MEC2115, cours #5
LabVIEW

5ième cours

Contrôle de l'ordre d'exécution

E/S sur fichiers

Variables locales

Contrôle de l'ordre d'exécution

- Un diagramme est composé de terminaux et de nœuds
- Un nœud est l'élément de base dans l'exécution du diagramme
- Dans le diagramme, c'est le flux de données qui détermine l'ordre d'exécution des nœuds

Objets du diagramme

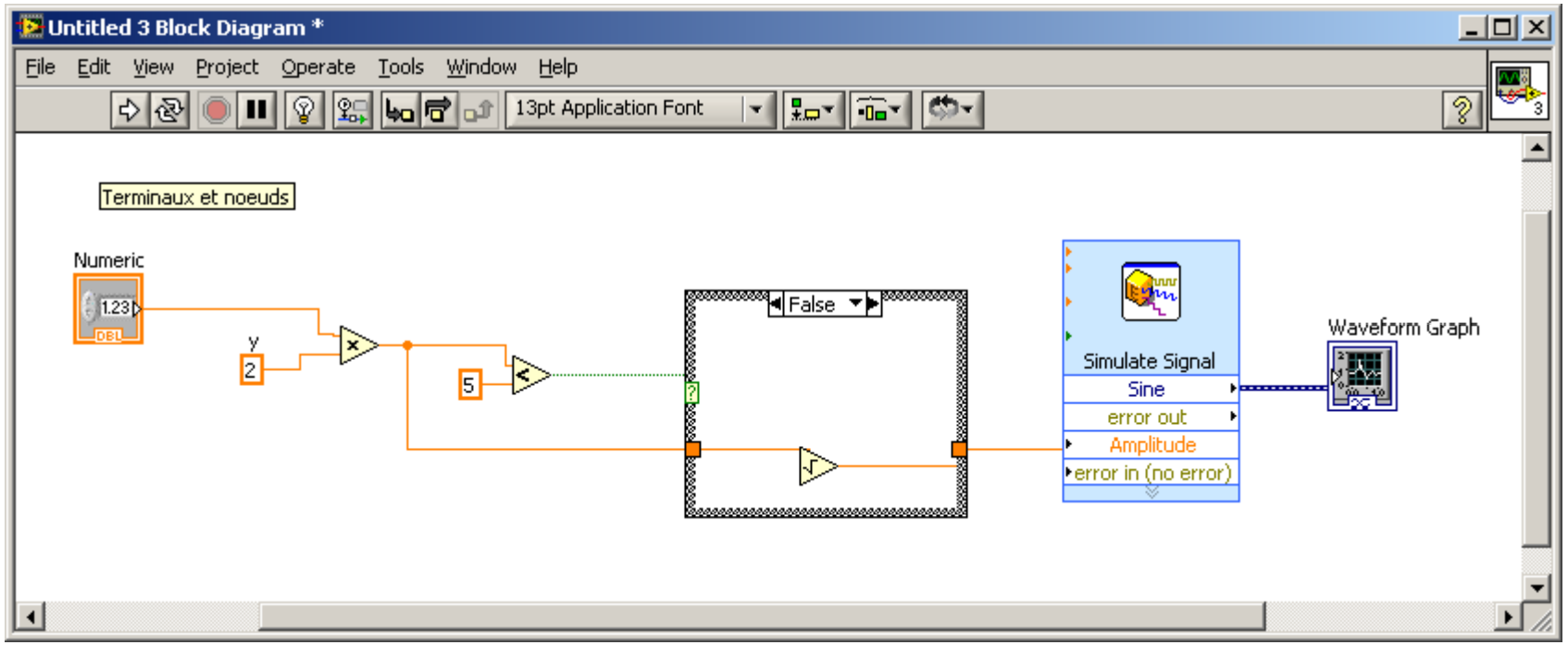
Terminaux *(Terminals)*

- Commandes (Entrées)
- Indicateurs (Sorties)
- Nœuds
- Constantes

Nœuds *(Nodes)*

- Possèdent des entrées et/ou des sorties
- **Réalisent des opérations**
  - Fonctions
  - Sous-VI
  - VI-Express
  - Structures (ex. WHILE, FOR, structure condition, etc.)

Combien de terminaux et de nœuds se trouvent dans ce diagramme?



`LV_cours5_VI1_H10.vi`

Terminaux (dans le diagramme)

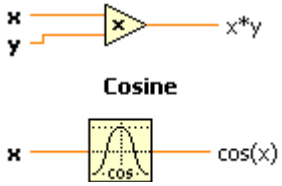
- Les objets de la face-avant apparaissent comme des terminaux de type icône ou de type données

- Les constantes n'existent que dans le diagramme et ont des valeurs fixes
  - Constantes universelles (**π, R, e, etc.**)
  - Constantes définies par l'utilisateur
- Il y a des constantes pour chacun des différents types de données

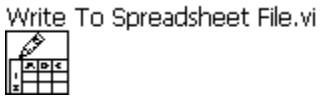
Noeuds

Fonctions, Sous-VI et VI-express

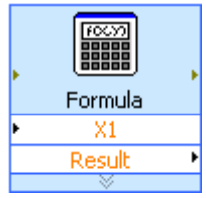
- Les Fonctions
  - Existent en grand nombre (sous-palettes spécialisées)
  - N'ont ni diagramme ni face-avant (non modifiables)



- Les Sous-VI
  - Fournis avec LabVIEW ou créés par l'utilisateur
  - Possèdent une face-avant et un diagramme (modifiables)

