MEC2115, cours #4
LabVIEW

4ième cours

Boucle FOR et groupage des données

(Tableaux, Clusters, Chaînes)

Groupage des données

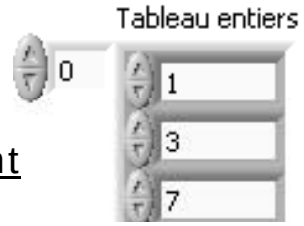
Trois façons dans LabVIEW

1. Tableaux (données du même type)
2. Clusters (données de types différents)
3. Chaînes (caractères ASCII)

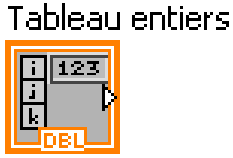
Tableaux (Arrays)

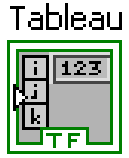
- Un tableau est défini par des éléments et des dimensions
- Les éléments sont les données qui constituent le tableau
- Une dimension est la longueur, la hauteur ou la profondeur d'un tableau
- Un tableau peut avoir une ou plusieurs dimensions et jusqu'à $(2^{31}) - 1$ éléments par dimension
- Vous pouvez construire des tableaux de données:
  - Numériques, booléens, chaînes, chemins
  - Waveforms , clusters

Ex. Face-avant



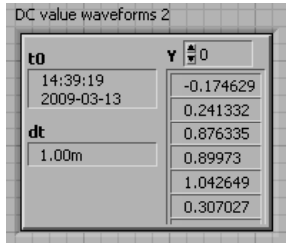
Diagramme





Tableaux (suite)

- Utilisez des tableaux lorsque vous travaillez avec beaucoup de données semblables et lorsque vous effectuez des calculs répétitifs
- Les tableaux sont utiles pour stocker des données collectées (ex. **Y** dans un waveforms) ou générées par des boucles où chaque itération produit un élément du tableau

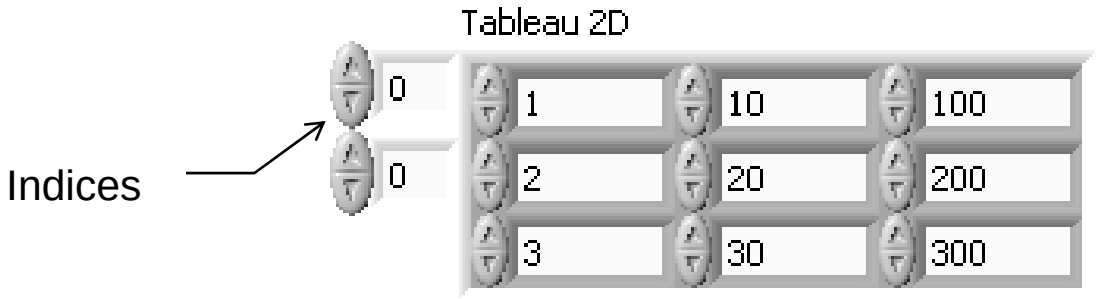


Restrictions

- Il n'est pas possible de créer des tableaux de tableaux et des tableaux de graphes
- Vous pouvez contourner cette restriction en créant un tableau de clusters où chaque cluster contient un ou plusieurs tableaux et des graphes

Indices

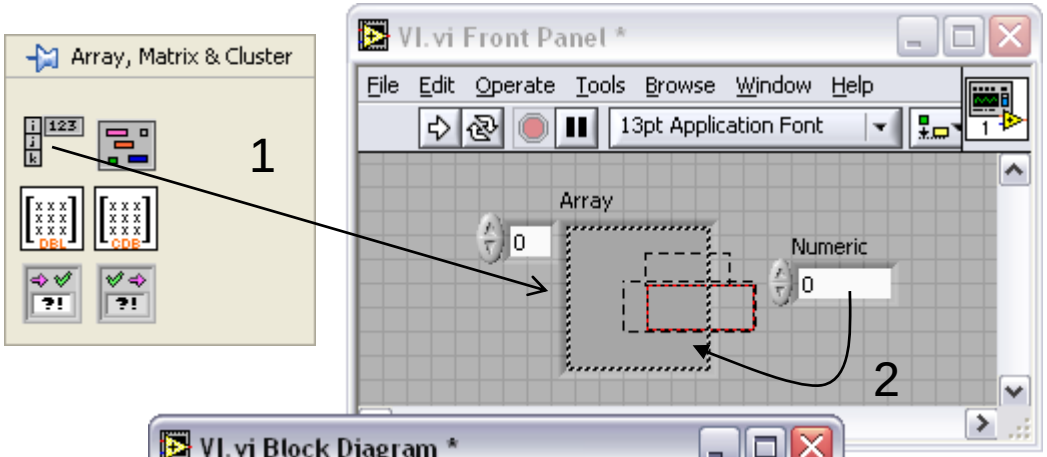
- Les éléments de tableaux sont ordonnés. Un tableau utilise des indices pour que vous puissiez accéder facilement à tout élément particulier.
- L'indice d'une dimension démarre à zéro, ce qui signifie qu'il se trouve dans la gamme de 0 à *n* – 1, *n* étant le nombre d'éléments dans cette dimension.



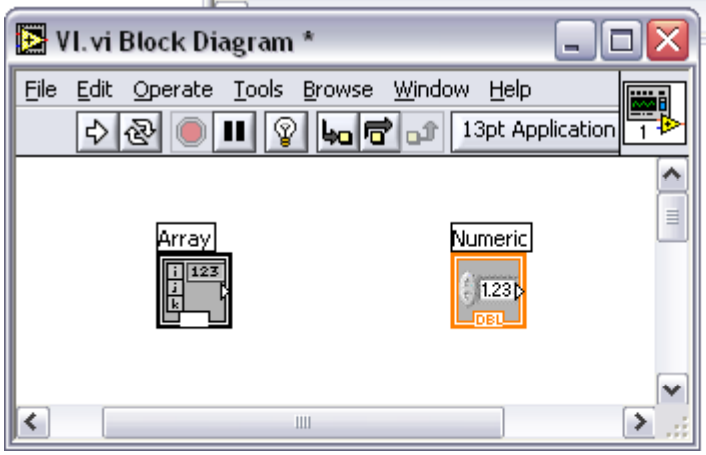
Ex. de tableau 2D (2 indices)

Création d'un tableau
(commande ou indicateur)

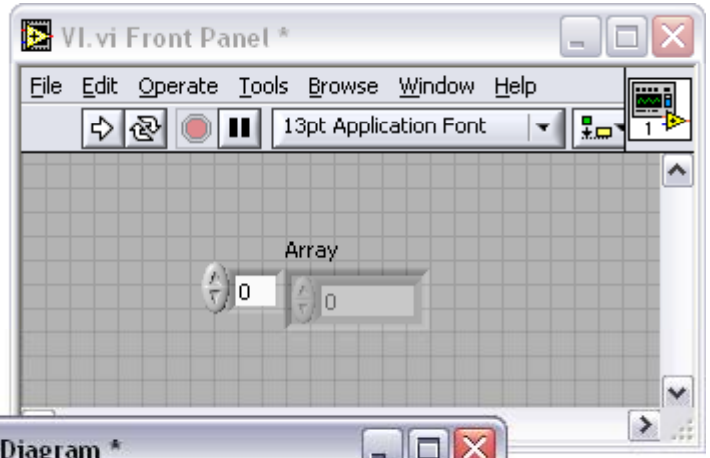
- Ajoutez un tableau (*array*) sur la face-avant. À cette étape, le tableau n'est pas associé à aucun type d'élément.
- Dans la palette des commandes, choisir l'élément de commande ou d'indicateur de votre choix (numérique, booléen, chaîne, etc.) et le glisser à l'intérieur du tableau
- Selon le type d'élément inséré, le tableau devient une commande ou un indicateur
- Par défaut, le tableau aura une seule dimension. On peut ajouter d'autres dimensions (ou indices) à l'aide du menu local du tableau (`add dimension`).

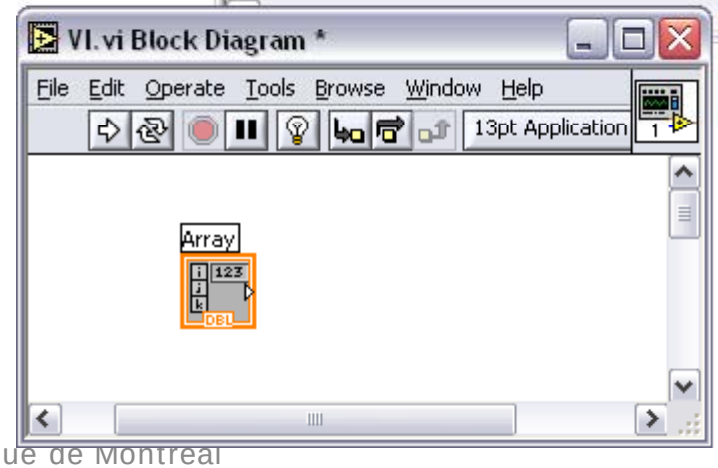


Étapes de création d'un tableau de valeurs numériques (commande)



Résultat final

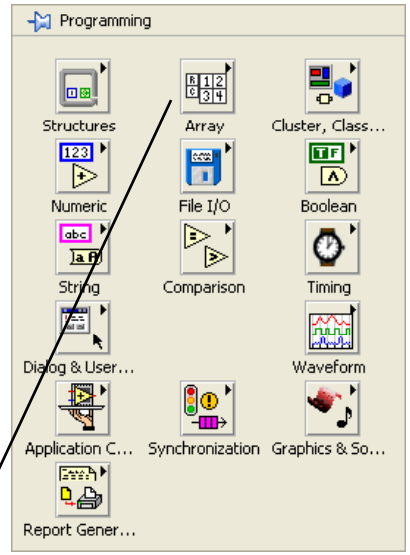


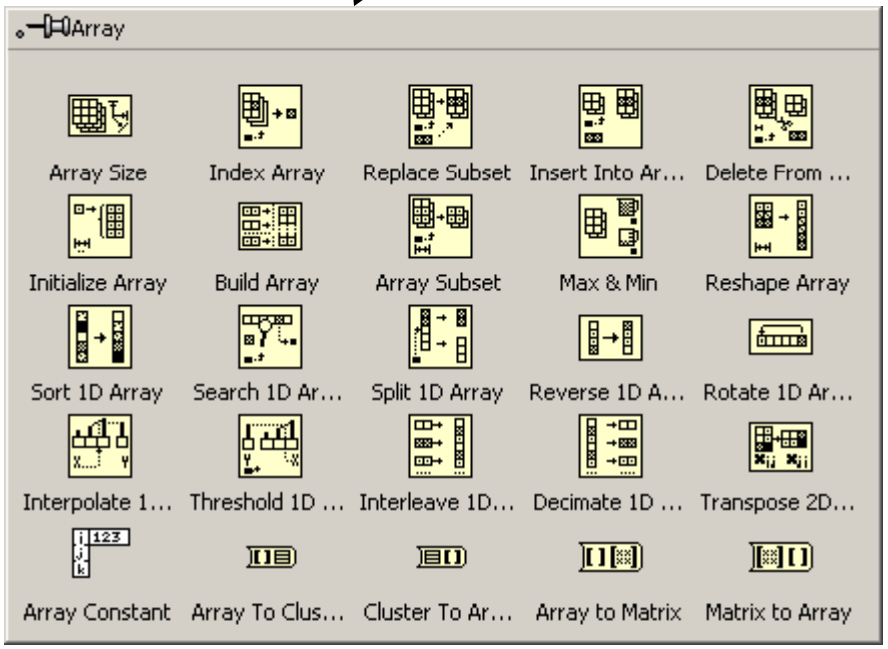


Fonctions de tableau
(Array functions)

