Laboratório de Instrumentação Elétrica – PSI3214 01/07/2020
Rodrigo Anjos de Souza, Elisabete Galeazzo.

Tutorial 1C:
Projeto de Simulador de Circuitos RC

Objetivos

1. Familiarização com desenvolvimento de um projeto em LabVIEW
2. Apresentação de macroestruturas de LabVIEW

Sumário

1 APRESENTAÇÃO DA ARQUITETURA "GENERAL PURPOSE VI"

Como descrito no documento "*Introdução ao Ambiente de Programação em LabVIEW*", todo o desenvolvimento na **linguagem de código G** se baseia em seu aspecto visual. Sendo assim, boa parte da leitura e criação de "VI's" é baseada na utilização de arquiteturas, isto é, **organizações de um ou mais VI's com finalidades determinadas**. Por exemplo, VI's com finalidade de interagir com o usuário, responsáveis pelo "*User Interface*", requerem estruturas determinadas para detecção da interação com o Painel Frontal. Apesar de não caber ao escopo deste tutorial o ensino de tais arquiteturas, será útil que a estrutura denominada "**General Purpose VI**", uma das principais arquiteturas utilizadas em projetos mais simples, seja desmembrada e explicada.

Essa organização, em geral, possui finalidade de cumprir uma tarefa ao longo de um período de tempo, tal como aquisição de dados ou processamento de dados. A partir dessa arquitetura surgem outras, mais avançadas e com propósitos mais específicos. O "**General Purpose VI**" é constituído de três partes: *Inicialização*, *Processamento* e *Finalização*, cada qual com suas especificidades.

A seção a seguir tem como finalidade apresentar o VI completo que iremos aprender a programar ao longo deste tutorial, ilustrando suas funcionalidades e a arquitetura selecionada.

1.1 Visão geral do executável a ser desenvolvido

Neste tutorial aprenderemos como realizar uma simulação de um circuito RC em LabVIEW. Veremos que apesar de simples, o processo de simulação de um circuito pode conter algumas armadilhas.

Como primeira tarefa, faça a extração dos arquivos contidos dentro do arquivo compactado [**Tutorial_1.zip**] em uma pasta adequada e abra o projeto do tutorial [**Tutorial 1.lvproj**]. Na janela aberta, denominada "*Project Explorer (Project: Tutorial 1.lvproj)*", vemos diversos arquivos de LabVIEW, os quais serão utilizados ao longo desse tutorial.

Inicialmente abra o arquivo [**tut1_main.vi**].

Repare que este arquivo contém toda a lógica de programação para realizar a tarefa almejada (neste caso, em particular, contém toda a simulação de circuitos de 1ª ordem). Para verificar o que o programa faz, execute-o por meio de um clique no botão de execução, situado no canto superior esquerdo da tela (vide Fig.1.1). Inicializada sua execução (indicada quando a "flecha" selecionada anteriormente fica preta), deslize os controles de R e C com o mouse. Observe que é criada uma curva RC, como exemplificado na Fig. 1.2, a seguir.

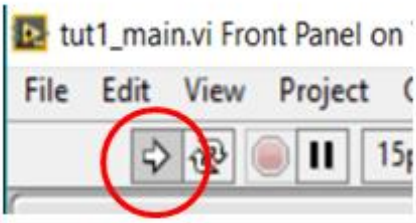

Fig. 1.1 – Botão de execução do programa.

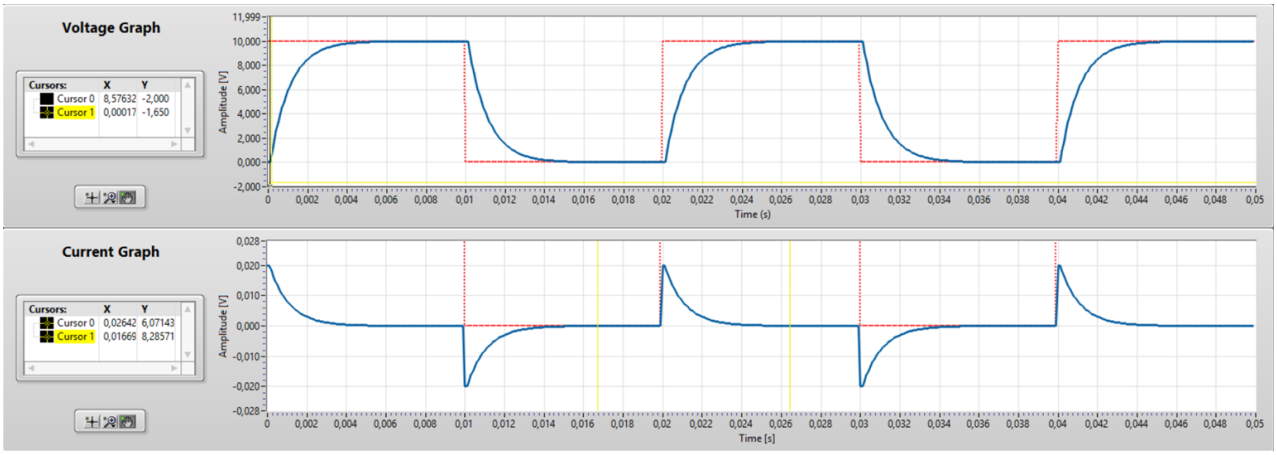

Fig. 1.2 – Curvas RC criadas na inicialização.

Verifique que podemos simular não só o comportamento do circuito RC neste "programa executável", como também podemos alterar os parâmetros da onda de entrada do circuito (sinal de excitação), que, neste caso exemplificado, é uma onda de formato retangular. Mude os controles "Signal Frequency" e "Duty Cycle" e veja o efeito no sinal de controle, como ilustrado na Fig. 1.3.

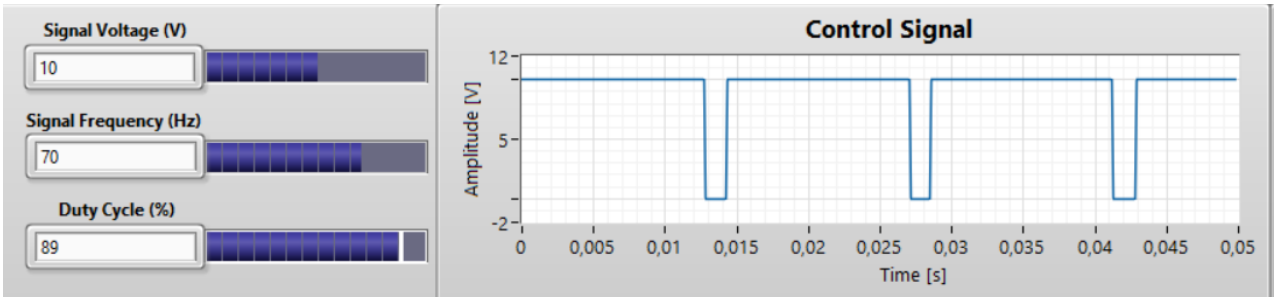

Fig. 1.3 – Sinal de controle (sinal na entrada do circuito RC).

Neste programa, a forma de onda do sinal de "controle" é ilustrada no gráfico denominado "Control Signal".

Feita a análise geral do comportamento deste "VI", vamos agora ver como foram criadas suas funcionalidades. Visualize o diagrama de blocos deste VI clicando "**ctrl-E**", ou, na barra de ferramentas, selecione Window → show Block Diagram. Nesta janela (Block Diagram) encontra-se toda a programação deste VI. Nas seções a seguir serão detalhadas suas principais partes.

1.2 Inicialização

Na etapa de Inicialização, como o nome implica, são inicializadas todas as estruturas necessárias para o funcionamento do VI. Nela ocorrem a inicialização de variáveis, dos vetores, abertura e configuração de portas. No programa em questão, nesta fase serão configurados os comportamentos de controles e indicadores, tais como os gráficos. Em geral, a etapa de inicialização de um programa que faz uso da arquitetura "*General Purpose VI*" é reservada para configuração de parâmetros para que o programa possa ser inicializado com sucesso.