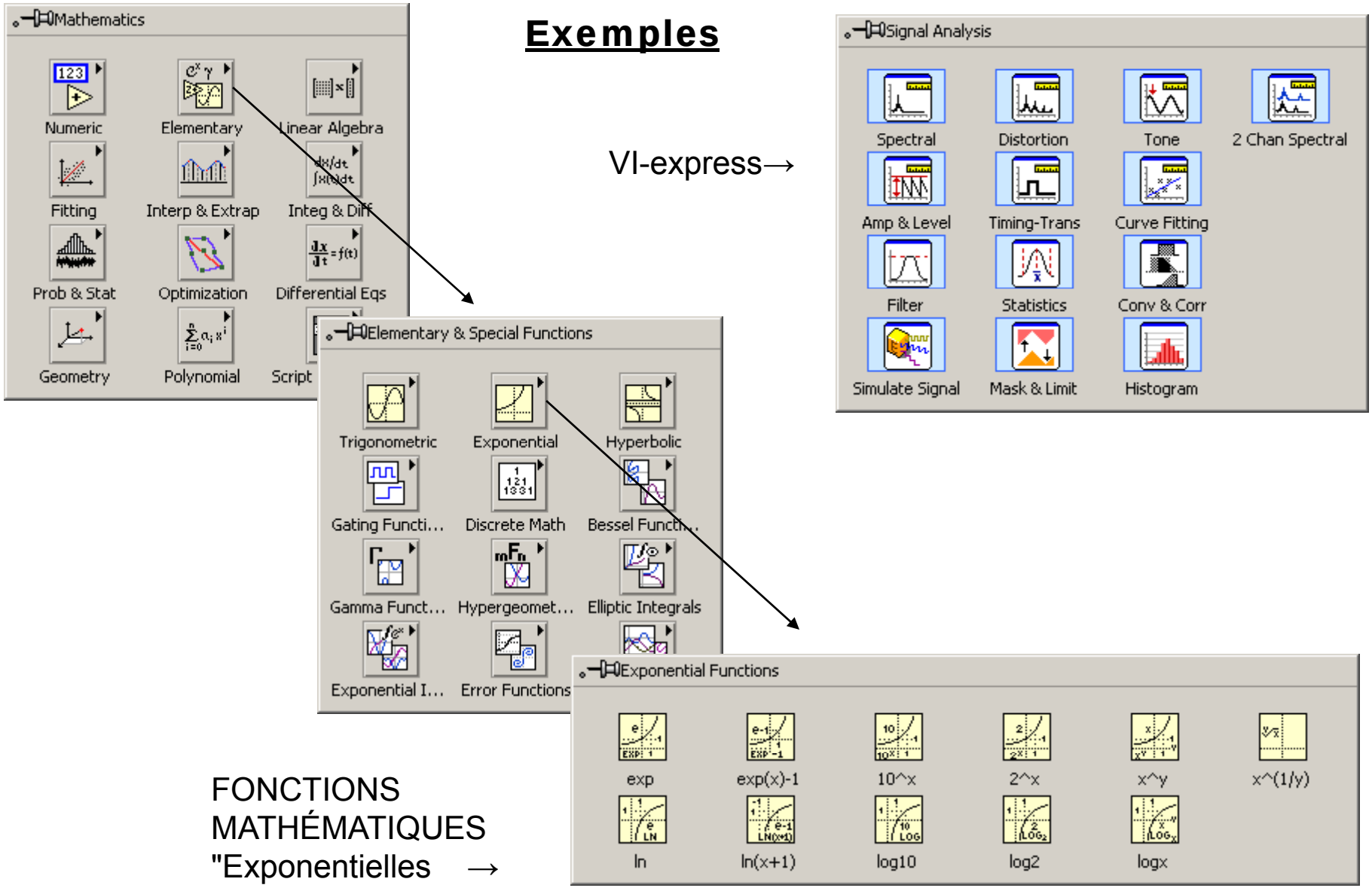


- Les VI-Express
  - Exécutent des tâches courantes
  - Sont configurés à l'aide d'un boîte de dialogue
  - Ne possèdent pas de diagramme mais sont transformables en Sous-VI qui eux sont modifiables



- Tous les noeuds possèdent des terminaux

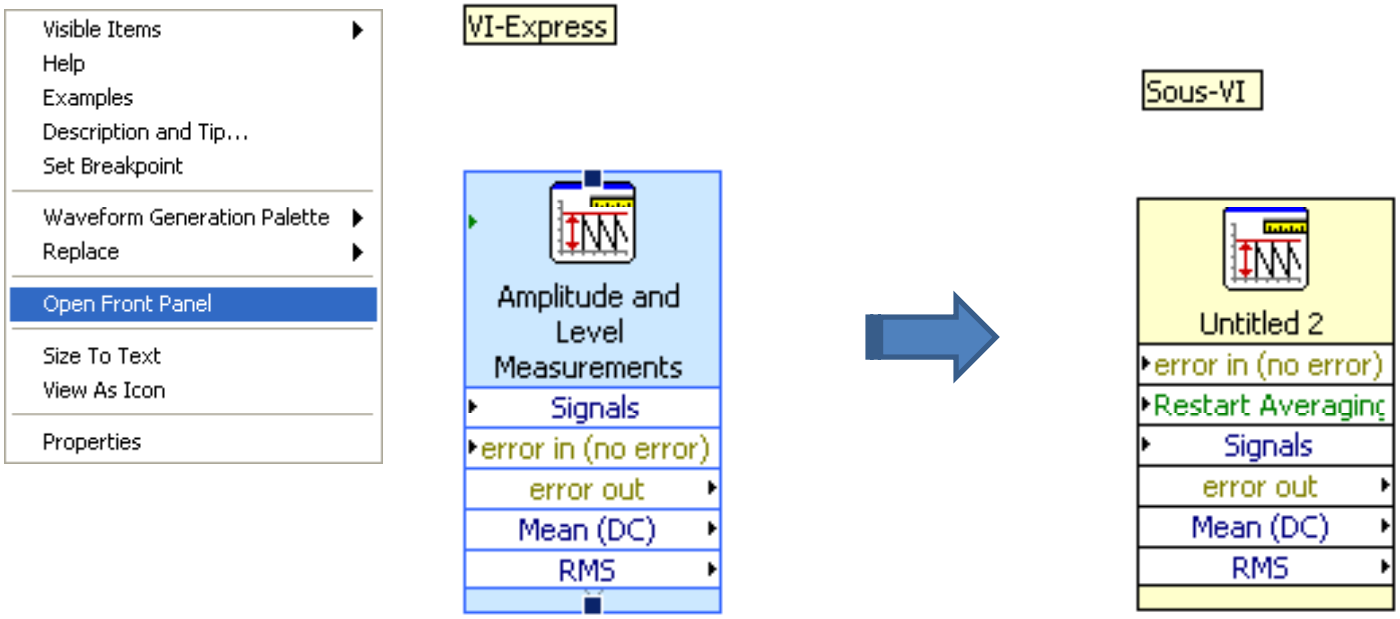
Exemples



FONCTIONS MATHÉMATIQUES "Exponentielles →

Transformer un VI-Express en Sous-VI

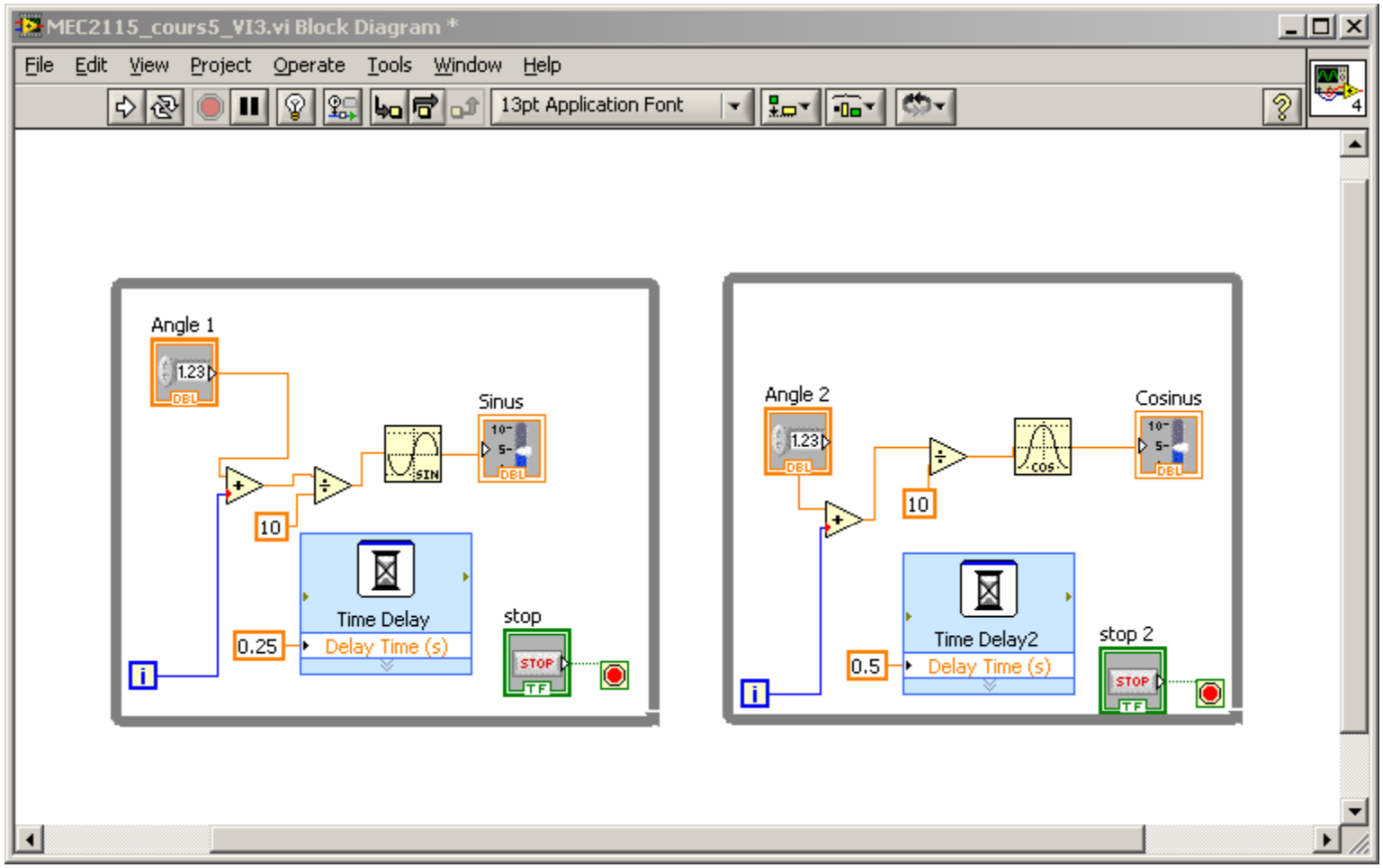
Dans le menu local du VI-Express, cliquer sur "Open Front Panel" pour créer un Sous-VI qui sera sauvegardé par la suite sous un nouveau nom. Le Sous-VI créé peut être modifié selon les besoins de l'utilisateur.