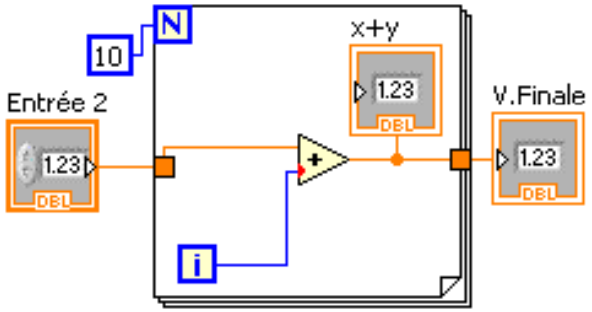
Servent à:

- Extraire des éléments d'un tableau
- Insérer, supprimer ou remplacer des éléments dans un tableau
- Créer, diviser des tableaux
- Pour une brève description des fonctions les plus utiles, voir le "Guide de l'étudiant", Chap. 2.



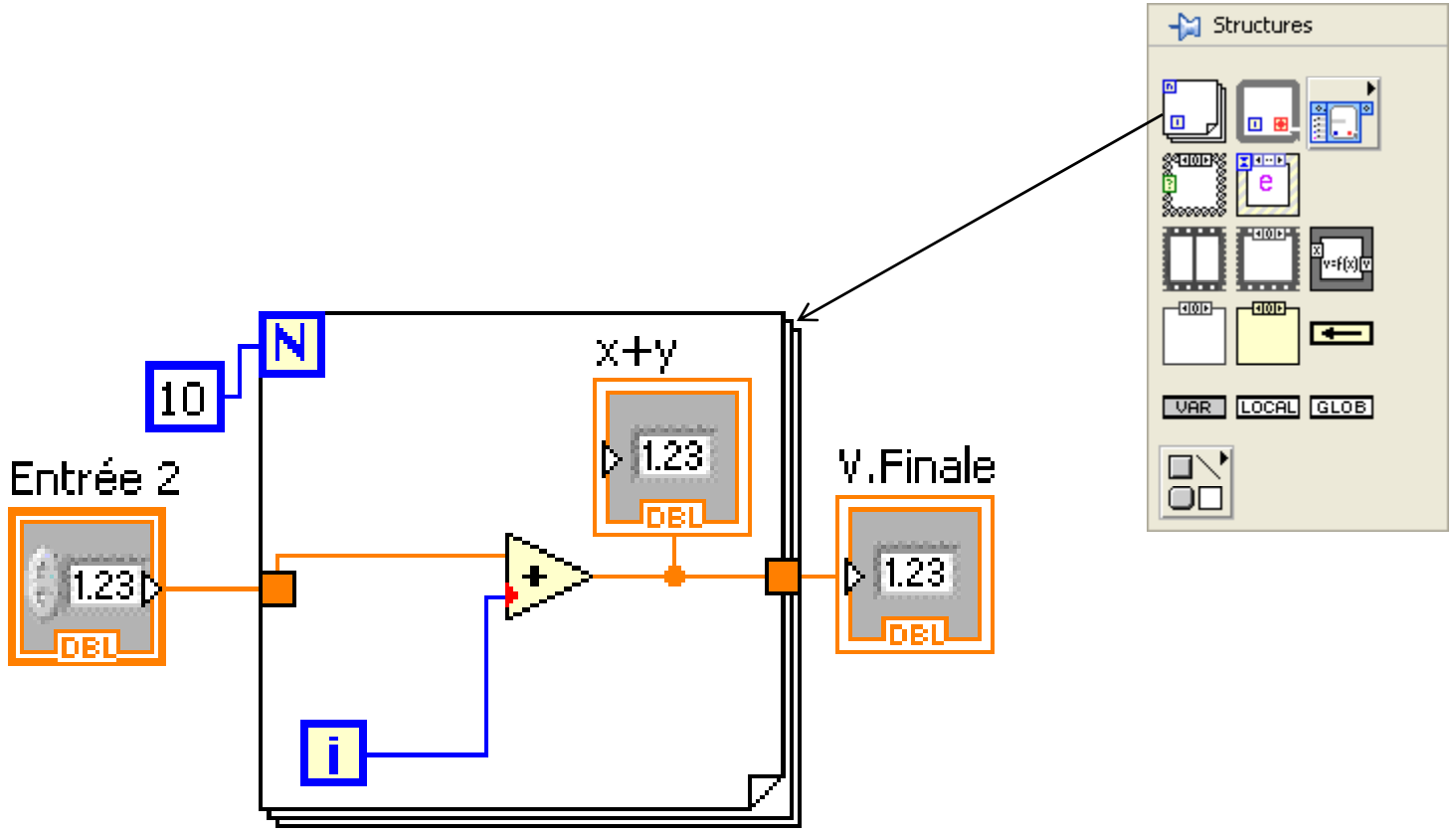


Boucle FOR



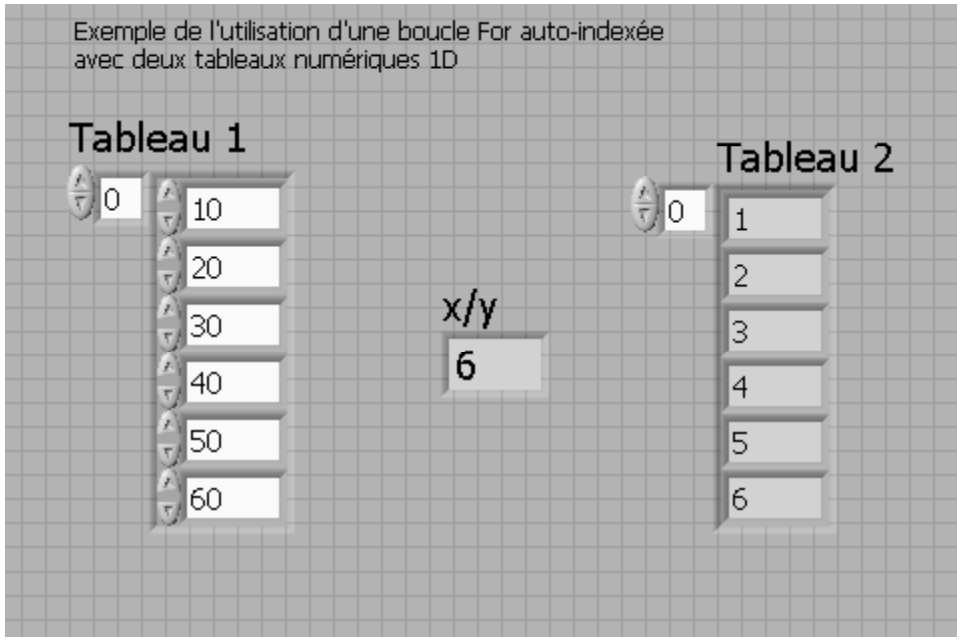
- Une boucle FOR exécute son sous-diagramme le nombre de fois défini par le terminal de comptage **N** (un terminal d'entrée)
- Le terminal d'itération **i** (un terminal de sortie), contient le nombre d'itérations achevées. Le comptage démarre toujours à zéro.
- Les terminaux de comptage et d'itération sont tous les deux des entiers signés 32 bits (2^{32} itérations max.)
- Les données d'entrée et de sortie passent par des tunnels situés sur le pourtour du cadre de la boucle
- On trouve la boucle FOR dans la sous-palette `Programming>Structure`

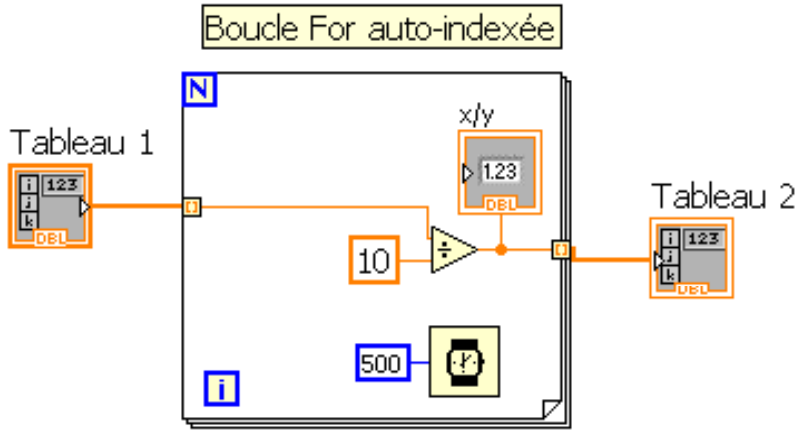
Boucle FOR (suite)



(Rappel)
Auto-indexation, boucles FOR et WHILE

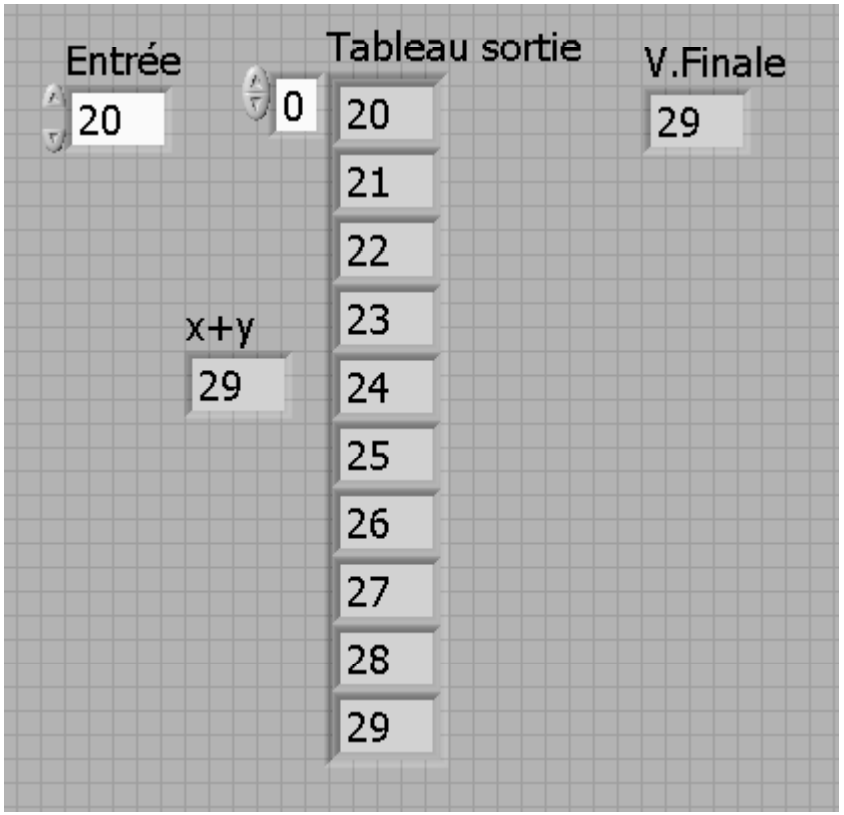
- **Tunnel d'entrée auto-indexé:** Lorsqu'un tableau (commande) est connecté au tunnel, la boucle lit chaque élément du tableau à raison d'un élément par itération, en commençant par le premier élément du tableau.
- Dans le cas d'une boucle FOR, si on désire lire tous les éléments d'un tableau, ne relier aucune valeur au terminal de comptage N.
- Un **tunnel de sortie auto-indexé** reçoit un nouvel élément par itération. Les éléments sont accumulés dans un tableau qui sera envoyé hors de la boucle après la dernière itération. Les boucles FOR et WHILE peuvent servir à remplir des tableaux grâce à l'auto-indexation.
- Si l'auto-indexation est désactivé (tunnel normal):
  - un tableau relié à un terminal d'entrée est lu en entier lors de la première itération
  - un tableau relié à un terminal de sortie est écrit avec les données de la dernière itération