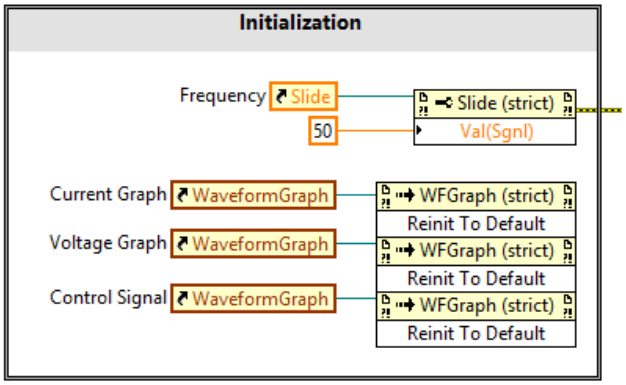

Fig. 1.4 – Inicialização do programa.

1.3 Processamento

Na etapa de processamento ocorre toda a manipulação de dados necessária para cumprir a função projetada: ou seja, desde o recolhimento de dados introduzidos em tempo de execução pelo usuário, até mesmo cálculos mais complexos como a determinação dos coeficientes da FFT (Fast Fourier Transform), por exemplo.

No programa em questão, **lembre-se de que o objetivo é de simular um circuito de primeira ordem**. Apesar da programação indicada na Fig. 1.5 ser um pouco avançada em um primeiro contato com a linguagem de programação em LabVIEW, o objetivo aqui não é a compreensão total do programa, mas sim das manipulações necessárias para que seja realizada uma simulação de um circuito de primeira ordem com sucesso. Todos os outros elementos presentes se prestam a enriquecer a experiência do aluno enquanto à utilização do programa.

Por exemplo, neste programa foi necessário elaborar uma lógica para identificar quando o circuito se encontra carregando ou descarregando, para então serem aplicadas as equações de tensão e corrente para simular o circuito (contidas no bloco em vermelho). Mas não se preocupe com isso agora. Vamos apenas utilizar este subVI.

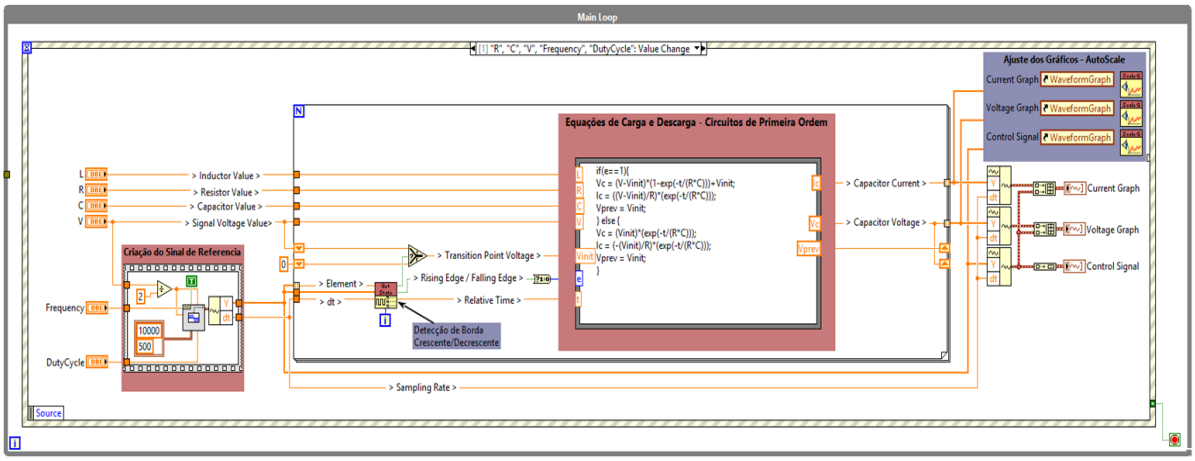

Fig. 1.5 – Etapa de processamento do simulador de circuitos RC.

Porém, é de suma importância identificar neste momento quais serão as partes que devem ser compreendidas e que serão programadas pelo aluno. Neste tutorial abordaremos apenas **como programar os elementos que se encontram dentro dos dois blocos vermelhos** **(indicados na Fig. 1.5)**. Se ainda não o fez, abra o **diagrama de blocos** do arquivo **tut1_main.vi** (apertando **Ctrl+E**), e veja com mais detalhes as seções destacadas em vermelho, que serão trabalhadas neste exercício.

1.4 Finalização

Na etapa de finalização, o programa não possui mais tarefas relacionadas à função projetada e precisa agora fechar todas as referências dinâmicas que foram criadas ao longo de sua execução. Deste modo, nessa etapa, configurações realizadas na inicialização são “resetadas”, portas de comunicação são fechadas e quaisquer erros encontrados durante a execução são tratados. No programa de simulação que será criado neste tutorial, a etapa de finalização não será utilizada, pois não existem portas de comunicação ou outras referências dinâmicas a serem fechadas.

2 GERAÇÃO E ANÁLISE DO SINAL DE REFERÊNCIA

Após ter uma visão geral do que se trata o programa a ser desenvolvido e qual a sua funcionalidade, vamos dar início à sua programação. Feche o programa [**tut1_main.vi**] e abra o VI [**pr1.vi**]. Você deve visualizar uma imagem equivalente à Fig. 2.1 quando abrir o Painel Frontal do VI.

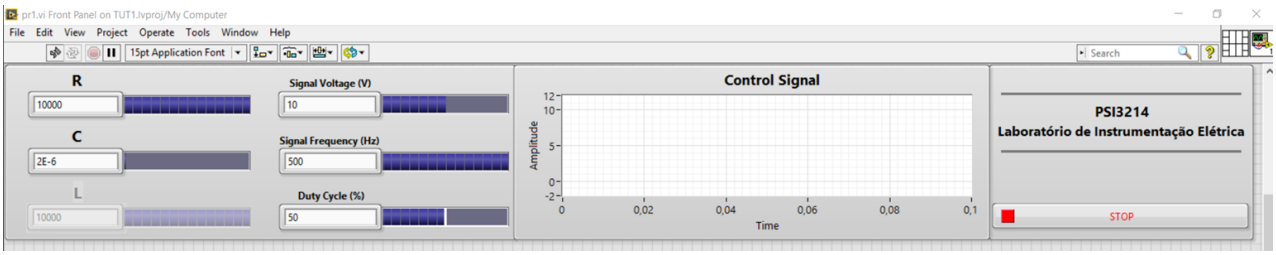

Fig. 2.1 – Painel Frontal do programa [**pr1.vi**].

Note que este programa, diferentemente do anterior, se encontra incompleto (a flecha no canto superior esquerdo apresenta-se "quebrada ou rompida").

Abra o diagrama de blocos do VI (por meio do atalho "**Ctrl+E**") e note que as áreas antes em vermelho (indicadas na Fig. 1.5) encontram-se vazias aqui. **Nosso objetivo é preencher estes blocos com suas funcionalidades ao longo deste tutorial.**

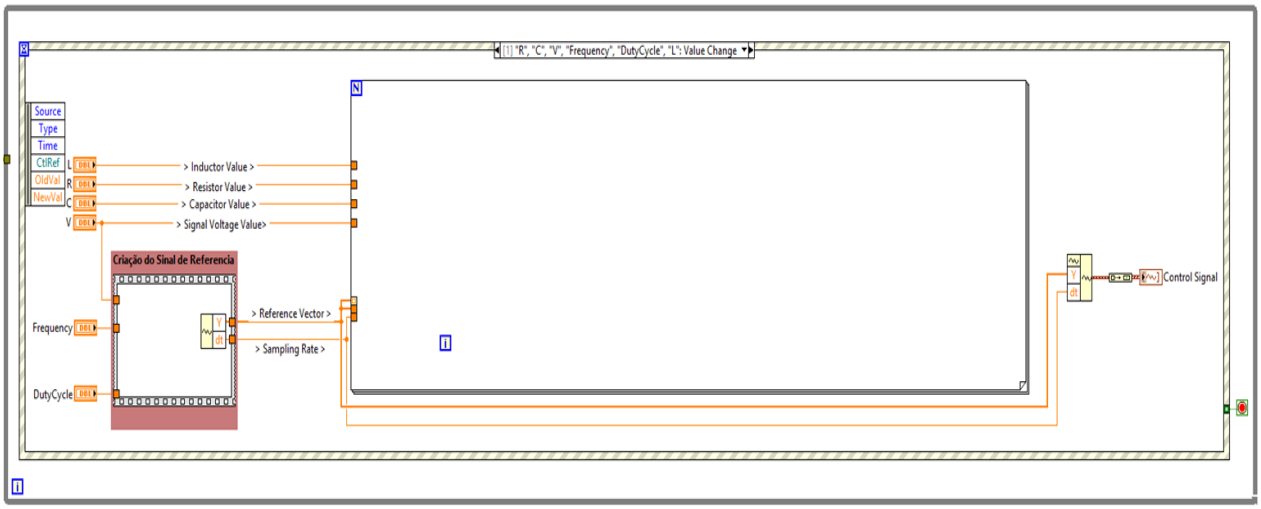

Fig. 2.2 – Diagrama de blocos do programa [**pr1.vi**].