Flux de données dans le diagramme

- Un nœud s'exécute lorsque **toutes les valeurs de ses entrées sont disponibles**
- Lorsqu'un nœud s'exécute, il produit des données de sortie qui sont dirigées vers le nœud suivant via les fils de liaison
- C'est le flux de données qui détermine l'ordre d'exécution des éléments du diagramme.
  - par ex. on peut avoir deux boucles *While* qui s'exécutent simultanément en l'absence de lien de dépendance des données (parallélisme d'exécution, voir diapo suivante)

Exécution de deux boucles WHILE en parallèle



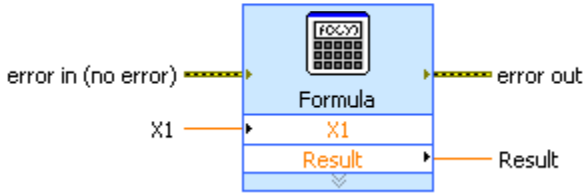
`LV_cours5_VI2_H10.vi`

Cas des nœuds non connectés

- Les nœuds du diagramme qui ne sont pas connectés par des fils au reste du diagramme peuvent s'exécuter dans n'importe quel ordre car il n’y a pas dépendance des données
- Pour ce type de nœud, on ne peut pas présumer que leur ordre d’exécution dans le diagramme se déroule de gauche à droite ou de bas en haut, ni à quel moment elle débute.

Méthodes pour contrôler l'ordre d'exécution

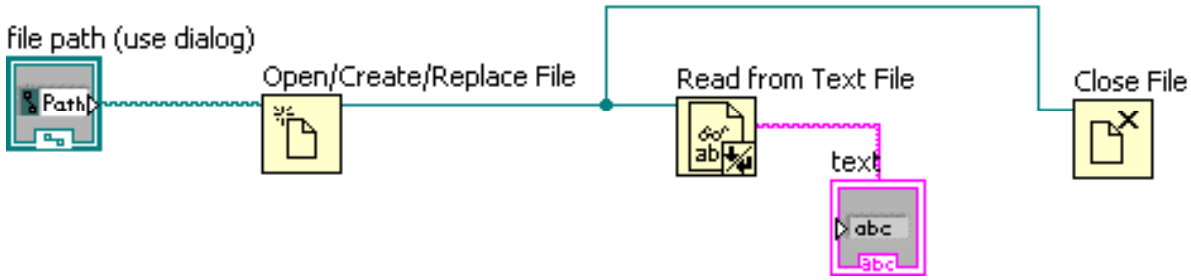
1. Si une dépendance naturelle des données existe
   - Câbler le flux en fonction de l'ordre d'exécution désiré
2. S'il n'y a pas de dépendance naturelle
   - Créer une dépendance "artificielle" avec les paramètres dupliqués. Ces paramètres ont la même valeur à l'entrée qu'à la sortie du nœud (ex. Cluster d'erreur : *error in, error out*)



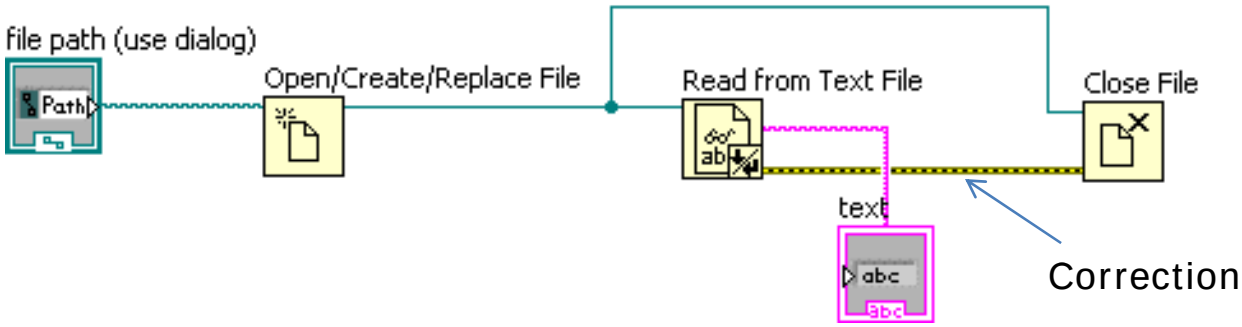
3. Utiliser des structures **Séquence**

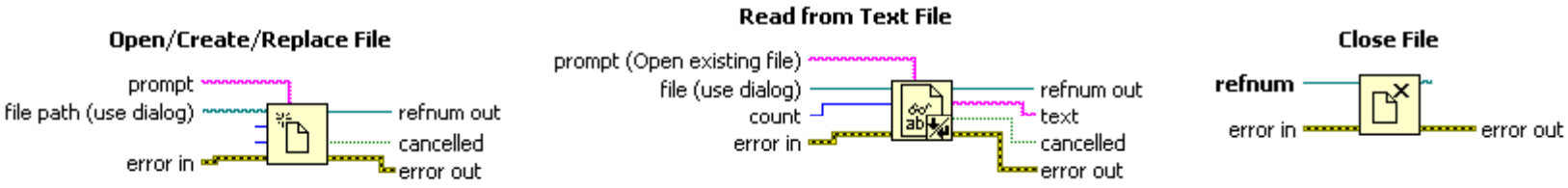
EXEMPLE DE L'EFFET DE LA DÉPENDANCE DES DONNÉÉS:

Ordre d'exécution déficient. La fermeture du fichier peut se produire avant sa lecture à caude de l'absence de dépendance des données.