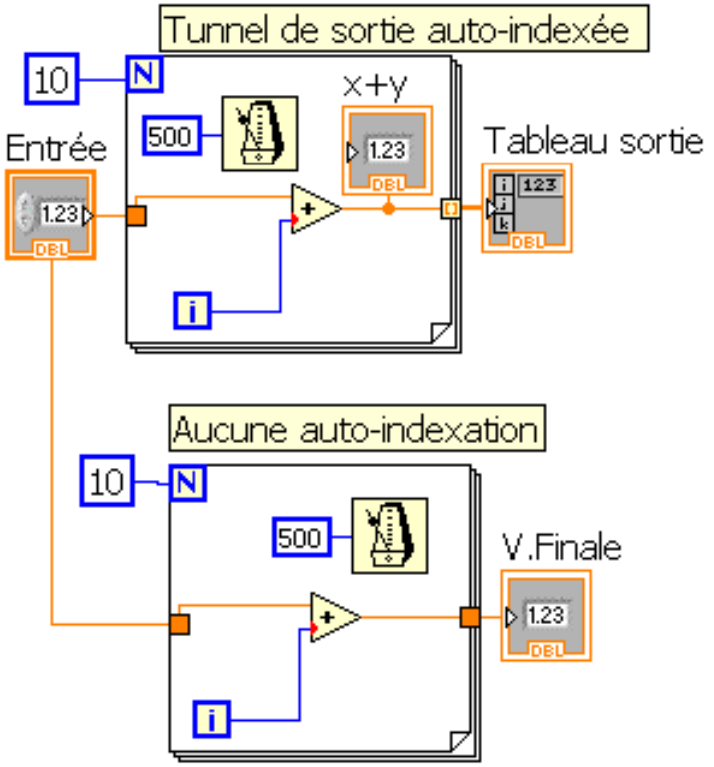




Réf.: LV_cours4_FOR_autoindex1_H10.vi

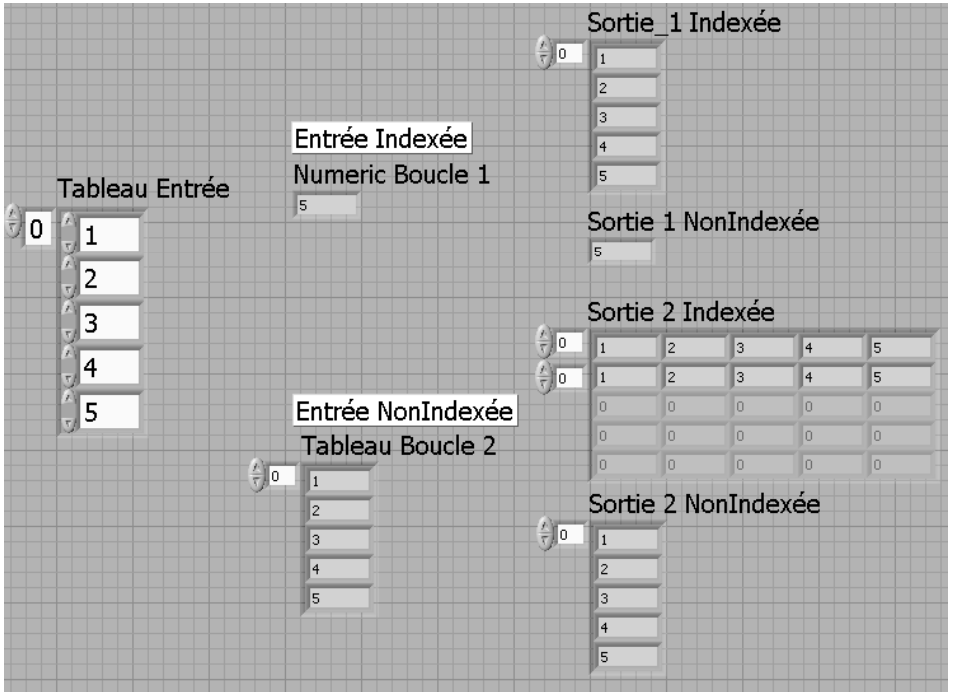
Tunnel de sortie
auto-indexé vs non auto-indexé (normal)