Primeiramente, para que possamos obter uma resposta temporal de um circuito RC, devemos definir qual o sinal de entrada do sistema. Assim como efetuado na disciplina PSI3212, analisaremos a resposta RC a partir de uma entrada de onda quadrada como excitação, que é um

sinal comum para aplicações digitais. As ondas quadradas simulam a ação de uma chave fechar ou abrir num circuito de forma repetitiva, possibilitando a visualização do efeito de carga e descarga do capacitor de forma periódica.

No LabVIEW já existe um subVI pronto, disponibilizado na sua biblioteca, cuja função é gerar um sinal de ondas quadradas, e ele pode ser (e será) utilizado diretamente no VI que estamos elaborando.

Ao lado, Fig.2.3, encontra-se o nome e a ajuda contextual de tal bloco (denominado "**Square Wfm**"), que deve ser encontrado no menu do diagrama de blocos através de: "**View > Function Palette > Signal Processing > Wfm Generation**". Note que também podemos acessar a *Paleta de Funções* por meio de um clique com o botão direito do mouse em qualquer região em branco do Diagrama de Blocos.

Clique no bloco e arraste-o para dentro da área denominada "Geração do sinal de Referência". Vemos que, de todas as opções de sinais de entrada e saída disponíveis deste bloco, precisamos apenas nos preocupar com: **offset**, **amplitude**, **frequency**, **duty cycle (%)**, **sampling info** e a saída **signal out**.

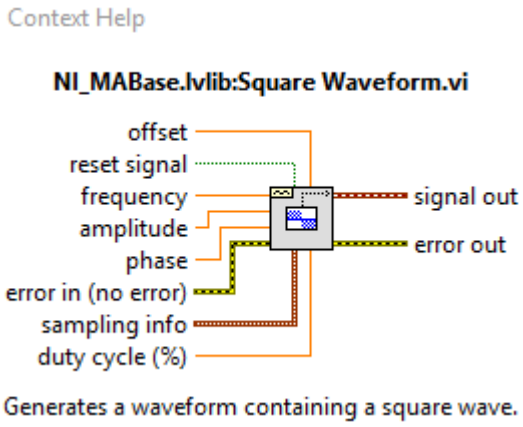

Fig. 2.3 – Ajuda contextual "*Square Waveform*" (Ctrl+H).

Devemos agora programar adequadamente este bloco vermelho para criar a onda quadrada de amplitude de [0, +V]. Para isso, deve-se aplicar um sinal V/2 nas entradas "**Offset**" e "**Amplitude**".

Siga os passos a seguir para construir o bloco "Criação do Sinal de Entrada", de acordo com as indicações das Fig. 2.4 e Fig. 2.5:

→ Crie um bloco "Divide" (selecionando-se o ícone apropriado na aba "Numeric" da paleta de

funções por meio de: **View > Function Palette > Numeric > Divide**);

→ Conecte, no terminal superior do bloco "Divide", o sinal V, e sua saída aos terminais "Offset" e "Amplitude" do bloco que gera a onda quadrada. Para efetuar a ligação entre os terminais, selecione o ícone "Connect Wire" na paleta de ferramentas: "**View > Tools Palette**" ou pressione "**Shift + Botão Direito**" em qualquer região em branco no Diagrama de Blocos.

→ No terminal inferior do bloco "Divide", crie uma constante (através do botão direito do mouse, selecione **create > constant** no menu aberto) e dê a ela o valor 2 (para isso clique na paleta de ferramentas o ícone "**edit text**").

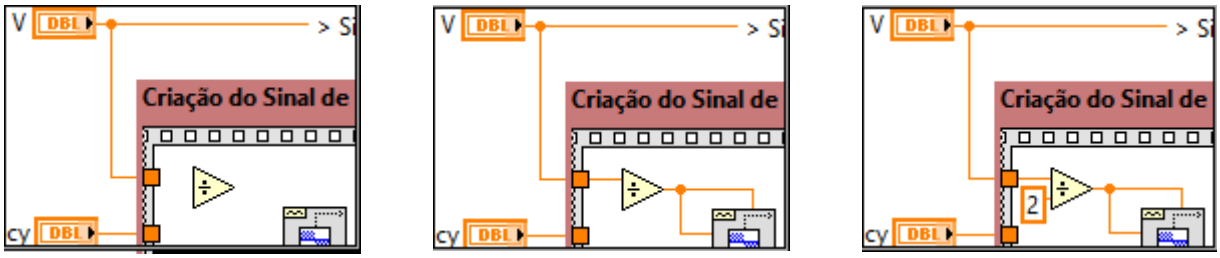

Fig. 2.4 – Criação das entradas "*Offset*" e "Amplitude".

As entradas "**Frequency**" e "**Duty Cycle**" deste bloco, que já estão ligadas diretamente aos controles correspondentes, devem ser agora conectadas ao bloco da onda quadrada, da seguinte forma:

→ Ligue os terminais "Frequency" e "Duty Cycle" na borda direita da área que estamos trabalhando aos terminais de nome correspondente no bloco que gera a onda quadrada.

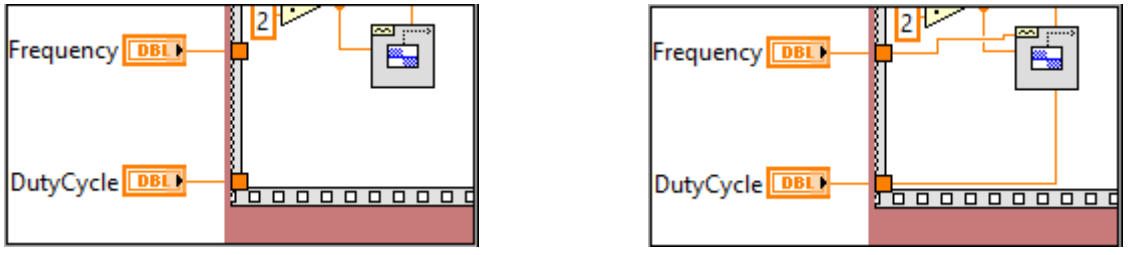

Fig. 2.5 – Conexão dos sinais "*Frequency*" e "*Duty Cycle*".

Você deve agora criar uma constante na entrada "**Sampling Info**", como indicado na Fig. 2.6, mantendo as configurações de taxa de amostragem constantes durante a execução do programa.

Para isso efetue:

→ Crie uma constante no terminal "Sampling Info" (utilize o botão direito do mouse, e no menu aberto selecione **create > constant**) e mude seus valores para os seguintes: Fs = 1E4, e #s = 500. Note que, para os valores inseridos, será gerado um sinal que inicia em 0 s e tem como valor final 500/1E4 = 0,05 s. Caso seja exigido um sinal com t_{FINAL} > 0,05 s, basta aumentar o número de amostras "**#s**".

→ Por fim crie uma constante no terminal "Reset Signal" e mude seu valor para "True".

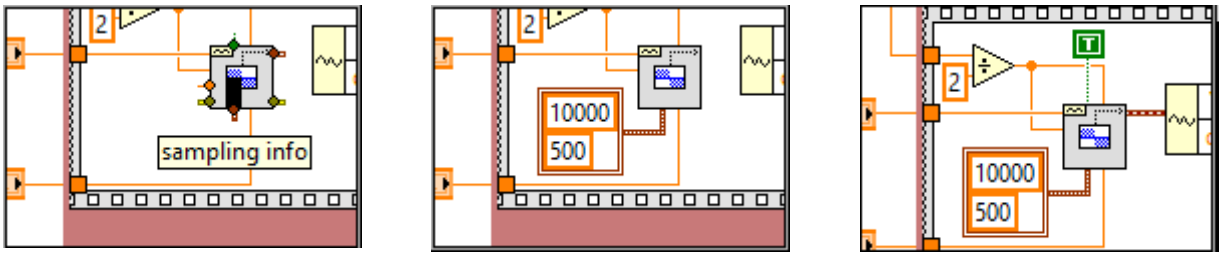

Fig. 2.6 – Criação da constante "*Sampling Info*".

