

Laboratório 4

ESTRUTURAS NÃO TEMPORIZADAS

4.1 Objetivos

Estudar e investigar os recursos de programação de modos estruturados em um VI ou em uma estrutura de controle tais como, *Loop FOR*, *Loop WHILE*, *Loop SEQUENCE* e modos estruturados de decisão *IF* e *CASE*.

4.2 Generalidades

Como em todo programa mais complexo ou em estratégia de supervisão e controle se faz necessário a geração de um programa estruturado com tomada de decisões, sequenciamento de rotinas ou tarefas e além de tudo de forma organizada, numerável e supervisionada.

Em linguagens convencionais estas características são proporcionadas com os recursos de comando para gerenciamento de *loops* ou sequências ordenadas. Iteração de comandos é uma das mais importantes estruturas que podem ou não ser contáveis, ou seja, que permitem enumerar e indexar resultados de cada iteração em um dado programa.

Os recursos mais comuns são disponibilizados no LabVIEW, porém como todos os outros recursos, em forma gráfica. Estes recursos são acessíveis no Pop-Menu *FUNCTIONS* da janela *DIAGRAMA* no sub-menu *Programming - Structures*. Esta forma de acesso é ilustrada na figura 4.1.

No menu de opções da janela *Structures* dispõe-se dos recursos principais : *While-Loop*, *Flat-Sequence* e *Case-Structure*, *Stacked-Sequence*, *For-Loop* e *Formula-Node*, *Feedback-Node*, *Timed-Loop* e *Event-Structures*, etc. O objetivo principal desta aula

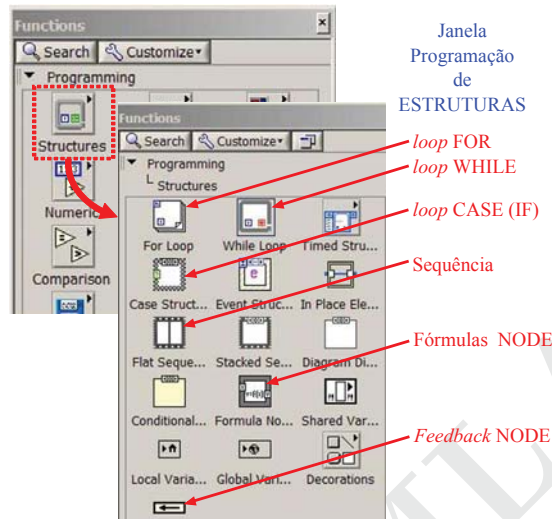


Figura 4.1: Menu e sub-menu de Estruturas

é estudar o uso de alguns destes recursos, os quais serão brevemente discutidos a seguir e posteriormente exercitados em exemplos.

4.3 Estrutura SEQUENCE

A estrutura SEQUENCE como o próprio nome diz e também como o desenho gráfico desta, é apresentado na janela DIAGRAMA, representa uma forma de se estabelecer a sequência desejada de execução de tarefas dentro de um VI. O desenho desta estrutura é o mesmo de uma película de filme ou como um *frame* de filme. A apresentação pode ser tipo *Flat* ou *Stacked*, ou seja, com os *frames* sequenciais linearmente ou com os *frames* sobrepostos respectivamente. Os *frames* são todos executados na sequência definida. No caso de *Flat-Sequence*, a disposições do desenho já determina a direção e disposição de passagem de valores entre os *frames*.

4.4 Estrutura CASE (Lógico e Múltiplo)

A estrutura CASE é por *default* uma estrutura tipo IF em que manipula um estado ou variável externa booleana *True/False*. Esta entrada deve ser providenciada externamente à estrutura. Neste caso as condições *True* ou *False* dão acesso a dois *frames* ou janelas onde são realizadas as operações do caso *True* ou do caso *False*.

Esta mesma estrutura funciona também como uma função CASE de condições múltiplas.

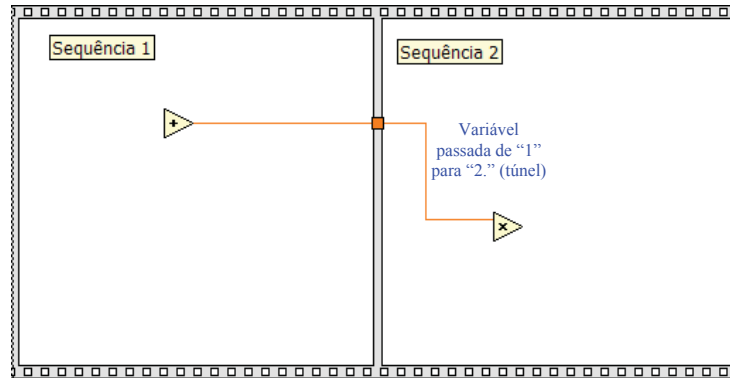
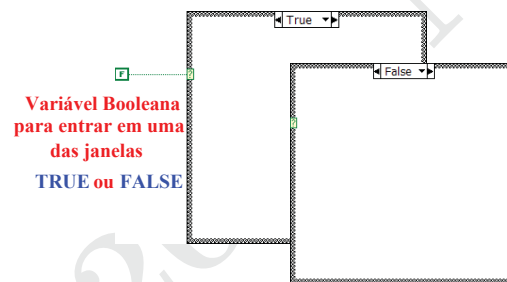
Figura 4.2: Exemplo de estrutura *Sequence*