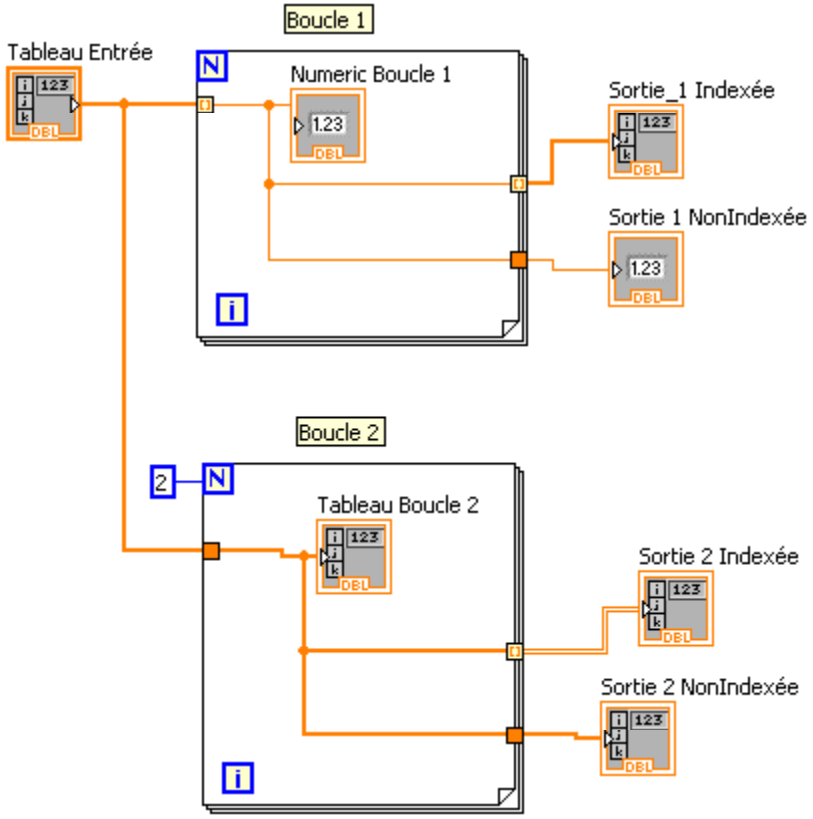




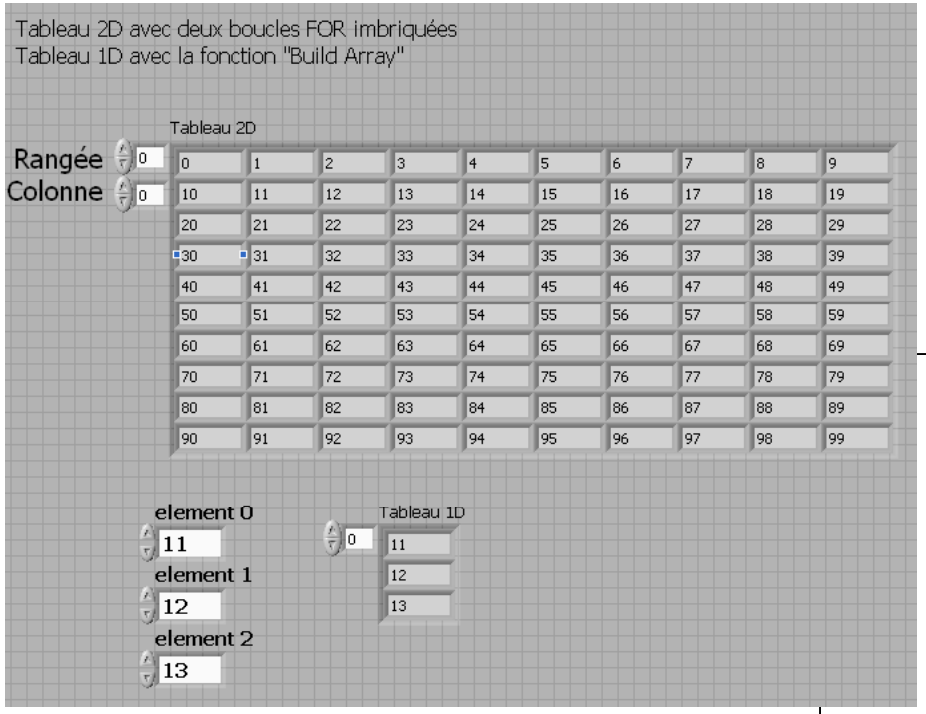
Réf.: LV_cours4_FOR_autoindex2_H10.vi

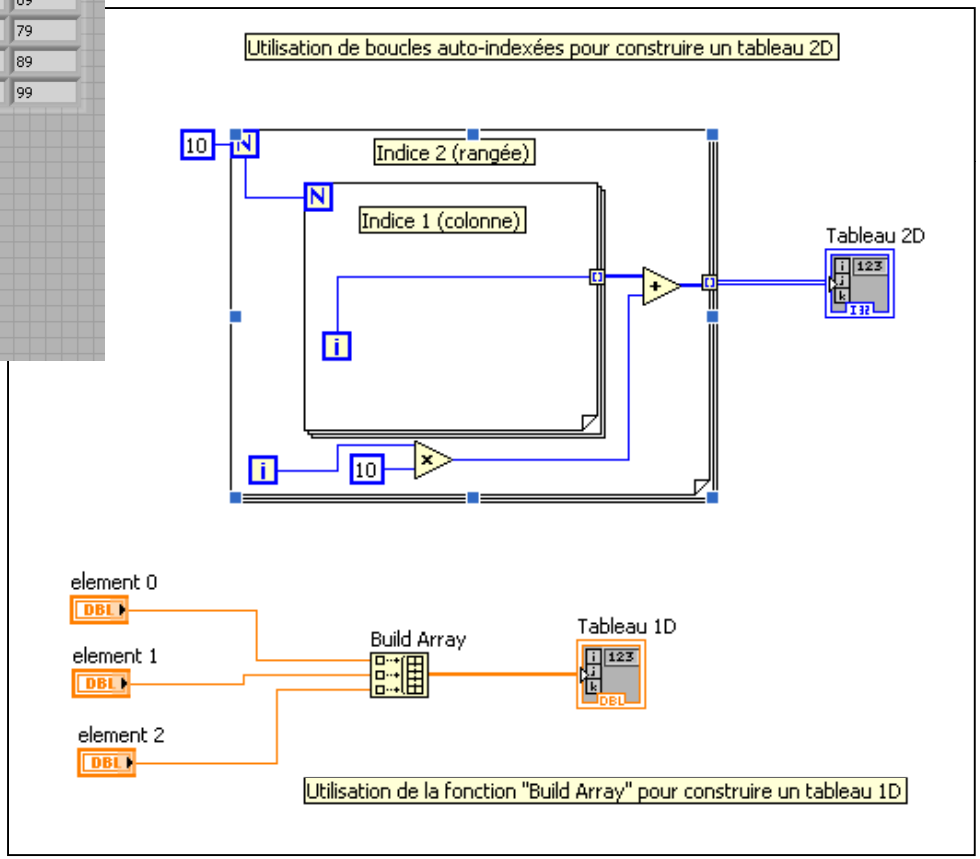
Autres exemples de l'effet du type de tunnel (auto-indexé et normal)





Réf.: LV_cours4_FOR_Tabl2D-1_H10.vi





Réf.: LV_cours4_FOR_Tabl2D-2_H10.vi

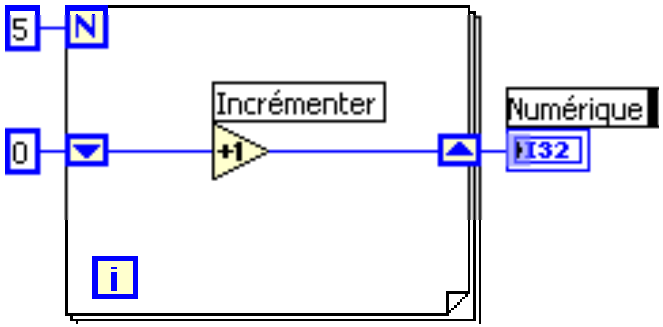
Registres à décalage (Shift registers)

Pour boucles FOR et WHILE

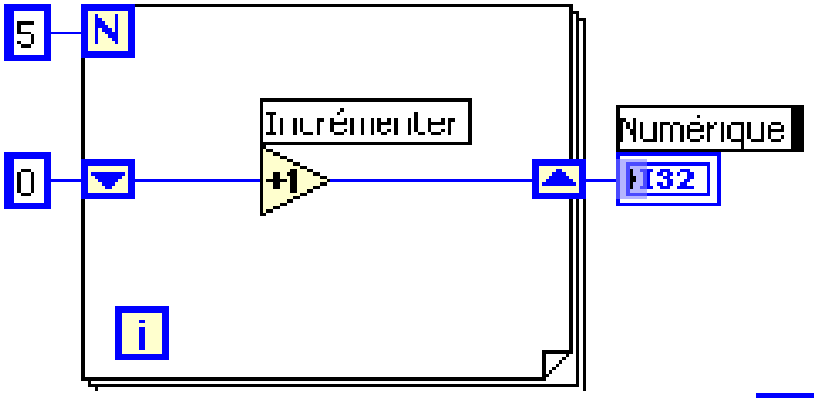
- Les registres à décalage sont utile lorsqu'on veut faire passer la valeur de l'itération précédente à l'itération suivante

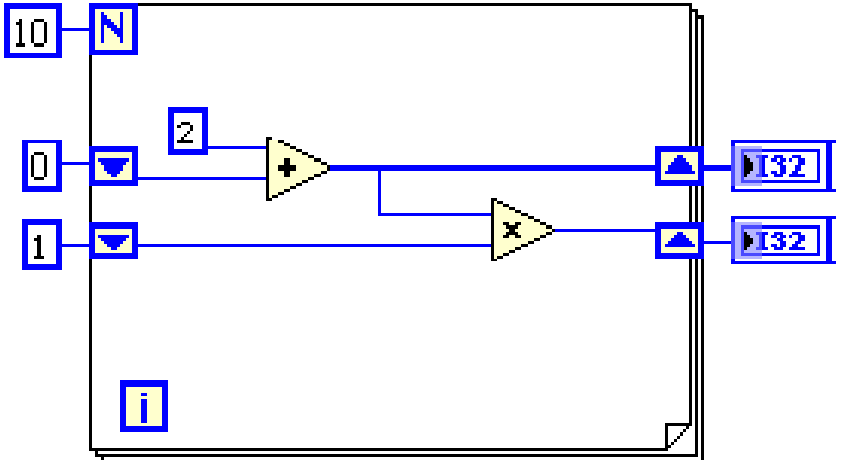
- Un registre à décalage apparaît sous la forme d'une paire de terminaux, représentés, directement à l'opposé l'un de l'autre sur les côtés verticaux du cadre de la boucle

- Pour créer un registre, ouvrir le menu local de la boucle et choisir `Add Shift Register` ou bien cliquer sur un terminal existant et choisir `Replace with Shift Register`



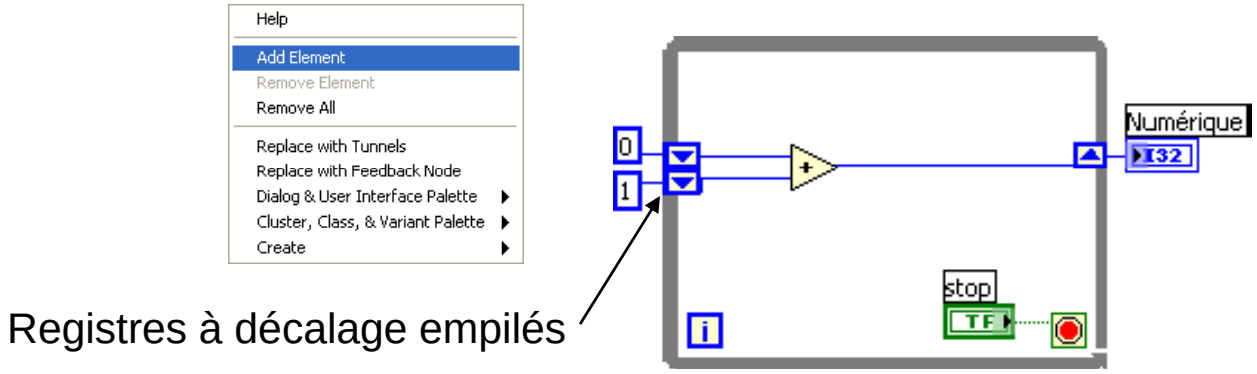
Une boucle peut avoir un ou plusieurs registres à décalage



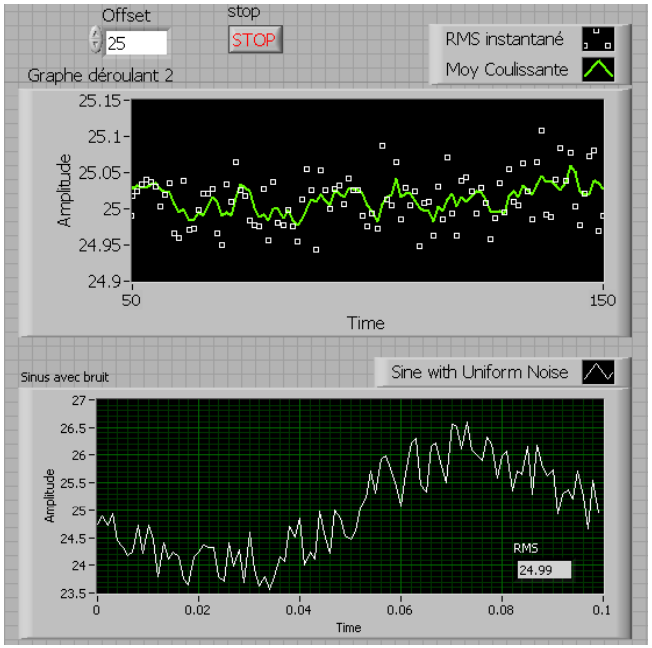


Registres à décalage empilés
(Stacked Shift Registers)

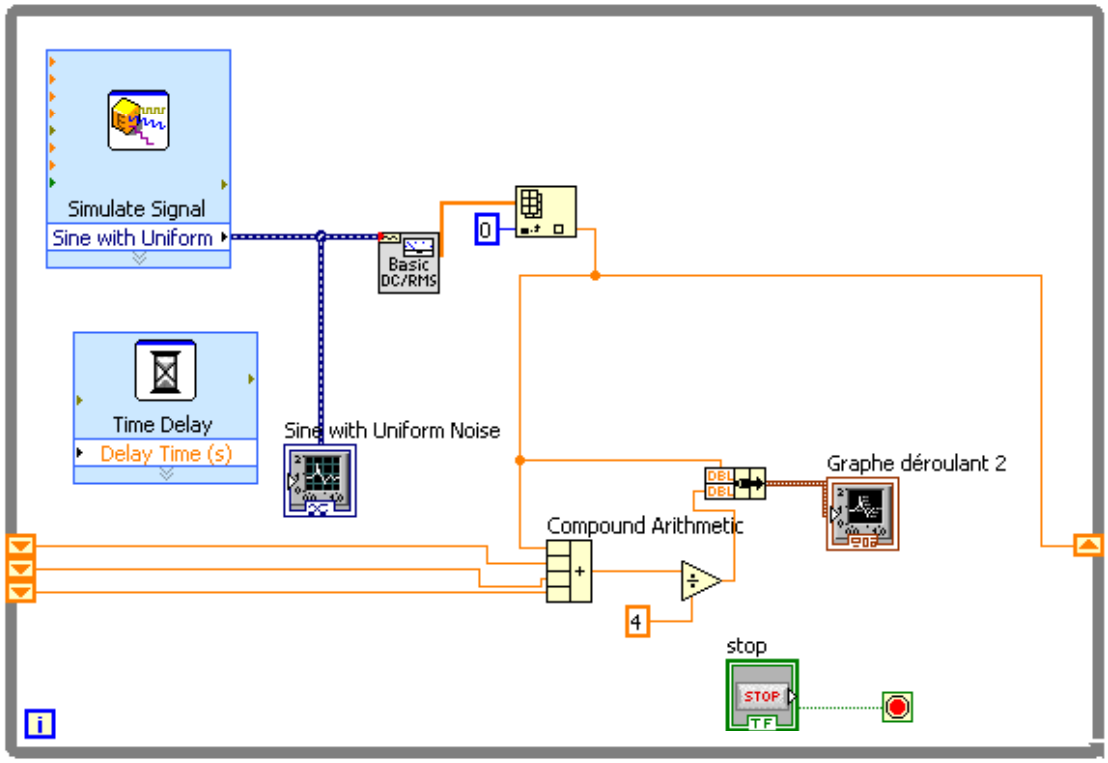
- Les registres à décalage empilés conservent en mémoire les valeurs de multiples itérations précédentes et transmettent ces valeurs aux itérations suivantes
- Pour créer un registre à décalage empilé, cliquez droit sur le terminal gauche du registre et sélectionnez `add element` dans le menu local



