface utilizada para a construção do circuito (programática ou netlist), o resultado da simulação deve ser padronizado.
- A saída deve conter as tensões nodais e, se aplicável, as correntes de variáveis de estado (indutores, fontes de tensão) em função do tempo.
- Os resultados devem ser armazenados em uma estrutura de dados de fácil acesso (por exemplo: um *dataframe* ou dicionário) para posterior análise ou plotagem.

## REQUISITO 6 - REPRODUTIBILIDADE

É obrigatório implementar um *pipeline* de Integração e Entrega Contínuas (CI/CD) para o projeto. Esta *pipeline* deve automatizar a validação do código, garantindo a sua robustez e a conformidade com as especificações.

### Especificações técnicas

- **Ferramentas de Automação:** O *pipeline* de CI/CD deve ser configurado utilizando GitHub *Actions* em conjunto com GitHub *runners*. Um arquivo de configuração .yml deve ser incluído no repositório, definindo os gatilhos e as etapas de execução da *pipeline*.
- **Testes:** A suíte de testes deve ser construída utilizando o *framework* Pytest. Os testes precisam cobrir as funcionalidades essenciais do *software*, verificando a correta aplicação da lógica e a integridade dos dados. A *pipeline* de CI/CD deve ser configurada para executar essa suíte de testes automaticamente a cada *push* ou *pull request* submetido ao repositório.
- **Finalidade:** A implementação dessa *pipeline* tem como objetivo principal automatizar a detecção de *bugs* e regressões, assegurando que novas alterações no código não comprometam a funcionalidade existente.
- **Referências:** Os arquivos de referência (*netlists* e suas respectivas saídas), necessários para a execução completa dos testes, serão fornecidos no ato da assinatura do contrato. A correta utilização desses arquivos na rotina de testes é crucial para a aprovação final do projeto. As saídas esperadas para cada um dos testes fornecidos estão especificados no **Anexo II - Descrição dos Testes para Avaliação do Simulador**.

## REQUISITO 7 - DOCUMENTAÇÃO

É obrigatório criar uma documentação completa para o protótipo do *software*. A documentação deve ser desenvolvida utilizando ferramentas de código amplamente utilizadas pela indústria. A qualidade da documentação será um dos critérios de avaliação mais importantes.

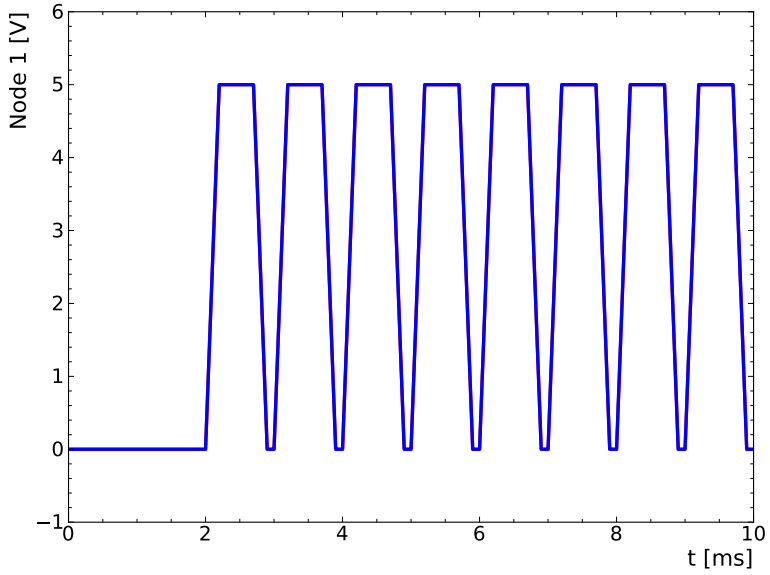
### Instruções para a Documentação:

- **Formato:** A documentação deve ser elaborada em formato Markdown (.md), um padrão da indústria para arquivos READMEs.