Figura 4.3: Exemplo de CASE Lógico

Neste caso, em vez de *True/False* a estrutura é ordenada numericamente. Novamente, cada caso numérico representa um *frame* ou janela onde são editados os comandos relativos a cada caso a ser manipulado. Diferente da estrutura tipo *frame* de sequência, apenas uma janela é executada de acordo com a variável passada para esta estrutura.

Para converter uma estrutura CASE booleana em estruturas CASE múltipla basta conectar uma variável numérica ao símbolo de comparação logo após a inserção da estrutura no diagrama. Desta forma a estrutura booleana se torna numérica sendo "0" o caso "default". Para acrescentar novos "casos" . basta acionar o Pop-Menu na borda da estrutura e escolher a opção desejada. Esta mesma estrutura pode operar com entradas tipo *string*. Para isto basta seguir o mesmo procedimento do caso numérico.

Para se obter o resultado da manipulação do CASE em uma variável, basta conectar o resultado na borda do *frame*. Se todos os casos vão manipular uma única variável,

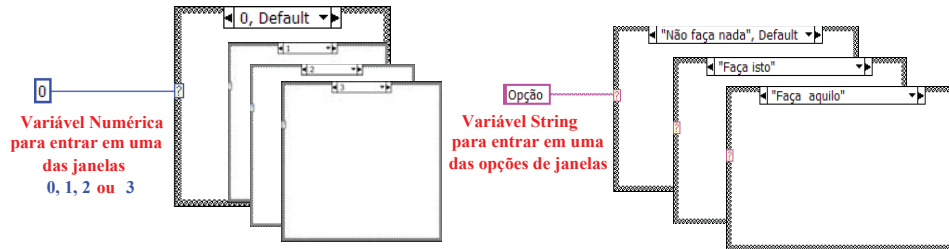


Figura 4.4: Exemplo de CASE Múltiplo

então em cada frame o resultado dever ser conectado neste mesmo ponto.

4.5 Estrutura FOR

A estrutura FOR é uma janela em que se executam “N” vezes os comandos dentro dela, sendo “N” inicializada externamente. “N” é necessariamente uma constante inteira. Desta forma a estrutura FOR realiza uma sequência finita e contável de operações. Para que uma variável seja disponível de forma indexada fora do *Loop* é necessário habilitar a indexação ao se executar a fliação das operações para fora (borda) do *Loop*. A forma normal é não-indexada. Pode-se ainda disponibilizar variáveis recursivas para serem utilizadas por exemplo em uma equação recursiva. Para isto, basta clicar na borda do FOR e adicionar *Shift-Register*. Isto representa que a cada iteração “k” no FOR, a variável conectada entrando no *Shift-Register* estará disponível como uma variável da iteração “ $k - 1$ ” na próxima iteração. Uma opção também é usar o *Feedback-Node*. Dentro da estrutura do FOR encontra-se ainda o ícone “i”, o qual contém o valor da iteração atual em execução e pode ser usado na tarefa em execução. Esta variável interna é sempre inicializada com 0 (zero) a cada re-início da estrutura FOR.