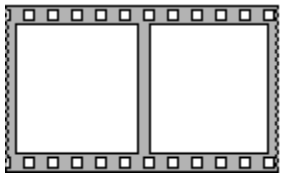


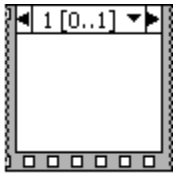
Correction du problème avec un lien de dépendance créé en reliant le terminal *error out* de la fonction *Read from Text File* au terminal *error in* de la fonction *Close File*.





Structures Séquence

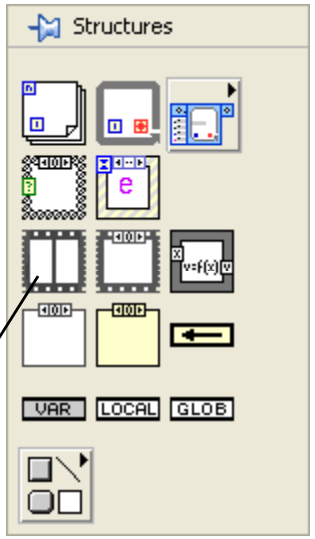


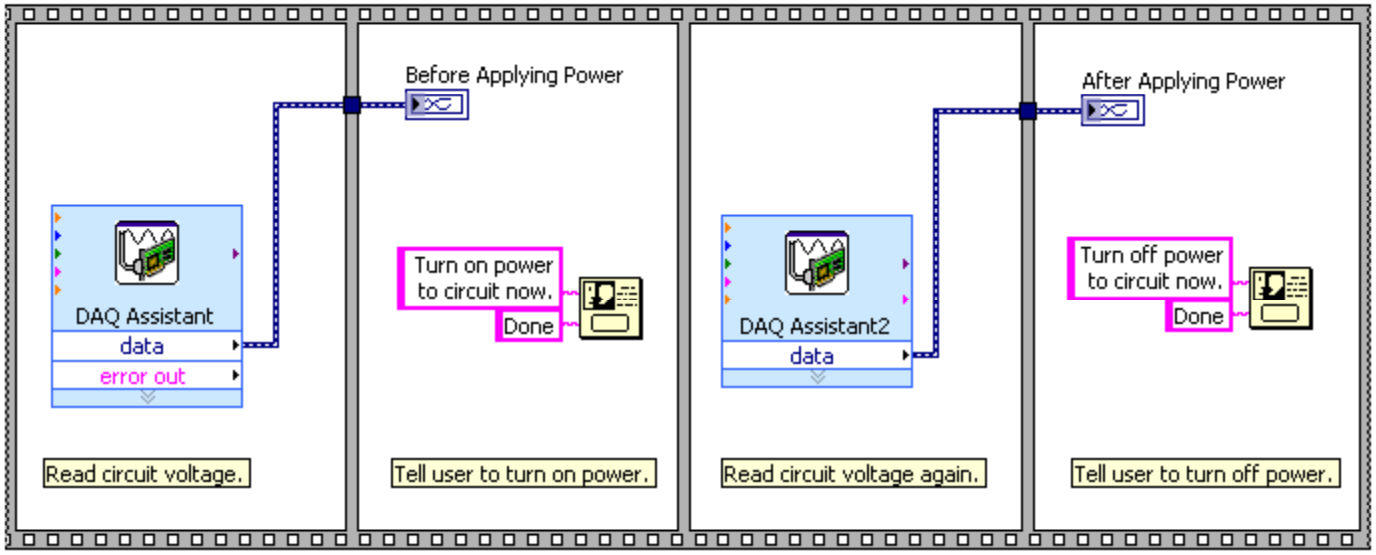


- Contiennent un ou plusieurs sous-diagrammes (étapes) qui s'exécutent dans un ordre séquentiel
- Dans chaque sous-diagramme, comme dans tout autre diagramme, la dépendance des données détermine l'ordre d'exécution des nœuds
- Les structures Séquence garantissent l'ordre d'exécution et interdisent l'exécution en parallèle de leurs différents sous-diagrammes
- Pour profiter du parallélisme d'exécution inhérent à LabVIEW, éviter l'usage excessif des structures Séquence.

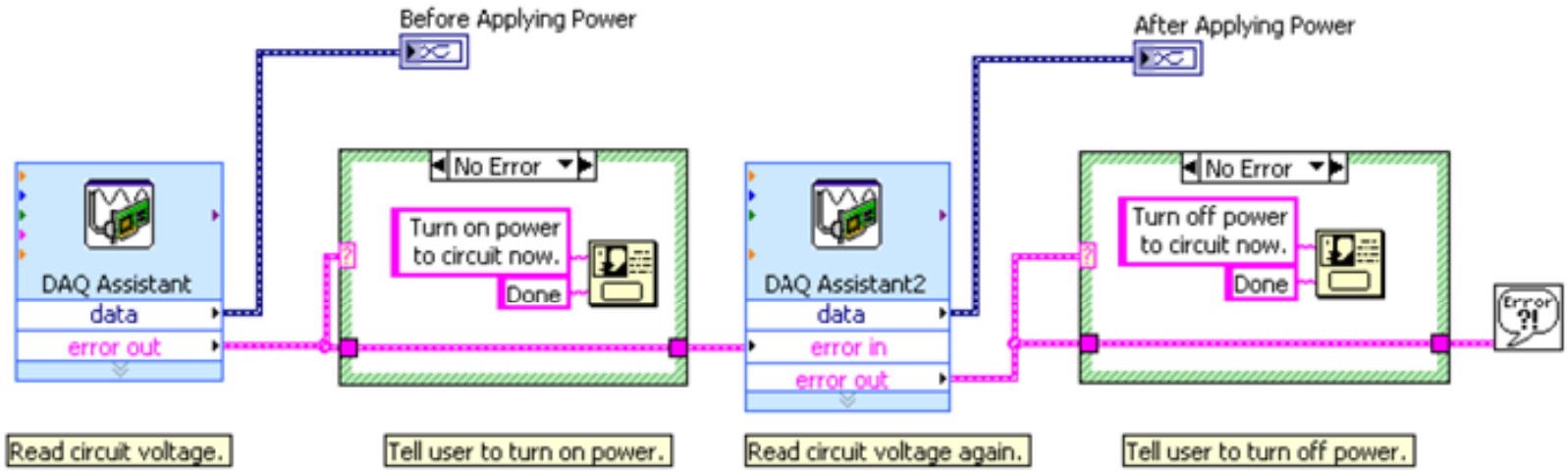
Structure Séquence Déroulée

- Affiche tous les sous-diagrammes (étapes) et les exécute de gauche à droite
- Les valeurs de sortie quittent chaque sous-diagramme lorsque celui-ci finit de s'exécuter