Feita a geração do sinal, realizaremos algumas ligações da saída do bloco.

Como indicado na Figura 2.7, conecte o bloco que gera a onda quadrada ao terminal de entrada da função "Get Waveform Components". Repare que ao fazer essa ligação estamos separando o vetor de floats, que compõe o vetor de referência, do **dt**, que está relacionado com o período entre amostras. Feito isso, execute o programa e veja como a mudança dos controles afeta o sinal gerado.

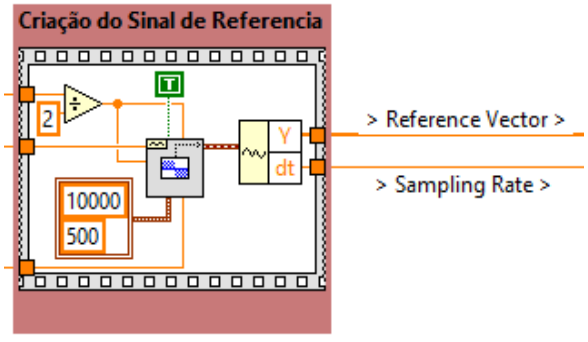

Fig. 2.7 – Montagem final.

Feitas todas as conexões corretamente, ao executar o programa uma onda quadrada deverá aparecer no gráfico “Control Signal”, assim como o da Fig. 2.8:

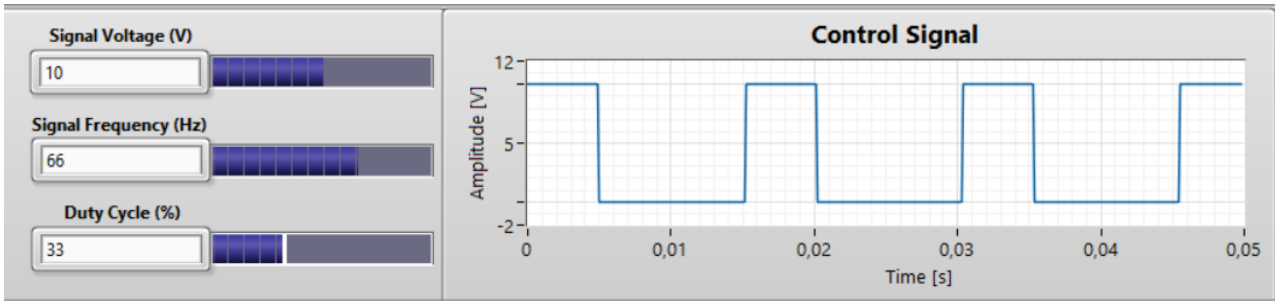

Fig. 2.8 – Resultado da programação realizada.

Mude os valores de “**V**”, “**Frequency**” e “**Duty Cycle**” e verifique se o programa criado se comporta como esperado.

3 APLICAÇÃO DAS EQUAÇÕES DE UM CIRCUITO RC

Abra agora o VI [**pr2.vi**].

Perceba que este programa contém as mesmas informações do programa **pr1.vi**, no entanto foram feitas pequenas modificações para acomodar melhor os gráficos de Tensão e Corrente, bem como alguns outros detalhes no diagrama de blocos, que serão trabalhados nesta sessão.

Visto isso, vamos à terceira parte do exercício, o processamento do sinal gerado para que obtenhamos uma resposta RC.

Analise as equações indicadas a seguir: elas correspondem às equações de carga e descarga de um circuito RC e deveriam ser familiares a você (como estudado em Circuitos I).

Veremos que apesar de tais expressões serem inteiramente adequadas no âmbito da disciplina de Circuitos I, existem algumas nuances em relação a seu uso ao serem aplicadas para realizar **uma simulação** de um circuito RC. Desta forma, iremos aos poucos modificá-las a fim de aperfeiçoá-las para tal finalidade.

[Equações de Carga]:

$$V_{CAP} = V_{CC} \cdot \left(1 - e^{\frac{-t}{R \cdot C}}\right);\ I_{CAP} = \frac{V_{CC}}{R} \cdot e^{\frac{-t}{R \cdot C}}$$

[Equações de Descarga]:

$$V_{CAP} = V_{CC} \cdot e^{\frac{-t}{R \cdot C}};\quad I_{CAP} = \frac{V_{CC}}{R} \cdot e^{\frac{-t}{R \cdot C}}$$

Cabe destacar também que, apesar destas equações serem expressões relativamente simples matematicamente, sua aplicação em LabVIEW seria desastrosa se realizada por meio de blocos básicos de soma, de multiplicação e de exponenciação. Deste modo, será apresentada uma ferramenta que nos ajudará a realizar contas complexas de modo simples e intuitivo dentro do LabVIEW: o bloco “Formula Node”.

3.1 Formula Node

Os “Formula Nodes”, assim como seu nome sugere, são blocos que permitem ao usuário definir entradas, saídas e seu comportamento por meio de fórmulas matemáticas e lógicas. Sua sintaxe é quase idêntica à de linguagem C, logo é uma ferramenta muito útil para aqueles que possuem algum conhecimento de C e procuram aprender LabVIEW.

Tomando a figura ao lado como referência, criaremos um bloco que possui seis entradas e duas saídas.

As entradas “**R”**, “**L”**, “**C”** possuem os valores de Resistores, Capacitores e Indutores que podemos utilizar em nosso circuito simulado. “**V”** corresponde à tensão do sinal de referência, “**t”** é o tempo simulado do circuito e, por fim, “**e”** é o estado do circuito, carga ou descarga. Já as saídas correspondem à corrente e tensão no capacitor do circuito RC.

Fig. 3.1 - “*Formula Node*”.

Para configurar o *Formula Node* efetue:

→ Crie na janela do **diagrama de blocos** um “Formula Node” (clicar em **View > Functions Palette > Structures > Formula Node**) e coloque-o em um lugar adequado, dentro da área denominada “RC Simulation”.

→ Crie as entradas e saídas necessárias para que sejam aplicadas as formulas desejadas. Clique com o botão direito do mouse na borda da estrutura e selecione “**Add Input**” para criar entradas e “**Add Output**” para as saídas. Crie as entradas “L”, “R”, “C”, “V”, “e” e “t” e as saídas “Ic” e “Vc”. Lembramos que em LabVIEW as entradas ficam à esquerda e as saídas à direita.

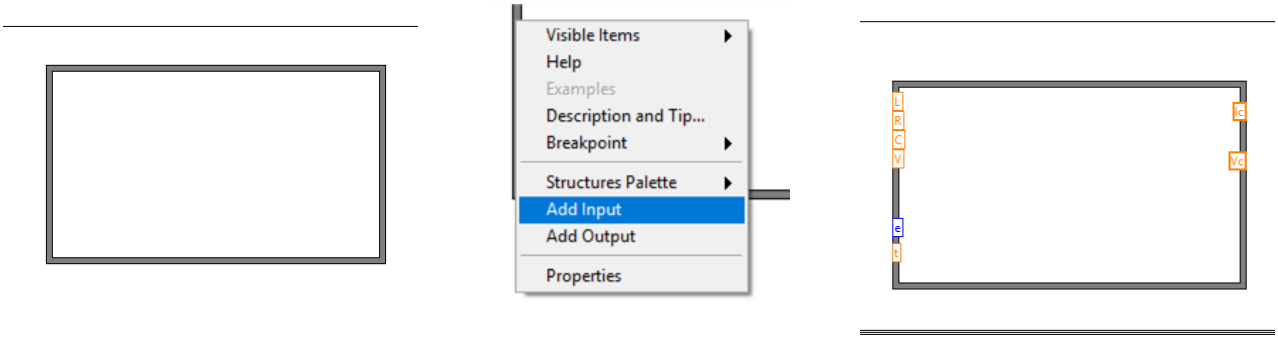

Fig. 3.2 – Criação de um "*Formula Node*".

Feito isso, basta conectar os terminais de mesmo nome para que possamos nos concentrar na aplicação das fórmulas a serem utilizadas.

3.2 Detecção de Borda / Formula Node com estruturas condicionais

Antes mesmo de iniciarmos a programação do bloco, encontramos nossa primeira barreira: como selecionar qual das equações a ser utilizada? Nelas pressupõe-se apenas uma carga, ou descarga do circuito, ou seja, não há uma equação única que descreve todo o comportamento do circuito.