Exemple : Calcul de la moyenne coulissante des quatre dernières valeurs RMS avec des registres à décalage empilés



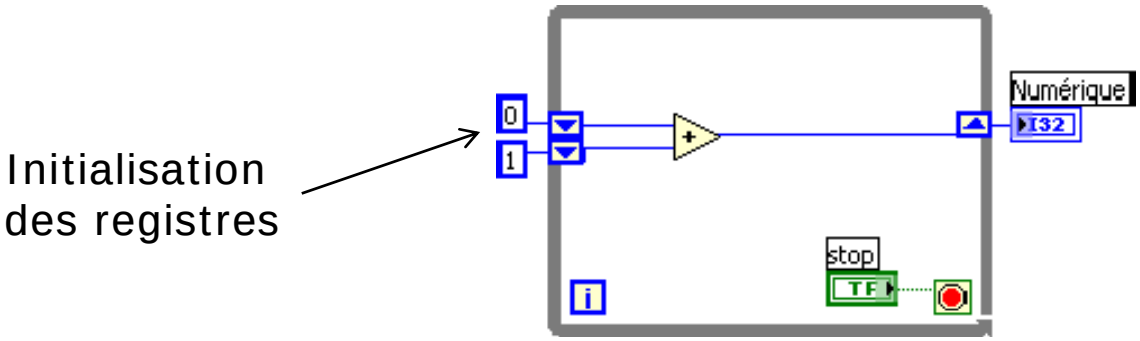
Réf.: LV_cours4_Moycoulissante_H10.vi



Registres à décalage empilés

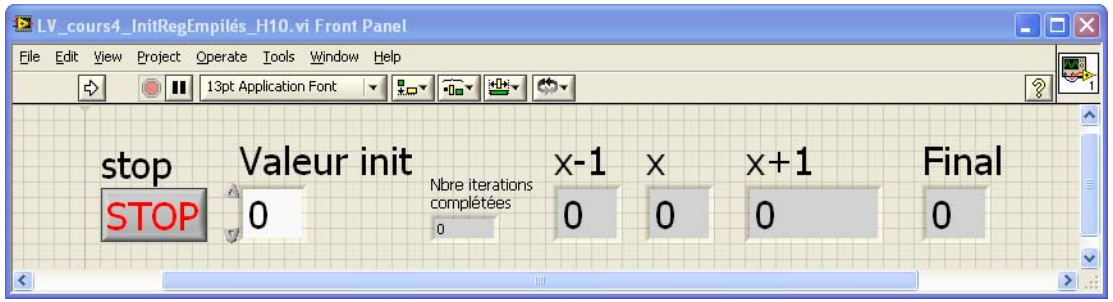
Initialisation des registres à décalage

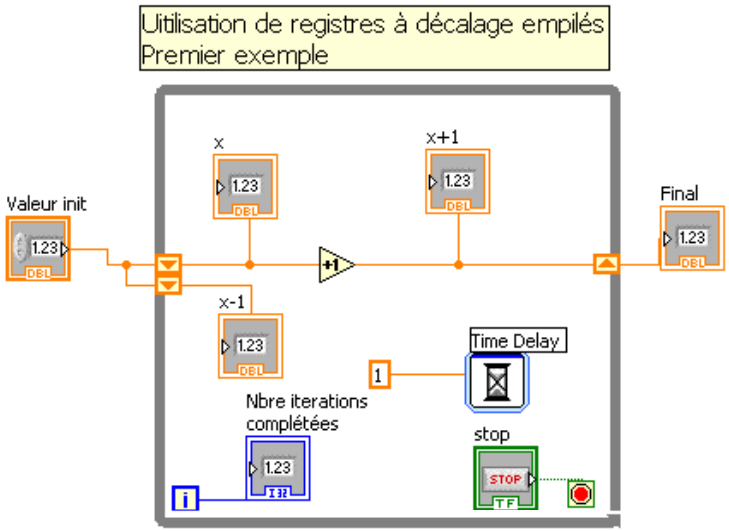
- L'initialisation d'un registre à décalage définit la valeur que le registre à décalage transmet à la boucle lors de sa première itération
- Initialisez un registre à décalage en câblant une commande ou une constante au terminal d'entrée du registre
- Si vous n'initialisez pas le registre à décalage, la boucle utilise la valeur écrite dans le registre lors de la dernière exécution de la boucle (utile pour conserver les informations d'état pour les exécutions ultérieures)



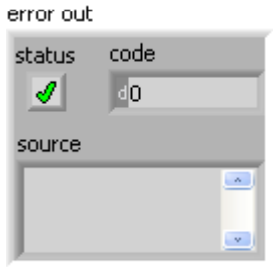
Exemple: Effet de l'initialisation des registres à décalage

Réf.: LV_cours4_IntRegEmpilés_H10.vi





Cluster

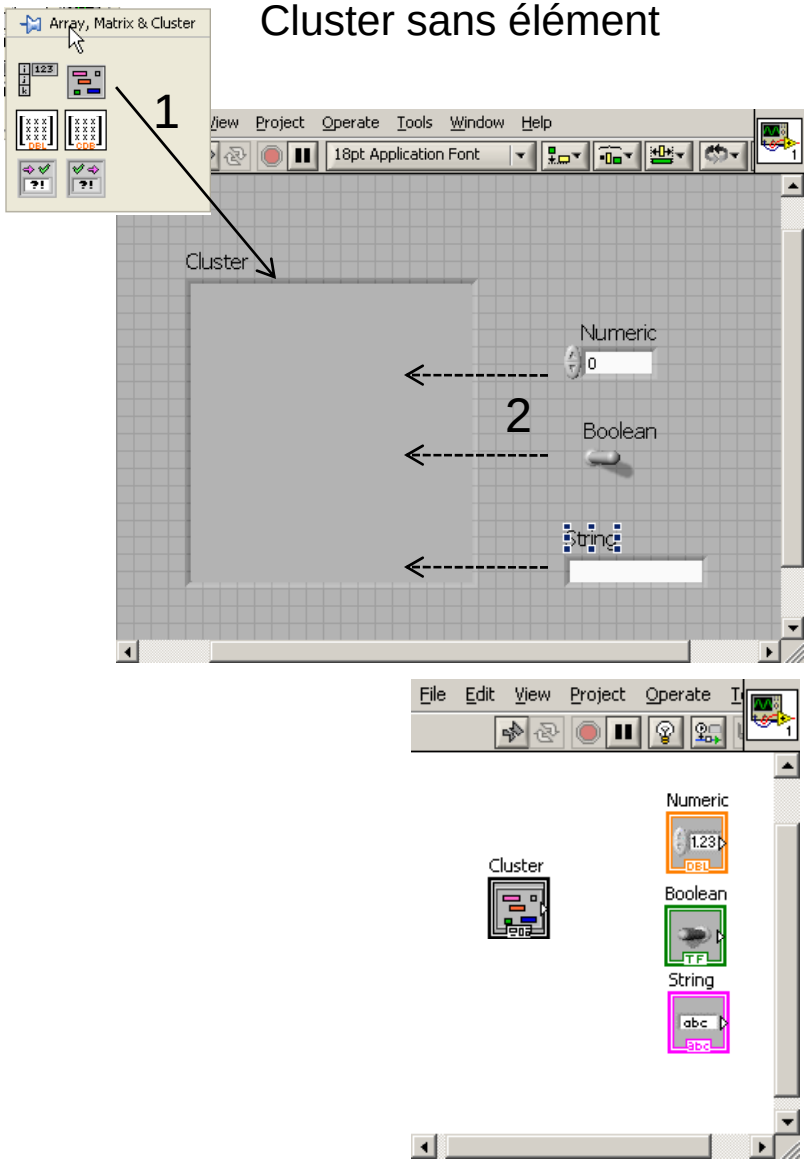


- Un cluster regroupe des éléments de données de types différents
- Le cluster d'erreur LabVIEW est un exemple de cluster; il comprend une valeur booléenne, une valeur numérique et une chaîne
- L'assemblage de plusieurs éléments de données dans des clusters :
  - élimine l'encombrement des câbles sur le diagramme
  - réduit le nombre de terminaux de connexion nécessaires pour les sous-VIs

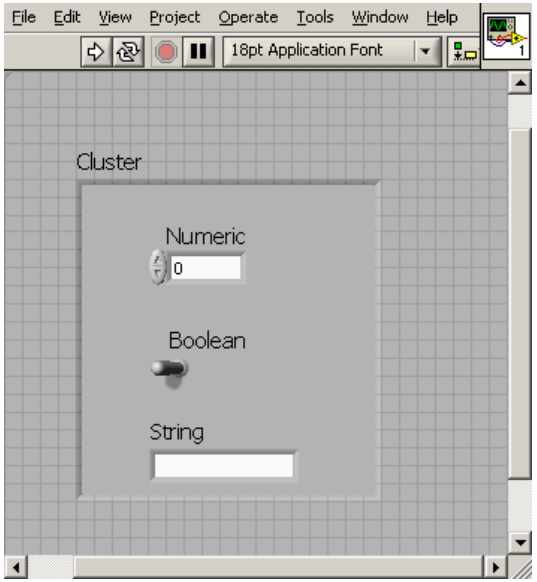
Créer un cluster

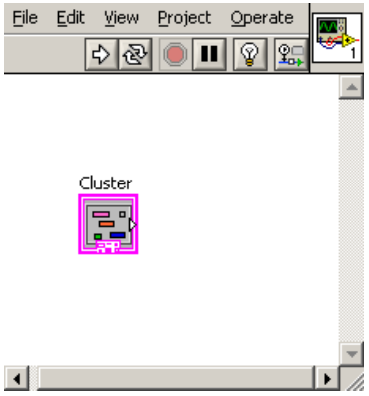
- Ajoutez un cluster sur la face-avant à partir de la palette des commandes (`Modern>Array,Matrix,Cluster.` À cette étape, le cluster est vide.
- Dans la palette des commandes, choisir les commandes ou indicateurs de votre choix et les introduire dans le cluster
- Vous pouvez aussi introduire dans un cluster des commandes ou indicateurs qui existent déjà sur la face-avant
- Le type du premier élément introduit dans le cluster, une commande ou un indicateur, détermine le type du cluster (commande ou indicateur). Tous les autres éléments du cluster seront du même type.