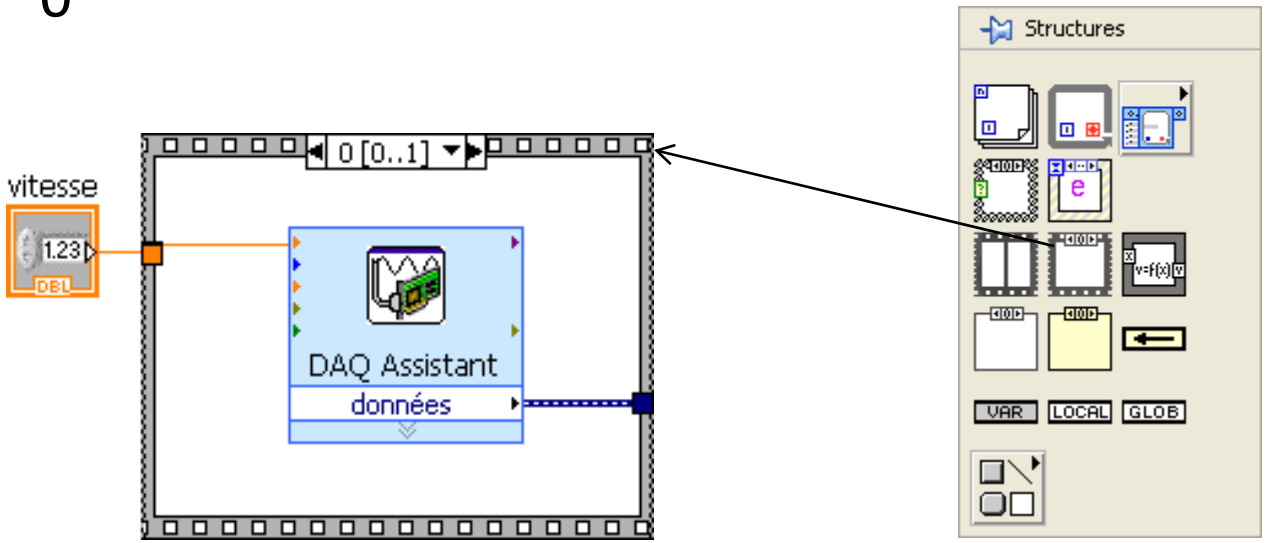
Structure Séquence vs. Contrôle de l'ordre d'exécution par dépendance artificielle des données



Ce diagramme qui contrôle l'ordre d'exécution avec des paramètres dupliqués, s'exécute dans le même ordre que la structure séquence de la page précédente

Structure Séquence empilée

- Empile les sous-diagrammes
- Vous ne pouvez voir qu'un sous-diagramme à la fois
- Exécute les sous-diagrammes dans l'ordre à partir du sous-diagramme "0"



Conseils pour la conception pour les diagrammes

- Présenter le flux de données de gauche à droite et de haut en bas
- Limiter la dimension du diagramme à un ou deux écrans
- Éviter de cacher les fils sous les objets ou les structures
- Documenter le code avec des étiquettes "libres" sur le diagramme

Conseils de conception (suite)

- Diviser le diagramme en Sous-VI qui remplissent des tâches spécifiques, aide à gérer le changement et à mettre au point
- Une planification est nécessaire pour créer une hiérarchie logique des sous-VI d'un programme
- Dans la face-avant des sous-VI, placer les commandes à gauche et les indicateurs à droite
- Transformer des VI existants en sous-VI
- Transformer des **VI-Express** en sous-VI

Entrée/Sortie sur fichiers *(File I/O)*

- Les opérations E/S sur fichiers servent à:
  - La création, l'ouverture et la fermeture des fichiers
  - La lecture et l'écriture de données dans des fichiers
  - Le déplacement des fichiers et des répertoires
  - Le changement de nom de fichier
  - La modification des caractéristiques d'un fichier
  - Etc.

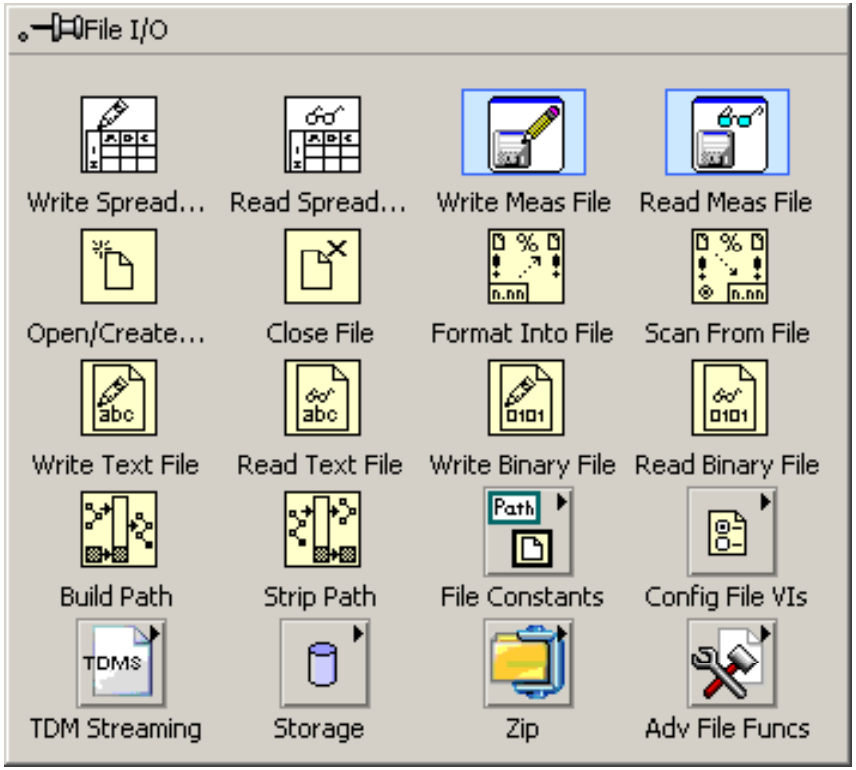
Format des fichiers

- Vous pouvez lire ou écrire des données dans trois formats :
  - texte (`.txt`)
  - binaire (`.tdm`)
  - journal (Datalog) (`.lvm`)