A solução que optaremos será detectar quando o circuito muda de carga para descarga, ou o oposto, e com isso vai modificar as equações utilizadas. No tutorial 3, veremos que existem outros modos de se representar um circuito RC, de modo a que sua simulação será simplificada.

Como não cabe a esse tutorial apresentar a lógica necessária para que seja detectada a mudança de carga/descarga, foi criado um subVI responsável por isso, denominado "**GetState.vi**", Fig. 3.3.

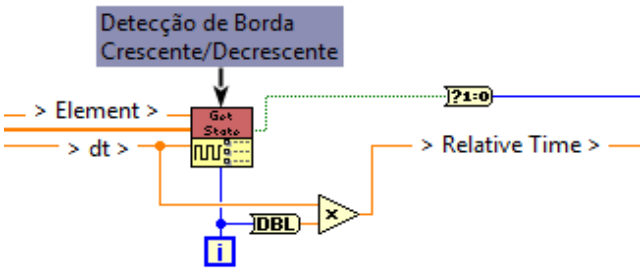

Fig. 3.3 – Detecção de Borda

Basicamente este bloco detecta se a amostra atual do vetor de referência é menor ou maior do que a amostra anterior. Caso seja maior ou menor, muda-se a saída "**e**" para "1" (Borda Crescente) ou "0" (Borda Decrescente). Caso as amostras sejam iguais, mantem-se a saída com o mesmo valor da última iteração.

Assim, para realizarmos uma simulação com ambos estados, basta adicionarmos ao *Formula Node* uma **estrutura condicional**, como indicado na Fig. 3.4. Utilizaremos o valor de "**e**" dentro de uma estrutura "if/else", análoga às estruturas condicionais vistas nas disciplinas de "Introdução à Computação", para selecionarmos qual equação a ser utilizada. Deste modo, escreva o seguinte programa dentro do *Formula Node* criado:

```
if(e==1){
      Vc = V*(1-exp(-t/(R*C)));
      Ic = (V/R)*(exp(-t/(R*C)));
} else {
      Vc = V*(exp(-t/(R*C)));
      Ic = -(V/R)*(exp(-t/(R*C)));
}
```

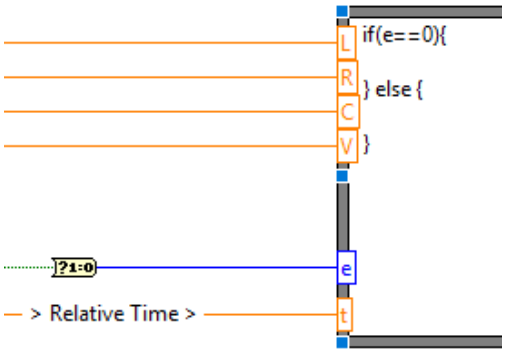

Fig. 3.4 – *Formula Node* com estrutura condicional.

Feita a programação do *Formula Node*, basta inserir os valores abaixo e executar o programa. Espera-se que seja obtido um gráfico como o da Fig. 3.5.

R = [500Ω], C = [2μF], V = [10 V], Duty Cycle = [50%], Frequency = [50 Hz]

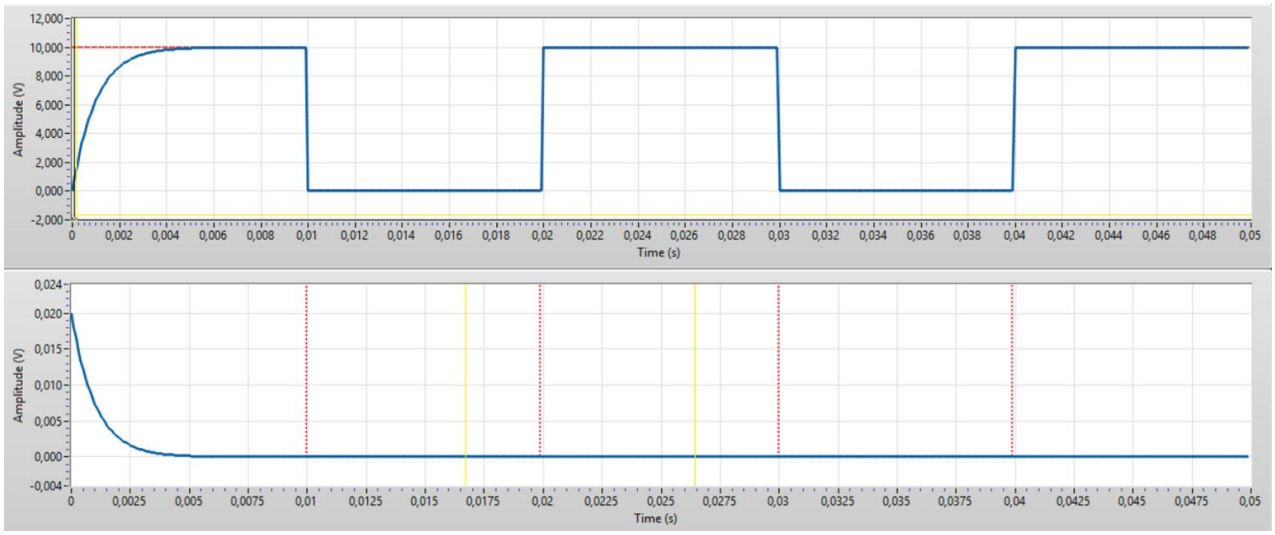

Fig. 3.5 – Resultados do *Formula Node* com estrutura condicional.

Apesar das fórmulas de carga/descarga terem sido implementadas corretamente, o gráfico acima não mostra um comportamento esperado de um circuito RC. Ao fazermos uma análise dos gráficos obtidos, notamos que enquanto a corrente tende a zero, a tensão segue a referência. Tal comportamento é consequência da variável "t" das equações, outra das limitações das equações canônicas.

Como visto em Circuitos I, a constante de tempo, quando inserida no contexto de um circuito de primeira ordem, corresponde ao amortecimento do sistema. Perceba, no entanto, que quando $t \rightarrow \infty$, a saída $V_C \rightarrow V_{IN}$. Deste modo, o comportamento da Figura 3.5 pode ser atribuído a esse fato.

Para que o comportamento RC seja corrigido, a variável que corresponde ao tempo, "t", deve ser zerada toda vez que o circuito muda de estado. Isso pode ser alcançado por meio de algumas estruturas condicionais e alguns cálculos, contudo, novamente isso excede o escopo do tutorial. Assim, para o mesmo efeito, basta realizar a **reconexão do fio "t"** como indicado na Figura 3.6(b). Com isso, ao mesmo tempo em que o subVI fornecido aponta uma borda crescente ou decrescente, ele zera a variável "t" utilizada nas fórmulas.

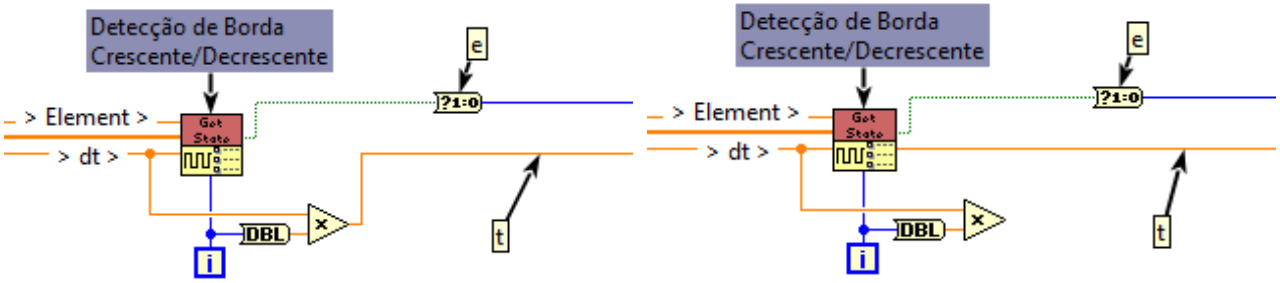

Fig. 3.6 – Reconexão da variável "t": (a) ligação que deverá ser eliminada (selecione o fio e delete); (b) ligação correta do fio para o desejado.

Execute o programa mais uma vez, utilizando os mesmos valores anteriores para simular o circuito, agora com a variável “t” sendo zerada a cada vez que se muda de estado. O resultado deve ser equivalente ao ilustrado na Fig. 3.7.