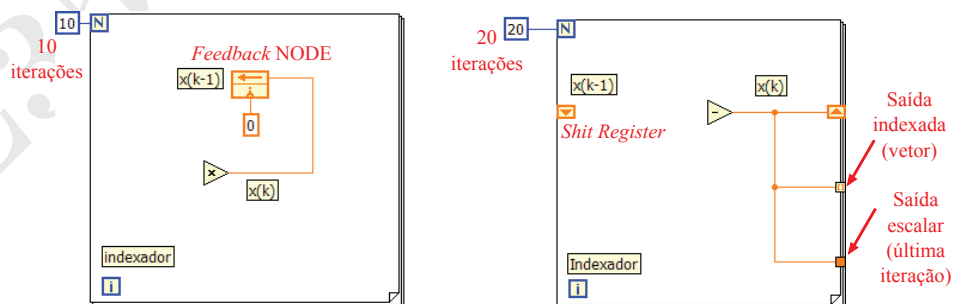


Figura 4.5: Exemplo de Loop Contável tipo FOR

4.6 Estrutura WHILE

A estrutura WHILE é semelhante àquela do FOR, porém o *Loop* é executado indefinidamente até uma dada condição lógica dentro do *Loop* seja avaliada e um resultado *False* determine a interrupção do *Loop*. Esta condição *False* deve então ser conectada ao ícone de interrupção da estrutura, tal como visto na figura a 4.6. Também neste caso é possível a inserção de variáveis indexadas para fora do *Loop* e variáveis recursivas para o próprio *Loop*. Na estrutura do WHILE dispõe-se também do contador de iteração cujo valor de cada iteração é disponível no ícone “i”. Se necessário os recursos de *Shift-Register* ou *Feedback-Node* podem ser usados dentro do WHILE da mesma forma como no caso do FOR.

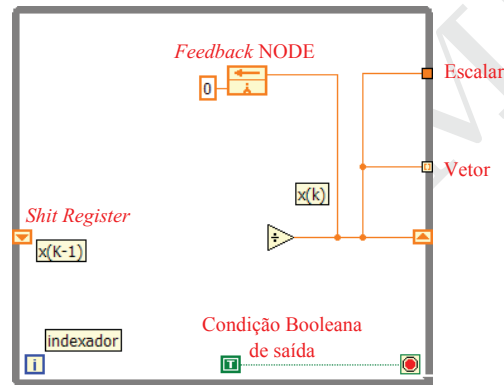
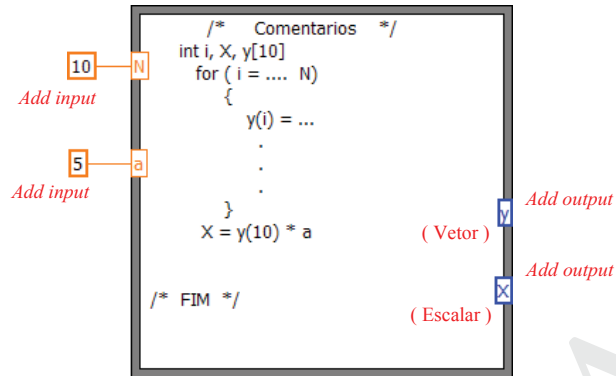


Figura 4.6: Exemplo de *Loop* não contável tipo WHILE

4.7 Estrutura FORMULA - NODE

A próxima opção de estruturas de programação é chamada FORMULA-NODE. Este recurso é o equivalente de se chamar uma sub-rotina externa. No caso insere-se uma sequência de comandos em linguagem equivalente a sintaxe C ou C++ em função das variáveis de entrada, internas e de saída. As variáveis de entrada e de saída devem obrigatoriamente ser nomeadas e são manipuladas dentro da estrutura pelo respectivo nome. Dentro da estrutura *Formula-Node* só podem aparecer expressões matemáticas.

Na sequência serão sugeridas algumas atividades como treinamento de uso dos recursos apresentados acima.

Figura 4.7: Exemplo de estrutura *Formula-Node*

4.8 Atividade de aplicação

Procure executar os VIs com as funções solicitadas a seguir sem utilizar recursos de temporização que serão vistos na próxima aula. Procure treinar também a geração e gerenciamento do VI por meio de um painel equivalente dotado entradas, saídas e mensagens “*Strings*”.

i) - execute um VI usando um *Loop FOR* para realizar a contagem de iterações e que indique a soma parcial das iterações s_p e a soma final s_T das N -iteraões.

$$s_f = \sum_{i=1}^N i \quad s_P = [s_{p1} \ s_{p2} \ s_{p3} \ \dots \ s_{pN}]$$

ii) - Execute o mesmo VI acima, mas usando um *Loop WHILE* para realizar o mesmo somatório do caso anterior. Acrescente neste VI um tipo de controle de interrupção intencional das iterações através do painel de controle.

iii) - Execute um VI que determine a execução dos itens (i) e (ii) em sequência. Use o valor s_T da primeira sequência item (i) para determinar o fim de contagem do item (ii) na segunda sequência.

iv) - Execute um VI em que uma condição lógica for *True* executa o item (i) e se for *False* executa o item (ii), ou seja, *True* executa FOR e *False* executa WHILE. Indique no Painel uma mensagem tipo *String* com a indicação de qual condição foi executada.

v) - Execute um VI com FORMULA-NODE envolvendo algum tipo de avaliação de uma expressão matemática algébrica, trigonométrica, exponencial etc. Consulte o *Help* para verificar as funções disponíveis. Exemplo as tarefas For ou WHILE realizadas nos

itens (i) ou (ii).

vi) - Execute um VI que dependendo de uma entrada de controle execute o VI do item (i) ou do item (ii) ou do item (v) : Estrutura tipo CASE numérico ou *String*. Novamente indique com uma mensagem *String* apropriada qual CASE foi executado.