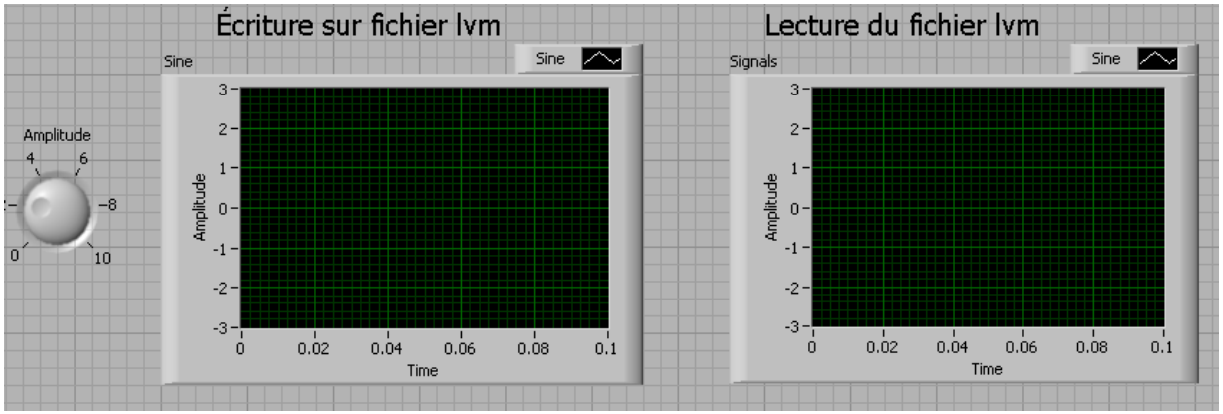
Choix du format des fichiers

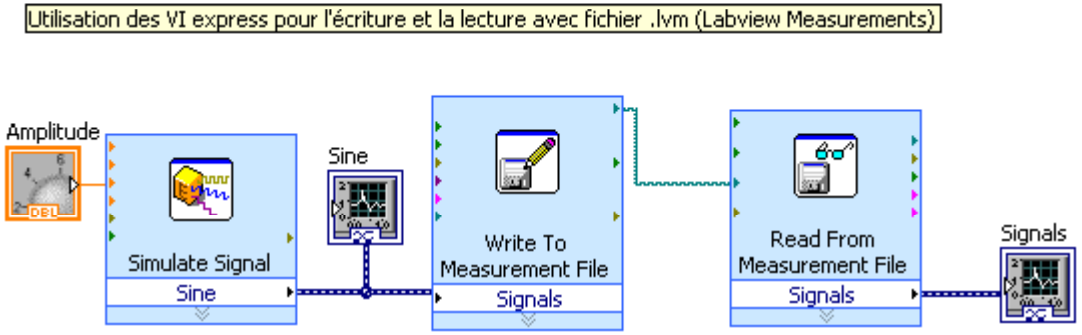
- Pour rendre vos données **accessibles** à d'autres applications(ex. Excel) utilisez un fichier **texte** qui est le plus portable.
- Pour un **accès aléatoire** utilisez un fichier **binaire** qui est plus performant que le fichier texte en termes d'espace disque et de rapidité.
- Pour manipuler des enregistrements de **données complexes** ou des types de données différents dans LabVIEW, utilisez le fichier **journal (`.lvm`)** qui représente le moyen le plus simple de stocker des données de type dynamique (VI-Express) dans un format texte ou binaire.
  - Les fichiers journaux sont spécialement conçus pour être relus avec le VI-Express *Read from Measurement File*. Cependant, on peut aussi relire les fichiers journal en format texte avec d'autres logiciels (ex. Excel).

Fonctions d'E/S *(File I/O)*



Exemple d'utilisation du fichier journal (.lvm) pour enregistrer et relire des signaux





`LV_cours5_lvm_H10.vi`

LabVIEW Measurement
Writer_Version 0.92
Reader_Version 1
Separator Tab
Multi_Headings Yes
X_Columns One
Time_Pref Relative
Operator lumarc
Date 2009/03/13
Time 17:26:00.872031
End_of_Header

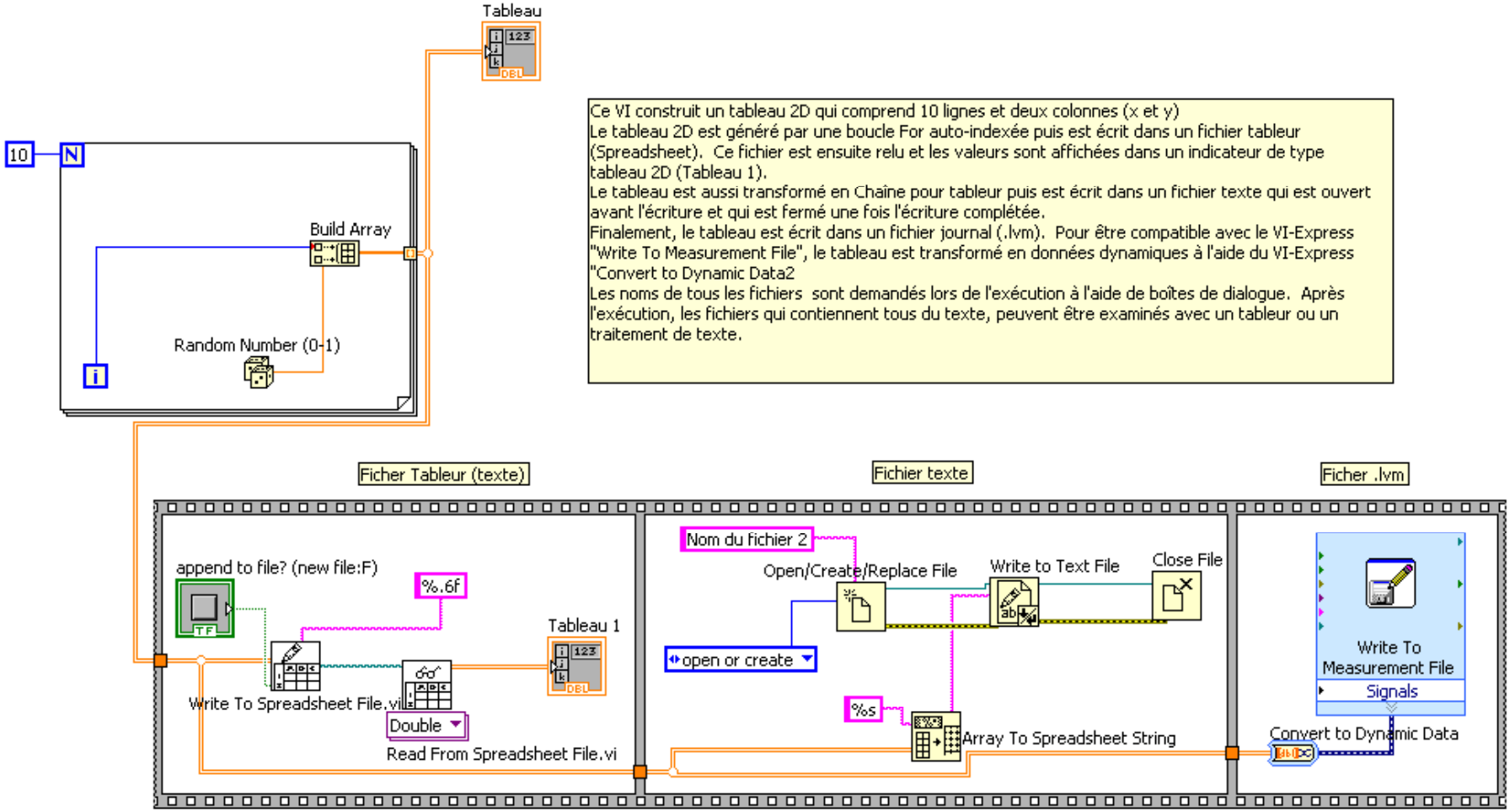
Channels 1
Samples 100
Date 2009/03/13
Time 17:26:00.872031
X_Dimension Time
X0 0.0000000000000000E+0

Delta_X 0.001000
End_of_Header

X_Value	Sine	Comment
0.000000	0.000000	
0.001000	0.138924	
0.002000	0.277289	
0.003000	0.414537	
0.004000	0.550117	
0.005000	0.683482	
0.006000	0.814095	
0.007000	0.941431	
0.008000	1.064977	
0.009000	1.184235	
0.010000	1.298726	
0.011000	1.407988	
0.012000	1.511582	
0.013000	1.609091	
0.014000	1.700122	
0.015000	1.784308	

Exemple d'écriture et de lecture de fichiers texte, tableur (spreadsheet) et journal

- Avec une boucle FOR, créer un tableau 2D de 10 lignes où chaque ligne contient un couple (x,y) de mesures.
- Enregistrer le tableau de mesures dans un fichier tableur, dans un fichier texte et dans un fichier journal (format texte)
- Ouvrir et examiner les trois fichiers obtenus

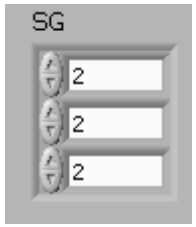


LV_cours5_ES_H10.vi

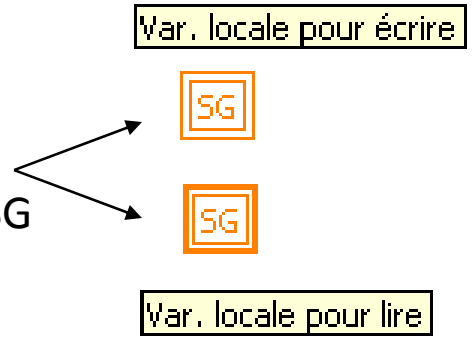
Variable locale (Local Variable)