Cluster sans élément



Cluster comprenant 3 éléments de type commande pris sur la face-avant

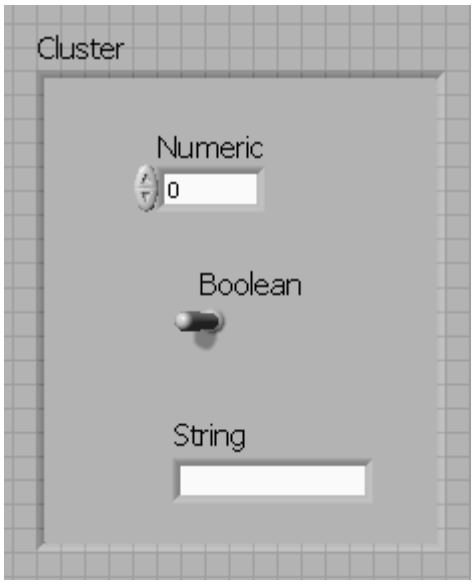


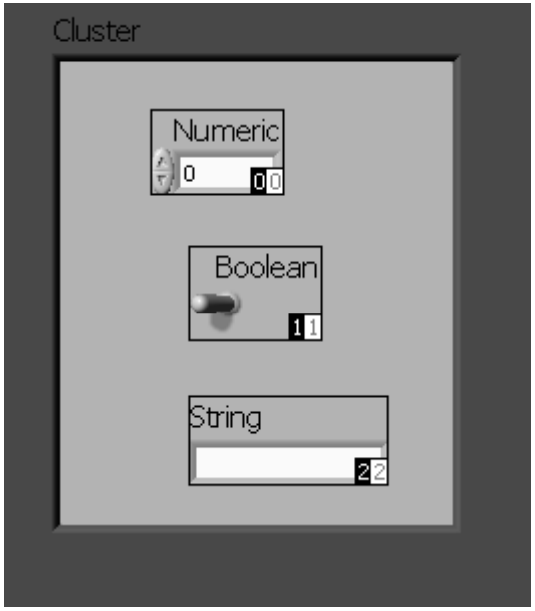


Ordre des éléments d'un cluster

- Comme les éléments d'un tableau, les éléments d'un cluster sont ordonnés. Ainsi, le premier objet que vous placez dans le cluster est l'élément 0, le deuxième est l'élément 1 et ainsi de suite.
- Vous pouvez afficher et modifier l'ordre des éléments du cluster en utilisant le menu local (`Reorder Controls in Cluster`)
- Il faut désassembler (`unbundle`) tous les éléments d'un cluster pour pouvoir accéder aux éléments individuels d'un cluster. Utiliser les fonctions `Unbundle` ou `Unbundle by Name` qui se trouvent dans la sous-palette `Programming >Cluster, Class,& Variant`

Exemple de l'ordre des éléments d'un cluster

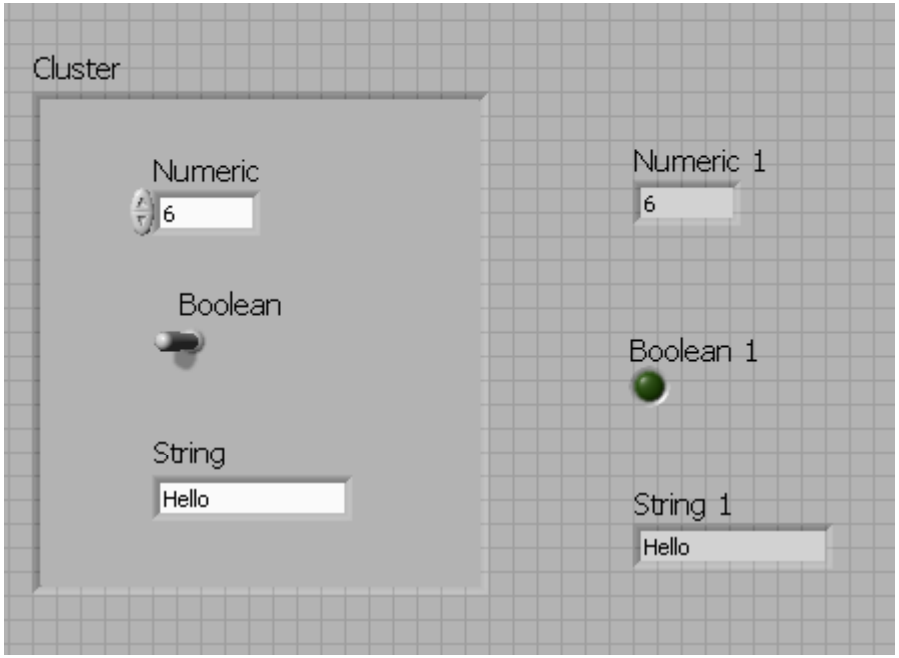




Faîtes apparaitre cette fenêtre avec:
Menu local > `Reorder Controls in Cluster`

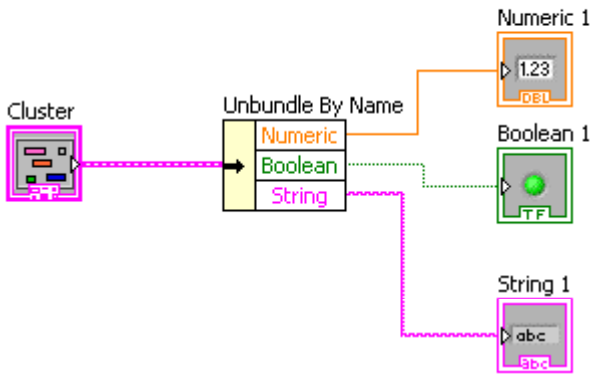
Désassemblage (`Unbundle` ou `Unbundle by name`) des éléments d'un cluster

Les trois éléments de commandes contenus dans le Cluster sont déassemblés puis affichés individuellement dans trois indicateurs du type approprié.



Face-avant

Diagramme

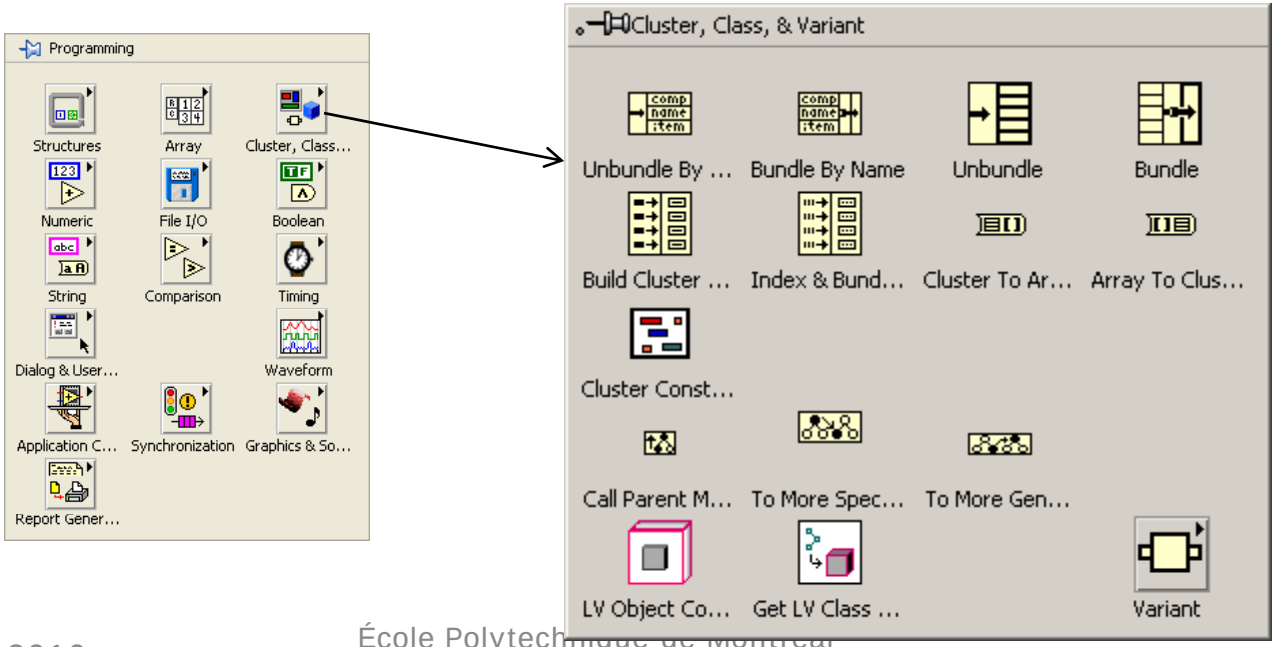


Réf.: LV_cours4_exCluster_H10.vi

Fonctions de cluster

Servent à:

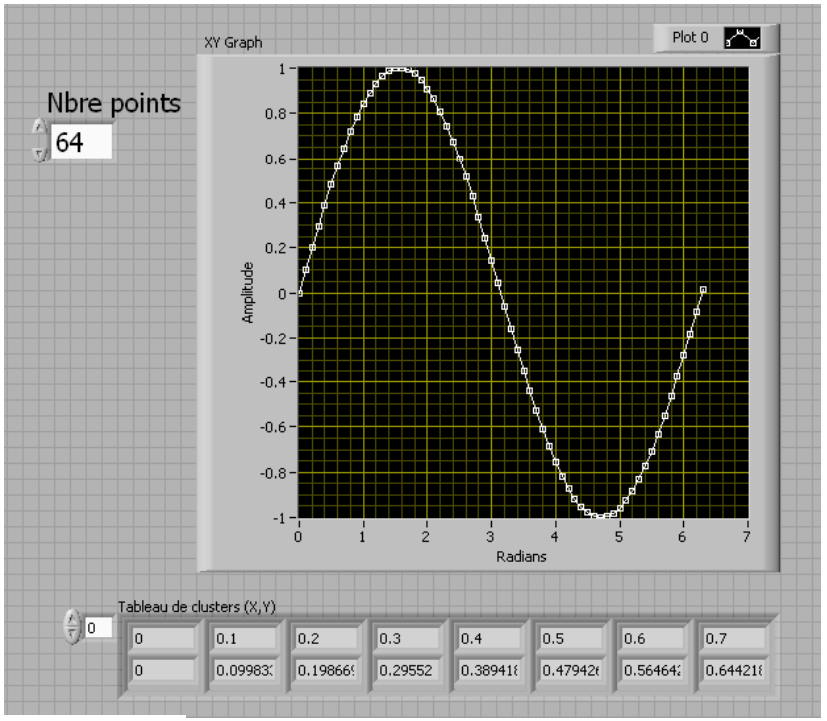
- Extraire des éléments de données individuels
- Ajouter des éléments de données individuels
- Transformer un cluster en tableau et vice-versa