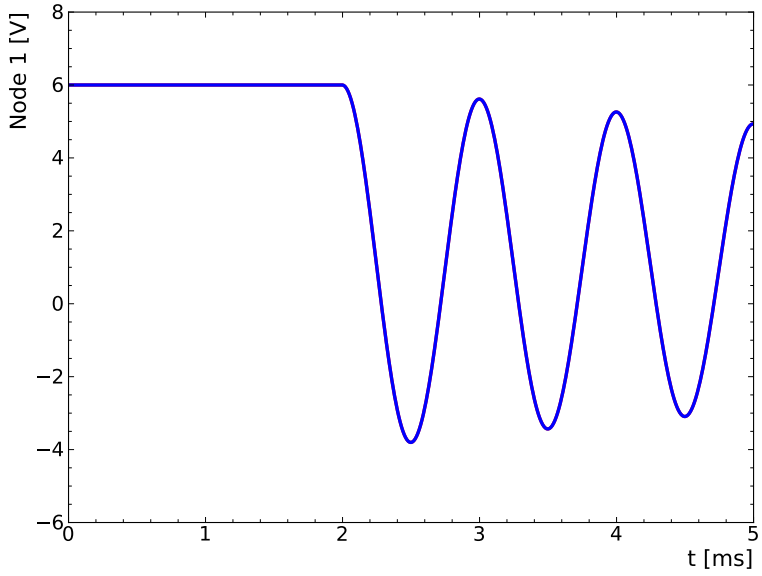
- **Ferramentas:** É obrigatório o uso de ferramentas de *markup* e renderização de código, tais como o Doxygen ou Sphinx, para gerar a documentação de forma automatizada a partir do código-fonte. Isso garante que a documentação esteja sempre sincronizada com o código.
- **Conteúdo:** A documentação deve incluir os seguintes itens:
  - **Visão Geral do Projeto:** Uma descrição concisa do propósito do software e suas principais funcionalidades.
  - **Estrutura do Código:** Um diagrama ou descrição detalhada da arquitetura do software, explicando a função de cada módulo, classe ou arquivo.
  - **Instalação e Configuração:** Instruções claras e passo a passo para instalar e configurar o programa, incluindo as dependências necessárias.
  - **Uso e Exemplos:** Exemplos de uso prático do *software*, com *snippets* de código ou comandos que demonstrem como interagir com o protótipo.
  - **API (se aplicável):** Se o protótipo incluir uma API, a documentação deve detalhar os *endpoints*, parâmetros de entrada, formatos de saída e códigos de erro.
  - **Testes:** Uma seção dedicada a descrever como executar os testes de continuidade implementados no projeto, explicando o objetivo de cada suíte de testes.
- **Diagrama de Classes:** A documentação do projeto deve incluir um diagrama de classes que ilustre a hierarquia, os atributos e os métodos das classes de elementos, validando o uso de herança e encapsulamento.
- **Localização:** A documentação deve estar localizada no diretório raiz do repositório do projeto, em um arquivo chamado README.md. A documentação gerada pelo Doxygen ou Sphinx deve ser adicionada a uma pasta específica, como /docs, e ser facilmente acessível.

# Anexo II - Descrição dos Testes para Avaliação do Simulador

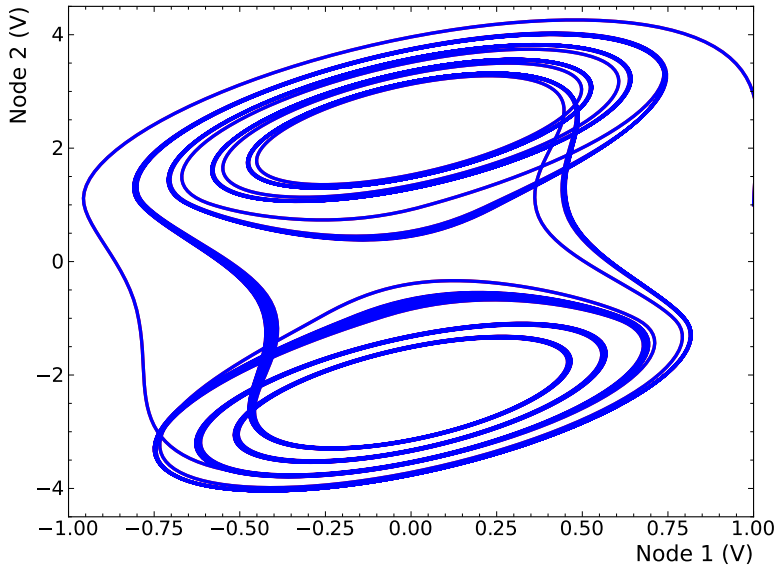
## TESTE 1 - SINAL PULSADO (pulse.net)