- Permet l'accès direct aux objets de la face-avant à partir du diagramme (concept avancé de LabVIEW)
- Une variable locale est associée à une commande ou un indicateur
- Avec une variable locale on peut:
  1. Lire et écrire dans une commande
  2. Lire et écrire dans un indicateur

Commande SG
(face-avant)



Exemples de
variables locales
pour la commande SG
dans le diagramme

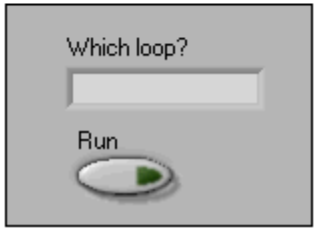


Variable locale (suite)

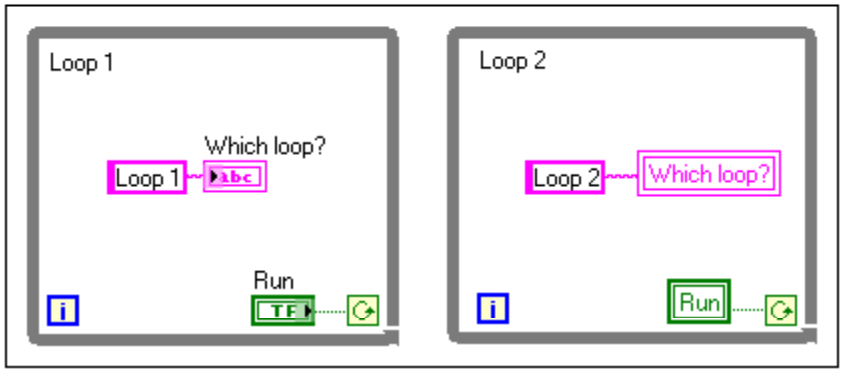
- Dans le diagramme, il n'est pas permis d'écrire dans une commande ou de lire un indicateur. Si on essaie, on génère une erreur. Avec une variable locale on contourne ces restrictions.
- Dans un diagramme, plusieurs variables locales peuvent être associées à la même commande (ou indicateur). Donc on peut accéder à un objet de la face-avant à plusieurs endroits dans le diagramme.
- Les variables locales peuvent aussi servir à transférer des données entre des nœuds du diagramme où il est difficile, voire impossible, de passer un fils.
- Pour plus de renseignements, consulter l'aide de LabVIEW: Sommaire > Fundamentals > Local Variables, Global……

Writing to One Front Panel Object from Two Block Diagram Locations

You can use a local variable to update a single front panel indicator from more than one location on the block diagram. For example, if you have a VI with two While Loops, you can update a front panel indicator to display which loop is currently executing. The following illustration shows the front panel for such a VI.



On the block diagram, place the indicator terminal in Loop 1 and a local variable instance of the indicator in Loop 2, as shown in the following illustration. Notice that this example also uses a local variable instance of the **Run** control to control the execution of both While Loops. You also can use a local variable to allow a single front panel button to control two parallel While Loops in a single VI. A wire does not connect the loops, and they execute simultaneously.



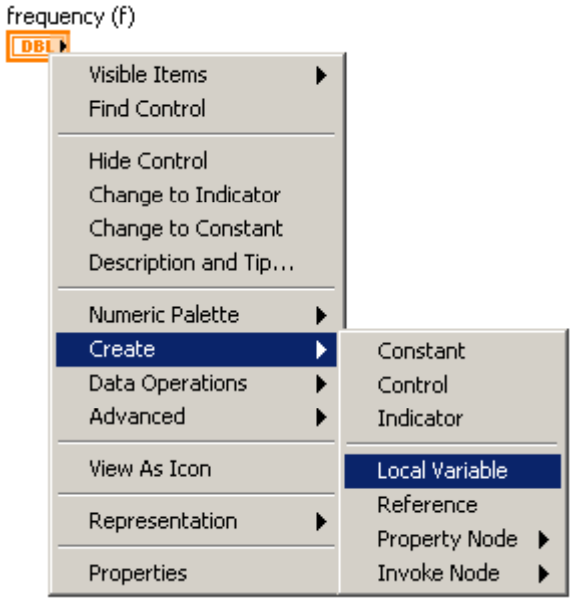
Note You cannot associate a local variable with a Boolean control or indicator that has its mechanical action set to latch when pressed or released.

Submit feedback on this topic

`LV_cours5_Boucles_et_VariablesLocales.vi`

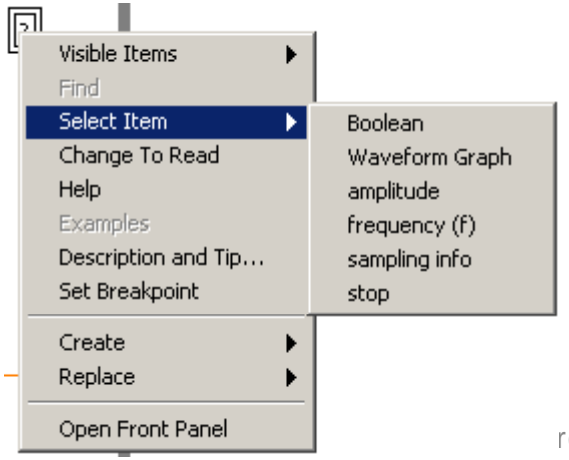
Création d'une variable locale, 1er façon

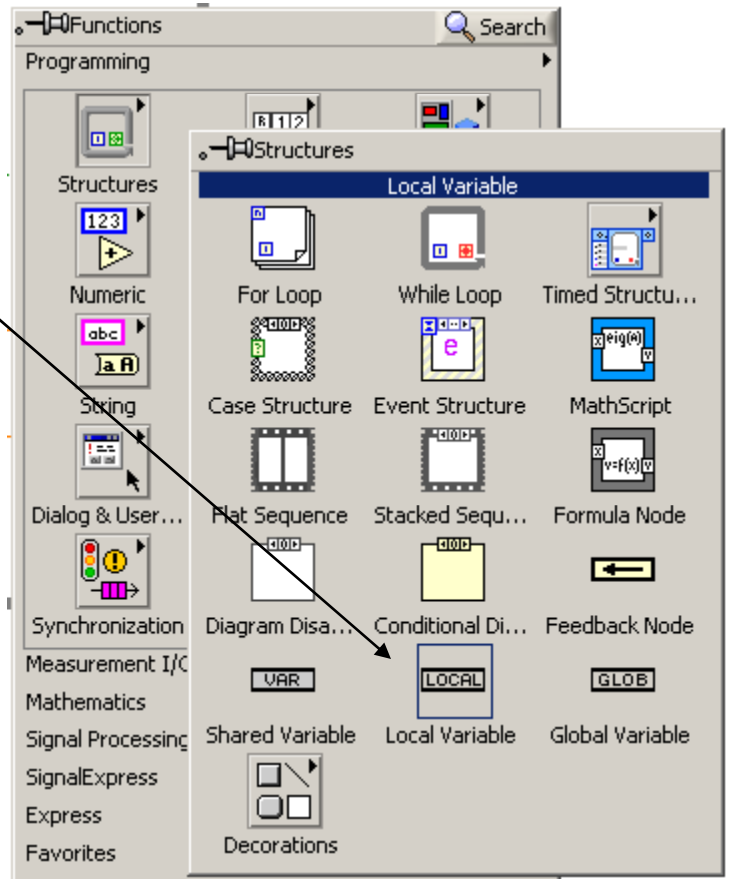
- Dans le diagramme, faire un clic-droit sur la commande ou l'indicateur pour ouvrir son menu local
- Choisir *Create>Local Variable*
- Cliquer droit sur la variable locale crée et modifier le type d'action (*Change to Read* ou *Change to Write*) si nécessaire
- Déplacer la variable locale à l'endroit désiré dans le diagramme