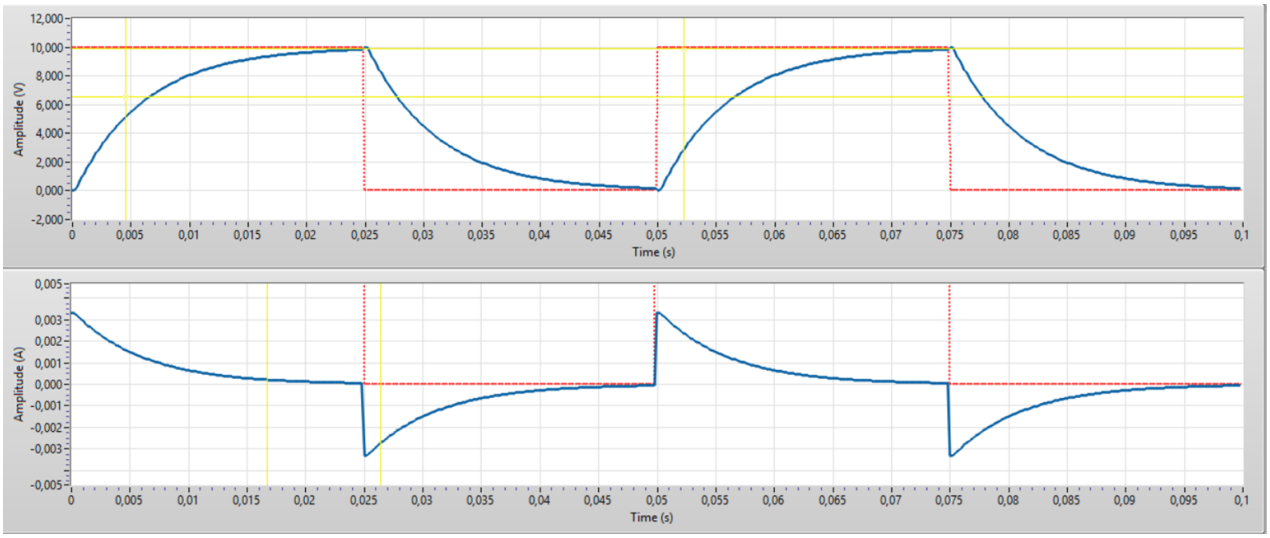

Fig. 3.7 – Resultado esperado do uso de Formula Node com estruturas condicionais.

Perceba que agora o comportamento do circuito parece muito mais com o esperado!

3.3 Criação de Paleta de Gráficos

Após serem criados os *Formula Nodes*, será necessário agora realizar medições e análises sobre as simulações realizadas. Certifique-se de que não haja erros de compilação ou execução em seu programa. Se não houver erros de programação, quando executar o programa com as mesmas configurações dos itens anteriores, obteremos curvas equivalentes à Fig. 3.8:

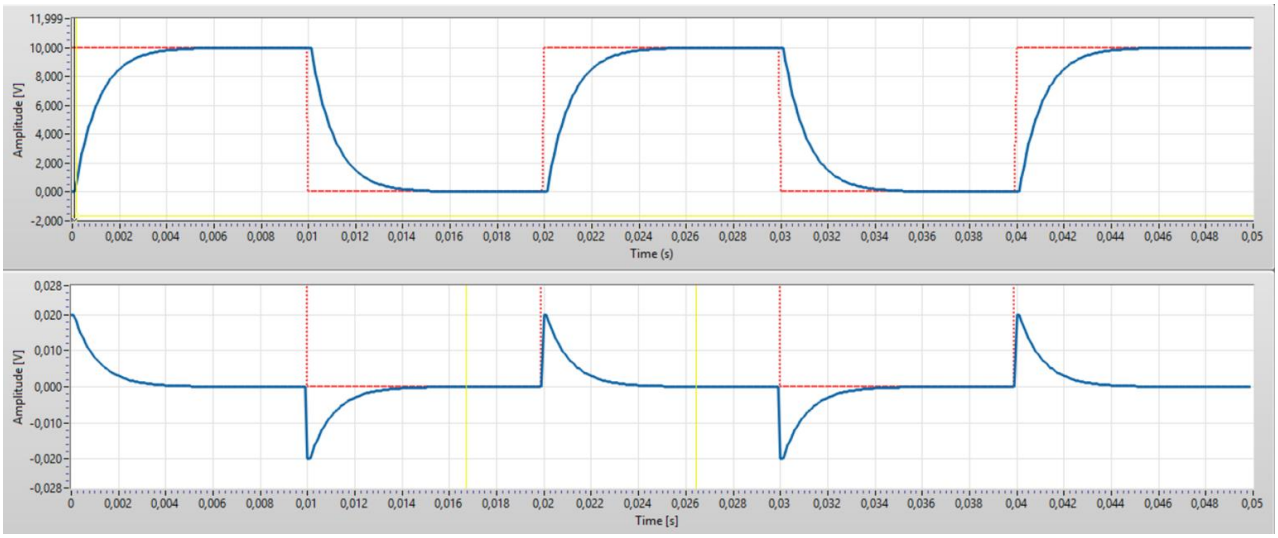

Fig. 3.8 – Sinal de referência para prosseguimento do tutorial.

Toda vez que realizamos esse tipo de simulação, nos interessamos por parâmetros temporais como “Tempo de subida”, “Amplitude de Sinal”, “Taxa de Amostragem”, entre outros.

Apesar de sua extração ser possível via programação, vamos começar a efetuá-las por meios mais simples, que ajudarão na criação de uma intuição do problema.

Para isso, utilizaremos a inspeção visual, por meio de **Cursores** e da **Paleta de Gráfico**. Para criá-los, basta clicar com o botão direito no gráfico desejado e na aba “**Visible Items**” selecionar “**Graph Palette**” e “**Cursor Legend**”, como exemplificado na Fig. 3.9.

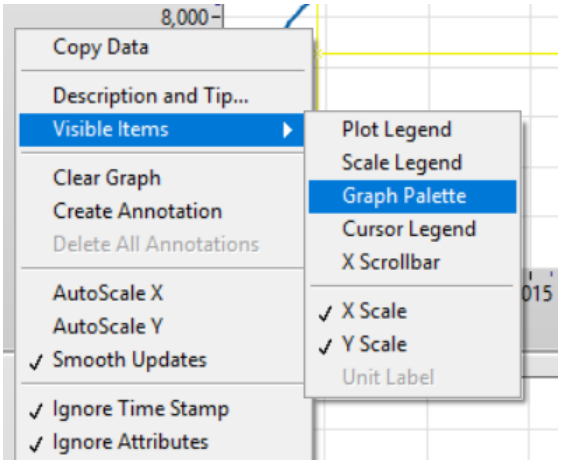

Fig. 3.9 – Paleta de Gráfico.

No conjunto de ícones da paleta de gráfico, podemos atuar sobre o gráfico em questão sem modificar seu conteúdo. Funções como **movimentação de cursores**, **ferramentas de zoom**, **movimentação do gráfico** estão presentes e permitem ao usuário focar em uma sessão específica desejada. Em futuros tutoriais essa funcionalidade ficará mais clara.

Fig. 3.10 – Ferramentas da Paleta do Gráfico.

A legenda de cursores (Fig. 3.11), por sua vez, permite a criação de cursores e visualização de seus valores. Por meio dela, realizaremos as medições pertinentes.

Para criar um cursor basta clicar com o botão direito na paleta e selecionar "**Create Cursor>Multi-Plot**". Crie alguns cursores no gráfico de Tensão e coloque-os em alguns pontos do gráfico. Perceba que cada cursor tem valores de X, Y e um nome.

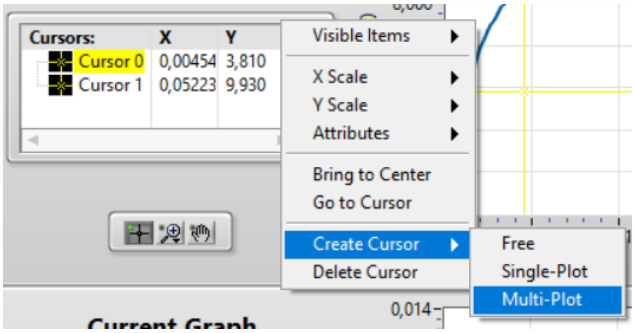

Fig. 3.11 – Legenda de Cursores.

Aplicação das ferramentas apresentadas:

Vamos agora aplicar esses conhecimentos e analisar algumas características do sinal. Sem fechar seu programa, abra o arquivo [**ex1.vi**] e veja que é praticamente idêntico ao VI que foi construído até agora. Contudo, diferentemente de um programa aberto em LabVIEW, este é um executável, o qual não temos acesso ao diagrama de blocos durante sua execução.