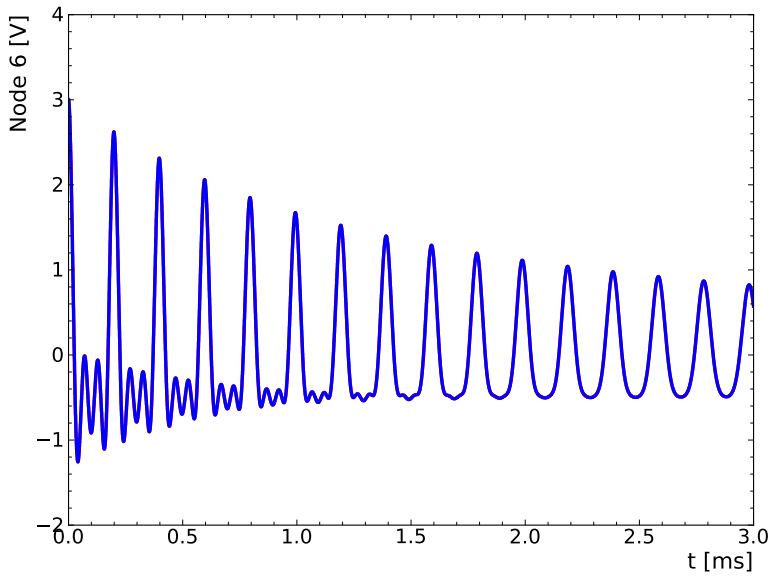

## TESTE 2 - SINAL SENOIDAL (sinusoidal.net)



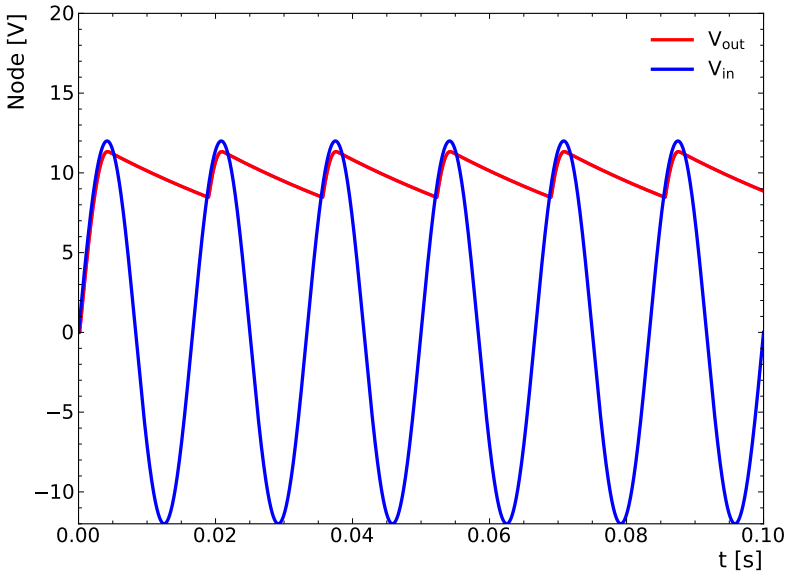
## TESTE 3 - CIRCUITO CHUA (chua.net)



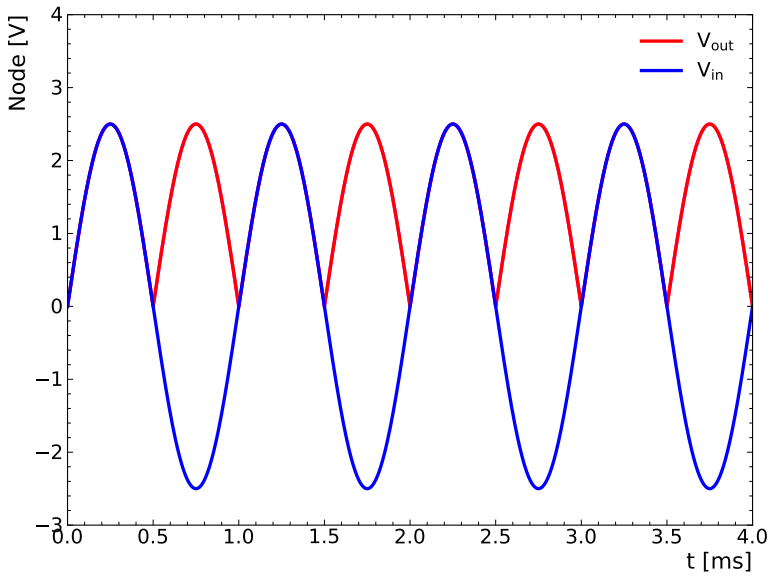

## TESTE 4 - CIRCUITO LC (lc.net)



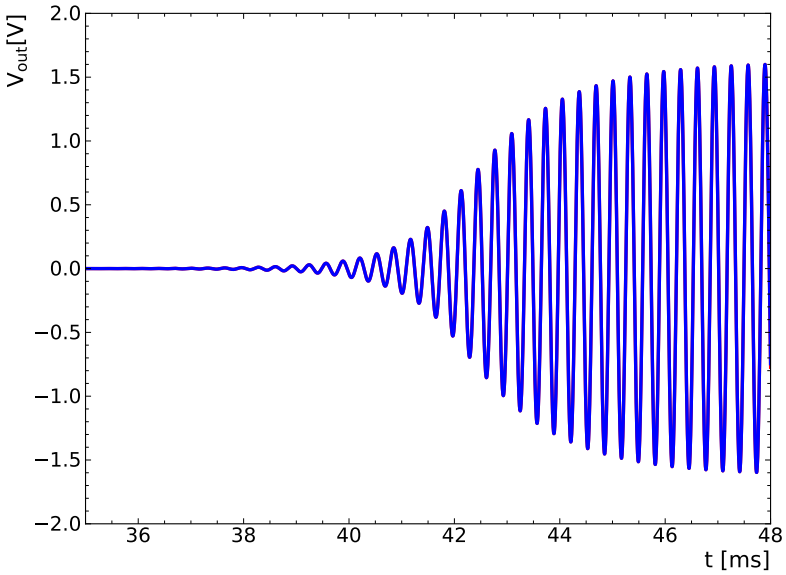
## TESTE 5 - RETIFICADOR DE MEIA ONDA (dc_source.net)




## TESTE 6 - RETIFICADOR DE ONDA COMPLETA (opamp_reticfier.net)



## TESTE 7 - OSCILADOR (oscilator.net)




## TESTE 8 - CURVA CARACTERÍSTICA DO MOSFET (mosfet_curve.net)