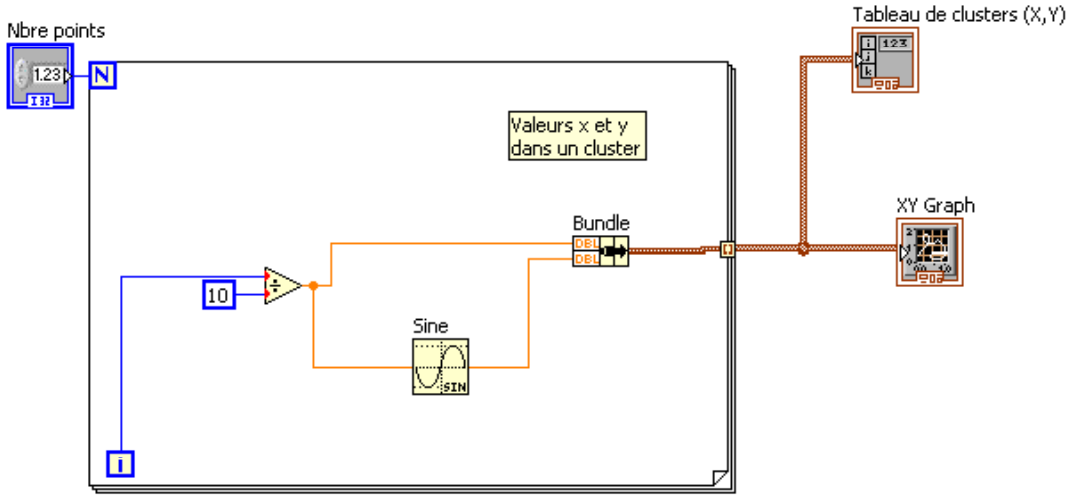
Exemple d'utilisation des clusters:

Générer une courbe dans un Graphe XY avec un tableau de clusters contenant les coordonnées x,y des points de la courbe.

(Voir LV_cours4_GrapheXY_A10.vi)

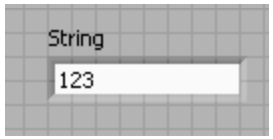


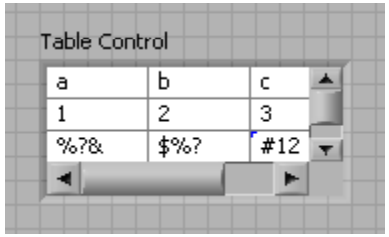
Les coordonnées X-Y de chaque point de la courbe sont groupées dans un cluster à l'aide de la fonction Bundle.
La boucle FOR génère les coordonnées du nombre de points spécifiés par l'utilisateur et les accumule dans un tableau (de clusters) grâce au tunnel de sortie auto-indexé. Après la dernière itération de la boucle FOR, ce tableau est envoyé au Graphe XY et à l'indicateur "Tableau de clusters (X,Y)"



Chaînes (Strings)

- Les chaînes offrent plusieurs possibilités:
  - Création de messages de texte simple
  - Transmission d'instructions à des instruments (ASCII)
  - Stockage des données numériques dans un fichier ASCII (vous devez convertir les nombres en chaînes avant d'écrire dans le fichier)
- Sur la face-avant, les chaînes apparaissent comme des boîtes d'entrée de texte et des tables (tableau)

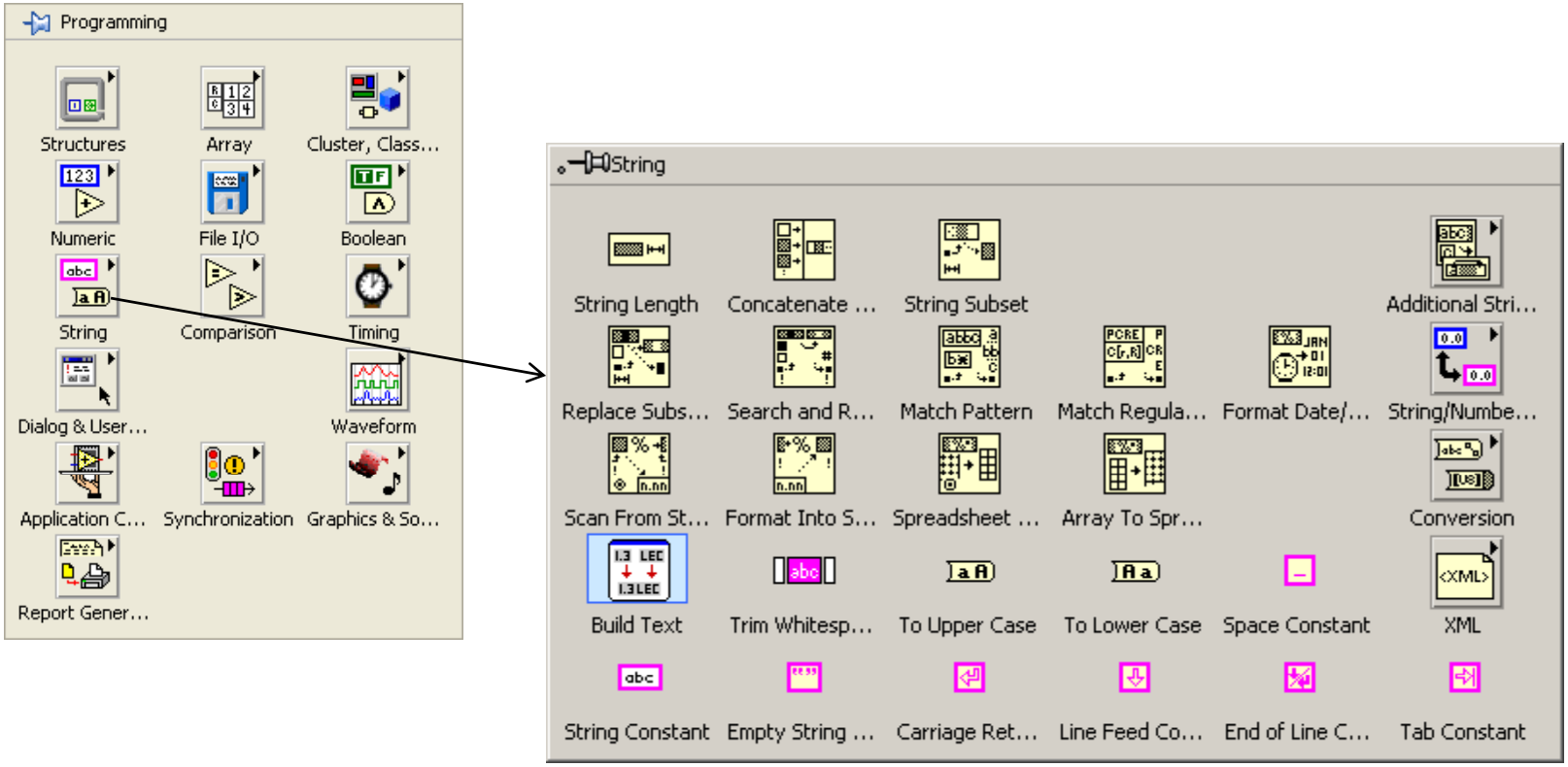




Fonctions Chaîne (String functions)

- Utilisez les fonctions Chaîne pour réaliser des tâches suivantes
  - Extraire un sous-ensemble de caractères
  - Convertir des données numériques en chaînes
  - Formater une chaîne pour pouvoir l'utiliser ensuite avec un traitement de texte ou un tableur
  - Utiliser les VIs et les fonctions d'E/S sur fichiers pour enregistrer des chaînes dans des fichiers texte et des fichiers de type tableur

Sous-palette des fonctions "String"

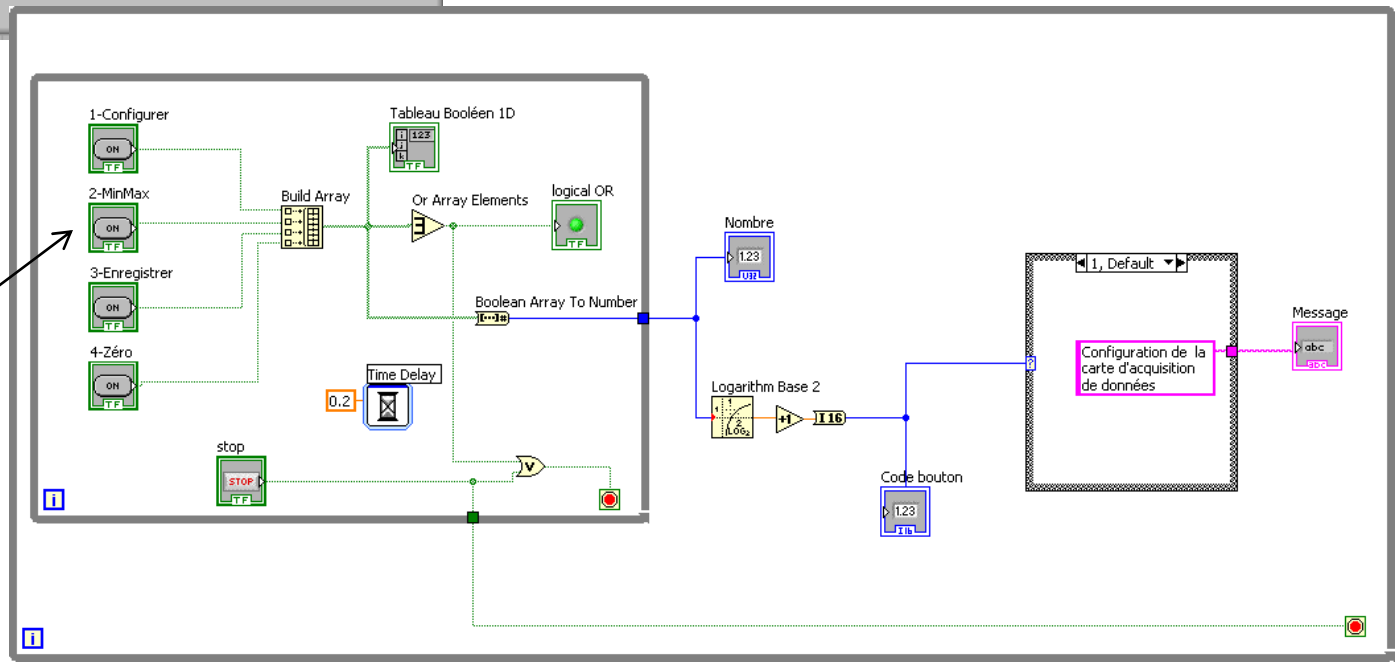


VI compléments au cours #3

- Les deux premiers VI qui suivent montrent comment on peut traiter des choix multiples avec une seule structure condition. Les méthodes montrées sont particulièrement utiles lorsque l'utilisateur du programme peut appuyer sur plusieurs boutons pour faire son choix.
- Le troisième exemple est un sous-VI qui simule le l'output des ponts de Wheatstone qui sont utilisés avec les quatre jauges de la poutrelle du TP-4: Simulation_Voltages_Ponts.vi