Note que neste programa não temos acesso aos valores V, R, C, L, Duty Cycle e Frequency. Pelo contrário, estes valores agora são entradas escritas, nas quais você deverá escrever suas respostas.

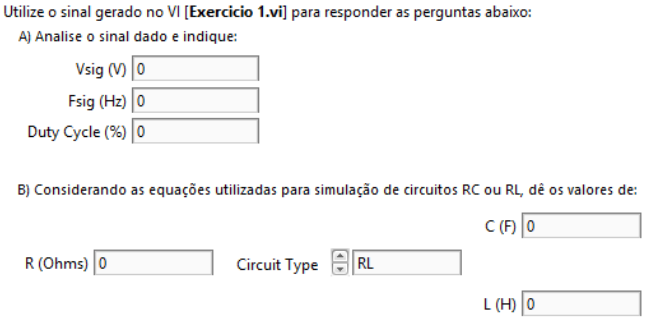

Fig. 3.12 – Tabela de respostas do exercício.

O intuito geral deste exercício é avaliar como podemos medir circuitos RC ou RL por meio de suas respostas quando submetidos a uma excitação com onda quadrada. Utilizando-se de todas as ferramentas expostas até agora, analise a resposta do circuito em questão e dê os valores pedidos (margem de 5%).

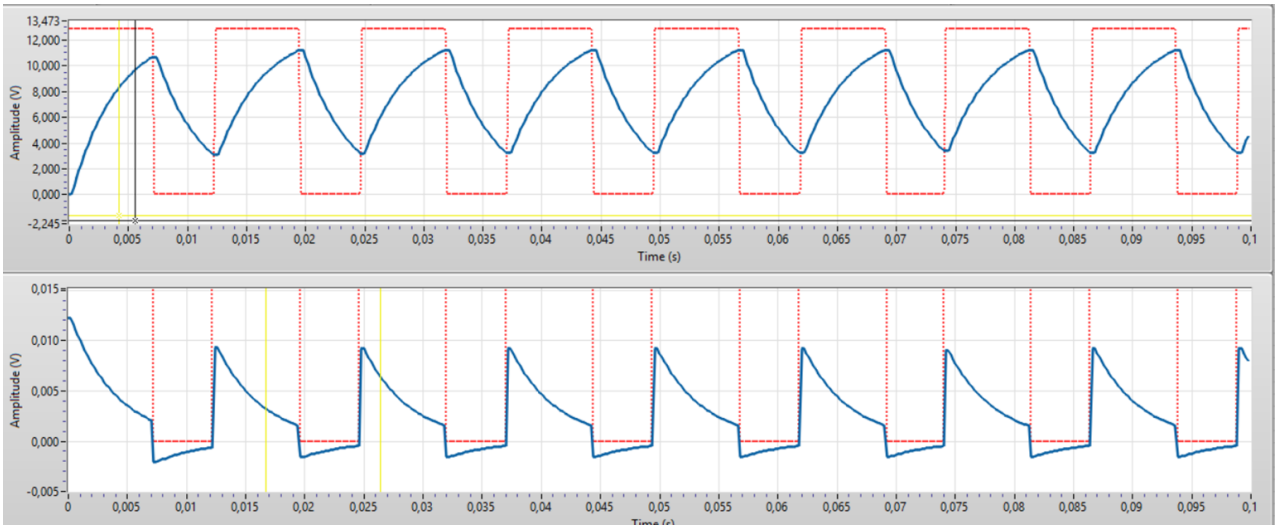

Fig. 3.13 – Exemplo de um exercício.

3.4 Análise do programa

Feito o exercício acima, veremos algumas limitações relacionadas ao modelo que temos de um circuito RC ou RL. Vamos agora verificar o comportamento do circuito para frequências maiores, e as limitações que surgem da atual simulação quando o ciclo de carga ou descarga não é completado. Voltando ao programa de LabVIEW que construímos (**pr2.vi**), insira os seguintes valores nos campos e execute o programa:

R = [1kΩ], C = [2μF], V = [10 V], Duty Cycle = [50%], Frequency = [100 Hz]

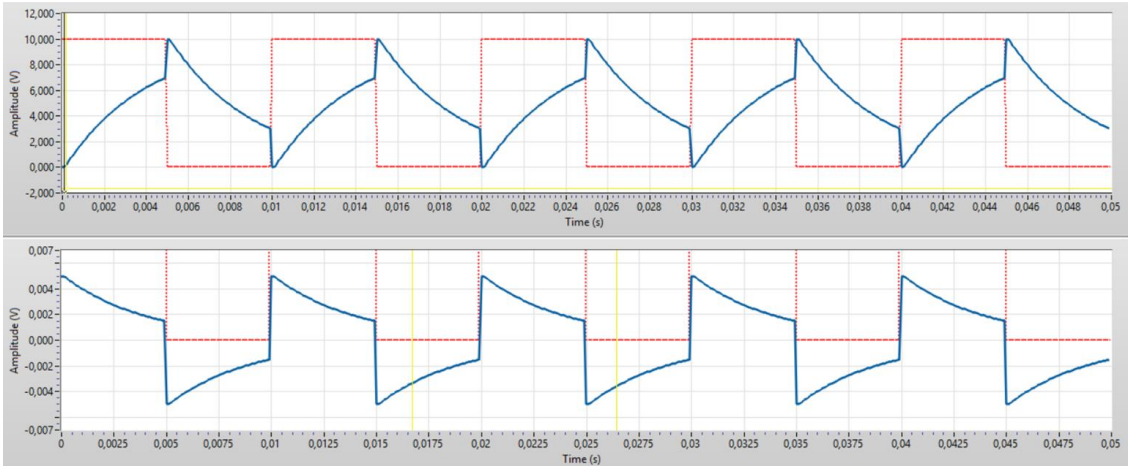

Fig. 3.14 – Limitações para frequências maiores.

Os gráficos da Fig. 3.14 são os resultados da simulação RC atual. **Perceba que tal resposta é equivocada**, pois quando o circuito realizar a transição de carga para descarga, ou de descarga para carga, observa-se um salto abrupto nas curvas, que foge completamente do comportamento esperado.

Isso ocorre por conta do equacionamento dado para o problema. Ou seja, quando o circuito passa de um estado para outro antes de carregar completamente (ou antes de descarregar completamente), o valor inicial para a equação exponencial muda, e passa a ser o valor indicado por V ao invés de ser utilizado o último valor de tensão no circuito.

Deste modo, para corrigir este problema (que ocorrerá toda vez que **não houver** tempo suficiente para carregar ou descarregar completamente o circuito) utilizaremos uma tensão V_{INIT}, que vem a ser a última tensão medida do circuito antes da troca de estados, ao invés de usarmos a tensão de entrada V_{CC} para a base da equação exponencial. Com tal correção, a equação de tensão do circuito RC, na **descarga** tem como primeiro ponto V_{INIT}, caindo exponencialmente a zero, como ilustrado ao lado.

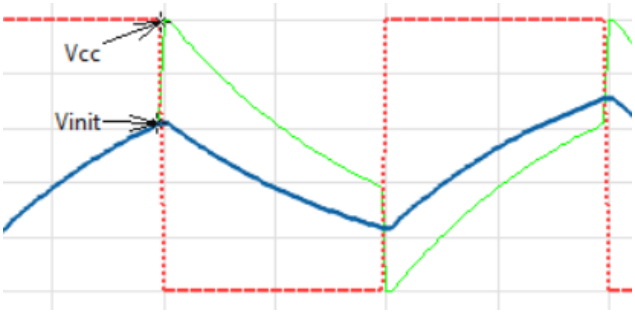

Fig. 3.15 – Compreensão do fenômeno de carga/descarga incompleta.

Assim, sua equação corrigida é dada por:

$$V_{CAP} = V_{INIT} \cdot e^{\frac{-t}{R \cdot C}}$$

Por sua vez, a equação de carga começa em V_{INIT} e termina em V_{CC}, de modo a que seja dada pela expressão ao lado.

$$V_{CAP} = (V_{CC} - V_{INIT}) \cdot (1 - e^{\frac{-t}{R \cdot C}}) + V_{INIT}$$

Já as equações de corrente, seguem o mesmo equacionamento dado inicialmente. Contudo, como a tensão inicial é diferente, temos as equações ao lado adaptadas.

Equação de Carga:

$$I_{CAP} = (\frac{V_{CC} - V_{INIT}}{R}) \cdot (e^{\frac{-t}{R \cdot C}})$$

Equação de Descarga:

$$I_{CAP} = -(\frac{V_{INIT}}{R}) \cdot (e^{\frac{-t}{R \cdot C}})$$

4 UTILIZAÇÃO DE "SHIFT REGISTERS" PARA SIMULAÇÕES COM MEMÓRIA

Como descrito na sessão 3.4, necessitaremos do último valor de tensão do circuito antes da mudança de estado. Para tal, é necessária a utilização de "**Shift Registers**", que são registros especiais que guardam o valor da saída de um laço e a disponibilizam no início do mesmo laço na próxima iteração. Tais registros permitem que o laço possua comportamento análogo de uma memória da saída de sua última iteração. Justamente é esse o recurso que precisamos para o cálculo de V_{INIT}. Você pode ler um pouco a mais sobre estes registros no Tutorial 1A introdutório à programação do LabVIEW disponibilizado nesta disciplina.

4.1 "Shift Registers"

Abra o programa [**pr3.vi**] e analise o diagrama de blocos. Novamente, temos algumas pequenas modificações em relação ao VI construído anteriormente. Com a adição de "Shift Registers" o programa agora é capaz de simular com certa precisão o comportamento de um circuito RC.

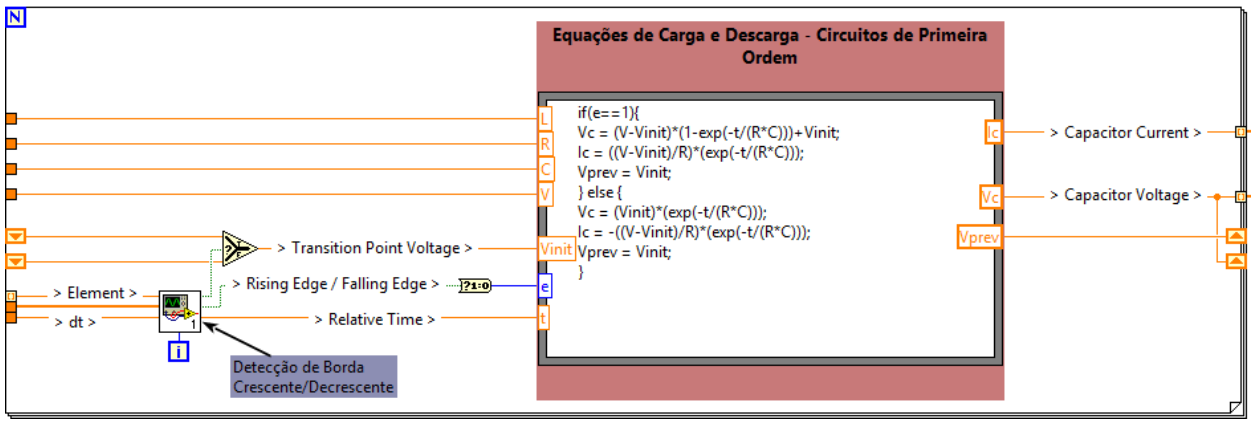

Fig. 4.1 – Programação Final.

Insira os valores abaixo nos campos e execute o programa. Perceba que agora mesmo que o circuito mude de estado durante carga ou descarga, temos um comportamento condizente com o que foi visto em laboratório!

R = [1kΩ], C = [2µF], V = [10V], Duty Cycle = [50%], Frequency = [500Hz]

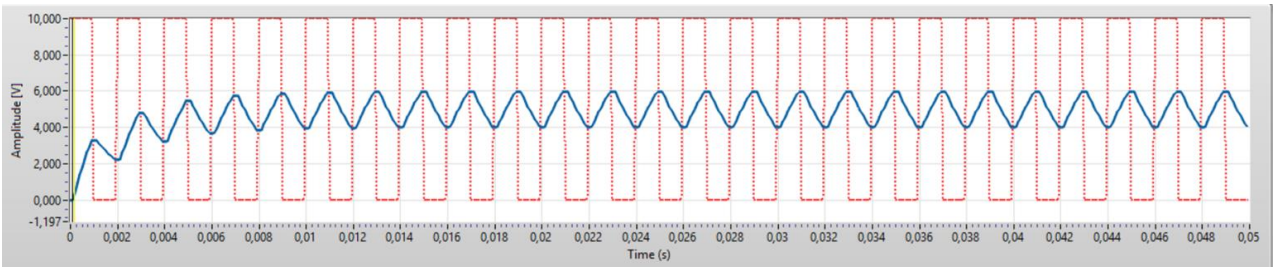

Fig. 4.2 – Resultado Final da Simulação.

5 CONCLUSÃO DO PROGRAMA

Ao final deste programa espera-se que o leitor tenha se familiarizado com o funcionamento do ambiente LabVIEW e algumas de suas ferramentas, de modo que seu desenvolvimento seja encarado de forma mais intuitiva. Apesar do tutorial ter sido concluído, seguem algumas sugestões de exercícios complementares para aqueles que se interessam em aprofundar seus conhecimentos.

5.1 Exercícios Complementares Sugeridos

a. Utilização do Circuito RC para obtenção de Ondas Triangulares:

→ Qual seria o procedimento para que, por meio de um circuito RC, possamos produzir uma onda triangular a partir de um sinal de entrada de onda quadrada?

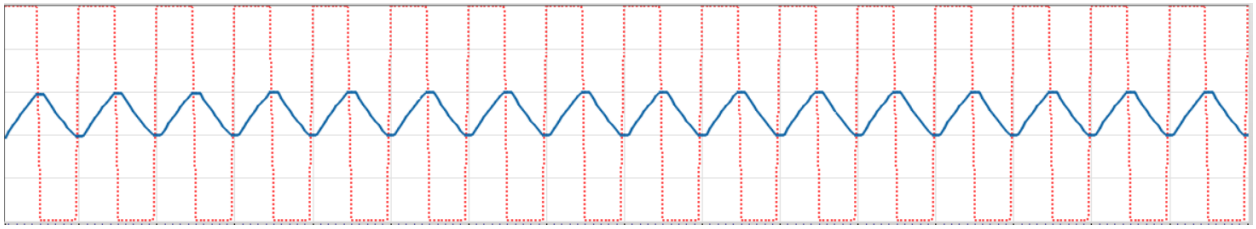

Fig. 5.1 – Onda Triangular

→ Porque a simulação do circuito que produz tal onda possui um formato exponencial no início?

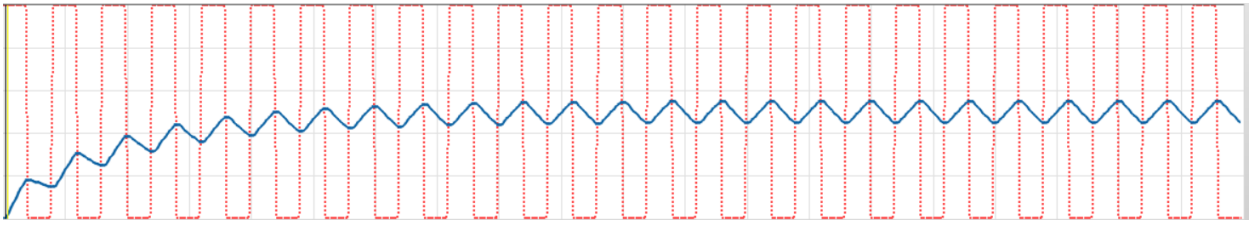

Fig. 5.2 – Início da formação de uma onda triangular.

b. Utilização do Circuito RC como filtro para sinais de PWM

Em muitas aplicações da indústria são utilizadas fontes chaveadas para fornecer energia para os equipamentos. Fontes para computadores, por exemplo, são fontes chaveadas. O termo que define esse tipo de fonte advém do fato que em seu interior são produzidas ondas quadradas com uma amplitude fixa e Duty Cycle variável, os quais passam por um filtro RC para produção de um valor DC. Por meio da variação do Duty Cycle varia-se o nível DC da saída. Suponha uma fonte chaveada que tenha:

V = [10 V], , Frequency = [1000 Hz]

Projete um filtro RC para que o tempo de acomodação do nível DC seja de 0,03 s para um Duty Cycle de 50%.

→Verifique que ao mudar o Duty Cycle, o valor DC aproximado no final muda proporcionalmente.