Coder des commandes booléennes avec un algorithme de type "*State Machine*" (LV_cours4_StateMachine_A10.vi)

Note:
On peut faire la même chose en utilisant une commande Boutons Radio, ce qui simplifie le diagramme (Voir diapo suivante)

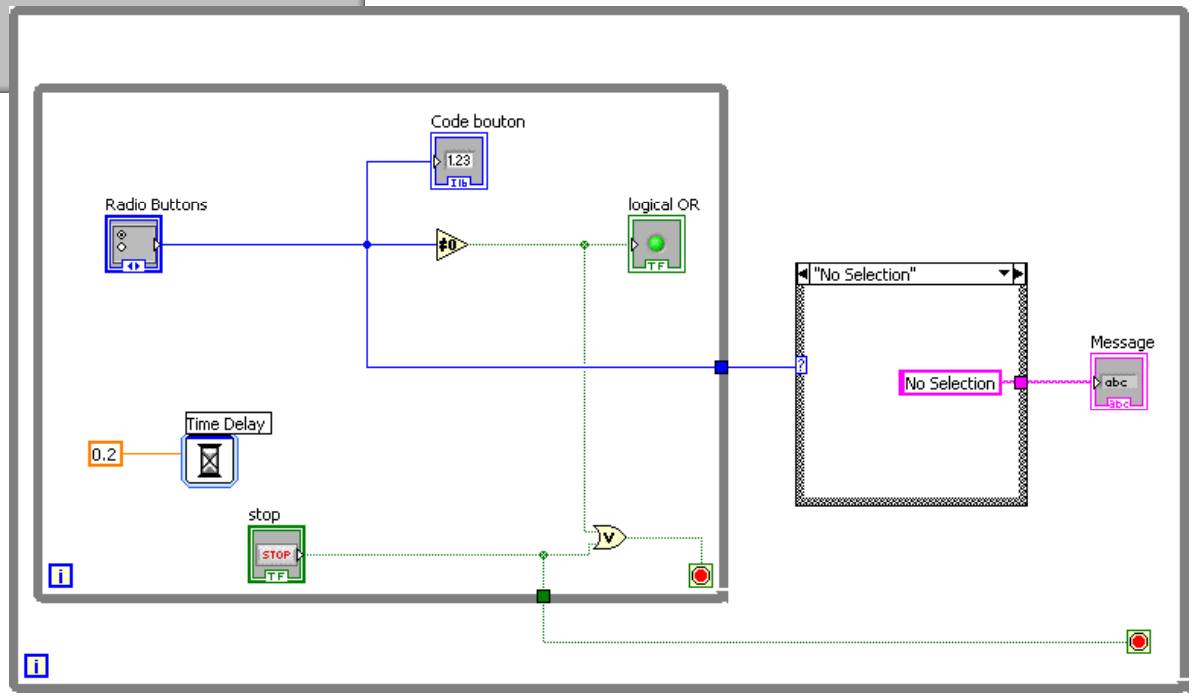


Commandes Booléennes à armement

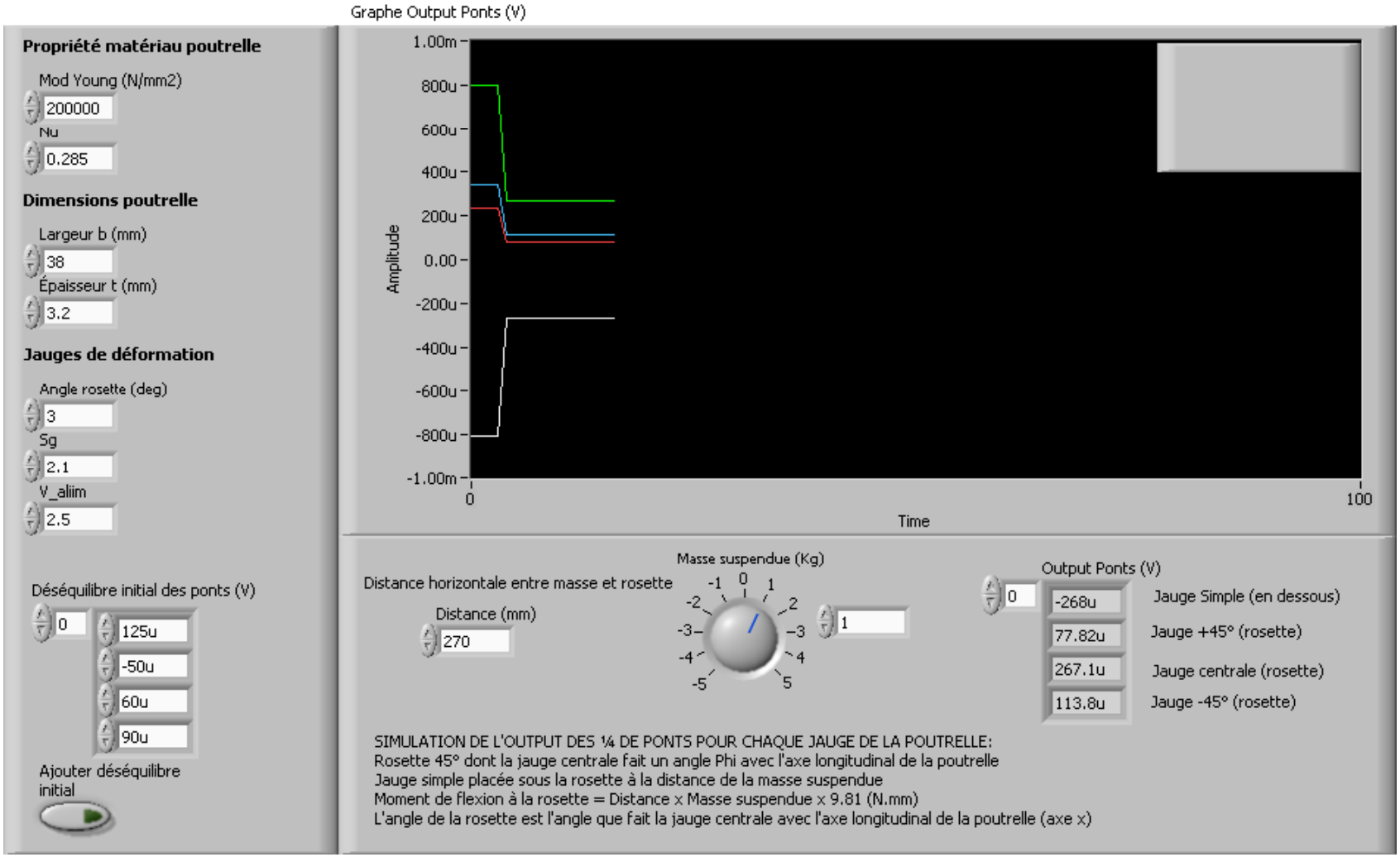
Utilisation d'une commande Boutons Radio (LV_cours4_RadioButtons_A10.vi)

Exemple démontrant comment on peut coder des choix multiples avec une commande de type Boutons Radio dont la valeur de sortie sera connectée à une structure Condition qui traite les différents choix de l'utilisateur.

La Commande "Radio Buttons" est un cluster de valeurs booléennes et dans le cas présent, l'option "Allow no selection" du menu local de la commande est activée. Cela a pour effet de générer une valeur "0" lorsqu'aucun des boutons n'est pressé.



Simulation_Voltages_Ponts.vi
(Jauges de la poutrelle du TP-4)



Lectures

Boucles FOR et WHILE

- [2], Chapitre 8 au complet

Groupage des données

- [2], Chapitre 9 au complet
- Guide de l'étudiant, Chap. 2

Exercices avec des structures Boucle

Exercice 1

But: Utiliser une boucle WHILE, son terminal d'itération et un tunnel de sortie

Scénario: Construire un VI qui génère continuellement des nombres aléatoires entre 0 et 1000 jusqu'à ce que le nombre généré égale celui qui a été choisi par l'utilisateur. Déterminer combien il a fallu générer de nombres aléatoires avant d'obtenir le nombre choisi.

Note: Pour obtenir des nombres aléatoires entiers compris entre 0 et 1000, utiliser la fonction "`Random Number`" dont le résultat est multiplié par 1000, puis arrondir à la valeur entière la plus près avec la fonction "`Round to Nearest`". Les fonctions requises se trouvent dans la sous-palette "`Numerics`"

Exercices avec des structures Boucle

Exercice 1

But: Utiliser les boucles WHILE et FOR, les terminaux d'itération et les tunnels d'entrée et de sortie

Scénario: Construire un VI qui génère continuellement des nombres aléatoires entre 0 et 1000 jusqu'à ce que le nombre généré égale celui qui a été choisi par l'utilisateur.

1. Déterminer combien il a fallu générer de nombres aléatoires avant d'obtenir le nombre choisi.
2. Répéter l'exercice 1000 fois et conserver les résultats (nombre d'itérations nécessaires pour trouver le nombre choisi) dans un tableau et les afficher dans un graphe.
3. Calculer la moyenne du nombre d'itérations (voir VI dans la sous-palette statistique)

Note: Pour obtenir des nombres aléatoires entiers compris entre 0 et 1000, utiliser la fonction "`Random Number`" dont le résultat est multiplié par 1000, puis arrondir à la valeur entière la plus près avec la fonction "`Round to Nearest`". Les fonctions requises se trouvent dans la sous-palette "`Numerics`"

Exercice 2

But: Choisir entre une boucle WHILE et une boucle FOR, celle qui est la mieux adaptée au scénario à réaliser.

Note: Vous pouvez simuler le signal des capteurs avec un sous-Vi qui génère des valeurs numériques ou encore avec le VI express `Simulate Signal`

a) Nous faisons l'acquisition d'une pression toutes les secondes pendant une minute, puis le programme sort de la boucle et exécute autre chose.
Programmez avec:
- Une boucle FOR
- Une boucle WHILE

Des deux types de boucles, laquelle est la plus facile à programmer?

b) Nous faisons l'acquisition d'une pression à toutes les secondes jusqu'à ce que sa valeur soit de 10 MPa ou plus, puis le programme sort de la boucle d'acquisition. Même question que a)

c) Nous faisons l'acquisition d'une pression et d'une température jusqu'à ce que les deux valeurs soient stables pendant 2 minutes. Même question que a)

d) Nous générons un voltage qui partant de zéro, augmente linéairement de 0.5 V à toutes les secondes jusqu'à un maximum de 5 Volts. Même question que a)

Exercice 3

Nous faisons l'acquisition de la température d'un procédé toutes les secondes. Nous voulons afficher dans un graphe déroulant la température instantanée ainsi que la moyenne coulissante des quatre dernières températures. Développez un VI qui réalisera ces opérations.

Note: Pour réaliser cet exercice vous devez faire appel à une boucle FOR ou WHILE ainsi qu'à des registres à décalage empilés.