2e façon

- Ouvrir la sous-palette *Functions> Programming > Structures*
- Choisir *Local* et insérer la variable locale dans le diagramme
- Cliquer droit sur la variable locale pour ouvrir son menu:
  - Choisir la commande ou indicateur associé avec *Select Item*
  - Modifier le type d'action (*Change to Read* ou *Change to Write*) si nécessaire





Lectures

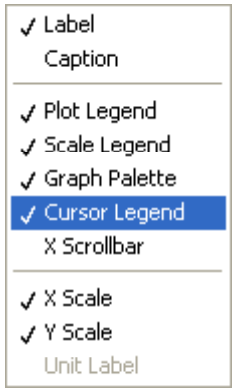
- Flux de données: [2] Chap. 5, p.5-11
- Structure Séquence: [2] Chap. 8, p.8-15
- Création de sous-VI: [2] Chap. 7

Exercice, cours 5
Écrire un VI pour la caractérisation d'un filtre

- Créer un VI qui servira à mesurer l'atténuation et le déphasage d'un filtre passe-bas à différentes fréquences d'excitation (sinusoïde).
- Les données recueillies sont enregistrées dans un fichier qui peut être relu avec Excel afin de compléter l'analyse (diagrammes de Bode et de phase) du filtre.

Exercice du cours 5

1. Récupérer le programme `LV_cours3_struc_cond1_H10.vi` pour le modifier.
2. Avec la fonction `Merge Signals` , combiner le signal original et le signal filtré puis l'envoyer au le graphe
3. VI-Express `Simulate Signal`: ajouter des commandes pour l'amplitude et la fréquence. Ouvrir le VI et enlever la génération de bruit et ajuster les paramètres `Samples per second` (180x la fréquence du signal) et `Number of samples` (180). Voir aussi la dernier point de cette liste et la diapo de configuration plus loin.
4. VI-Express Filter: ajouter une commande pour la fréquence de coupure (`Lower cut-off`) . Ouvrir le VI et choisir un filtre passe-bas (`Lowpass`) Butterworth d'ordre 1.
5. Dans le graphe, sur la face-avant, rendre visible les palettes `Scale Legend`, `Graph Palette` et `Cursor Legend` (menu local, `Visible Items`)



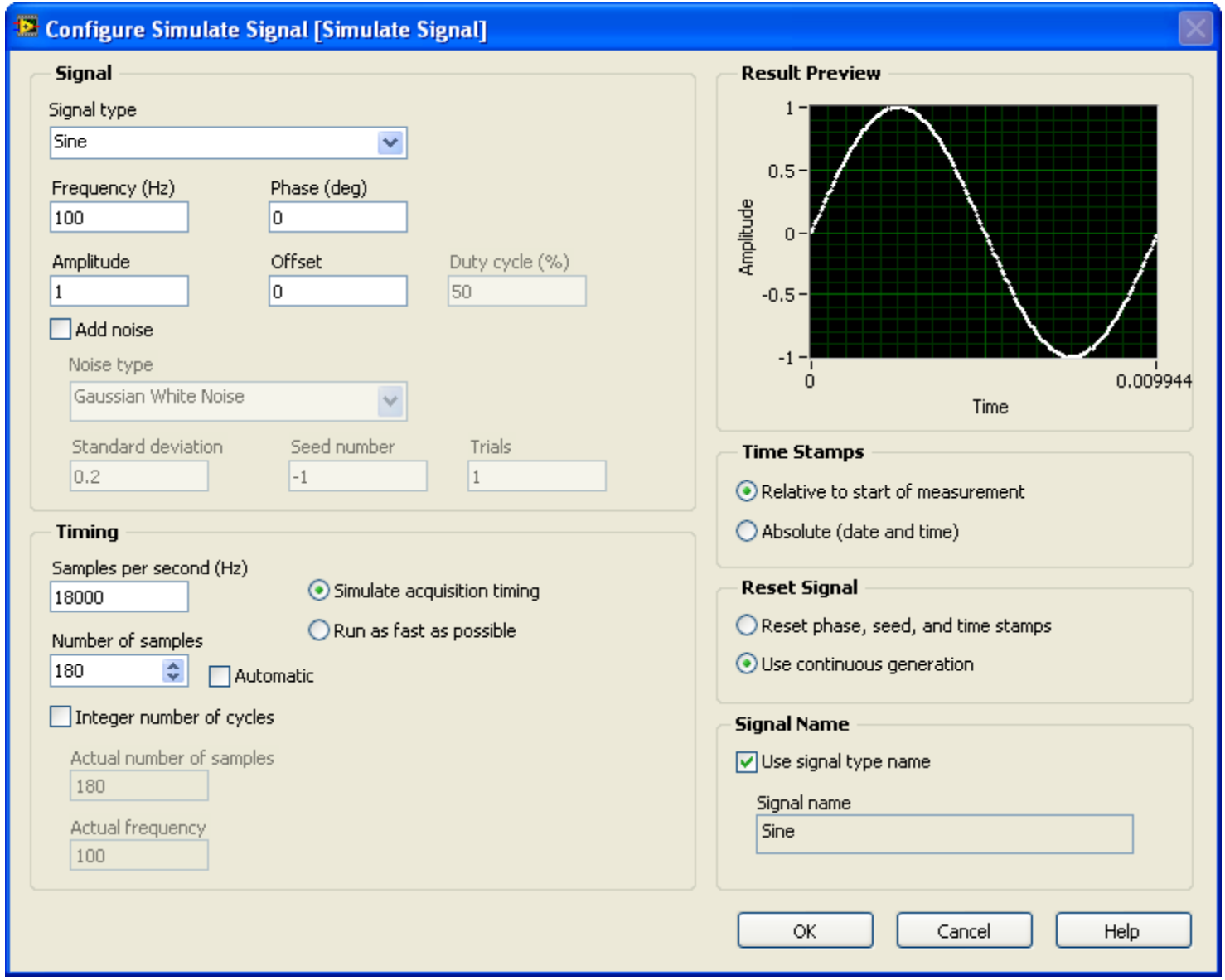
Exercice (suite)

6. Ajouter deux curseurs dans le graphe à l'aide du menu local de la légende des curseurs (`Create Cursor > Single Plot`) et assigner un curseur à chaque courbe avec le menu local du curseur et la commande `Snap to.`
7. Tester le bon fonctionnement du programme avec une fréquence de coupure <= à la fréquence du signal. Déplacer les curseurs et observer la variation des coordonnées des curseurs dans la légende.
8. Dans le diagramme, créer les nœuds de propriétés pour lire les positions X et Y des deux curseurs du graphe (voir diagramme plus loin)
9. À partir de la position des curseurs, implanter dans le diagramme le calcul de l'atténuation (rapport d'amplitudes) et du déphasage (en degré) entre le signal filtré et le signal original.
10. Sur demande de l'utilisateur, enregistrer dans un fichier texte la fréquence du signal, la fréquence de coupure, l'atténuation et le déphasage du signal filtré.

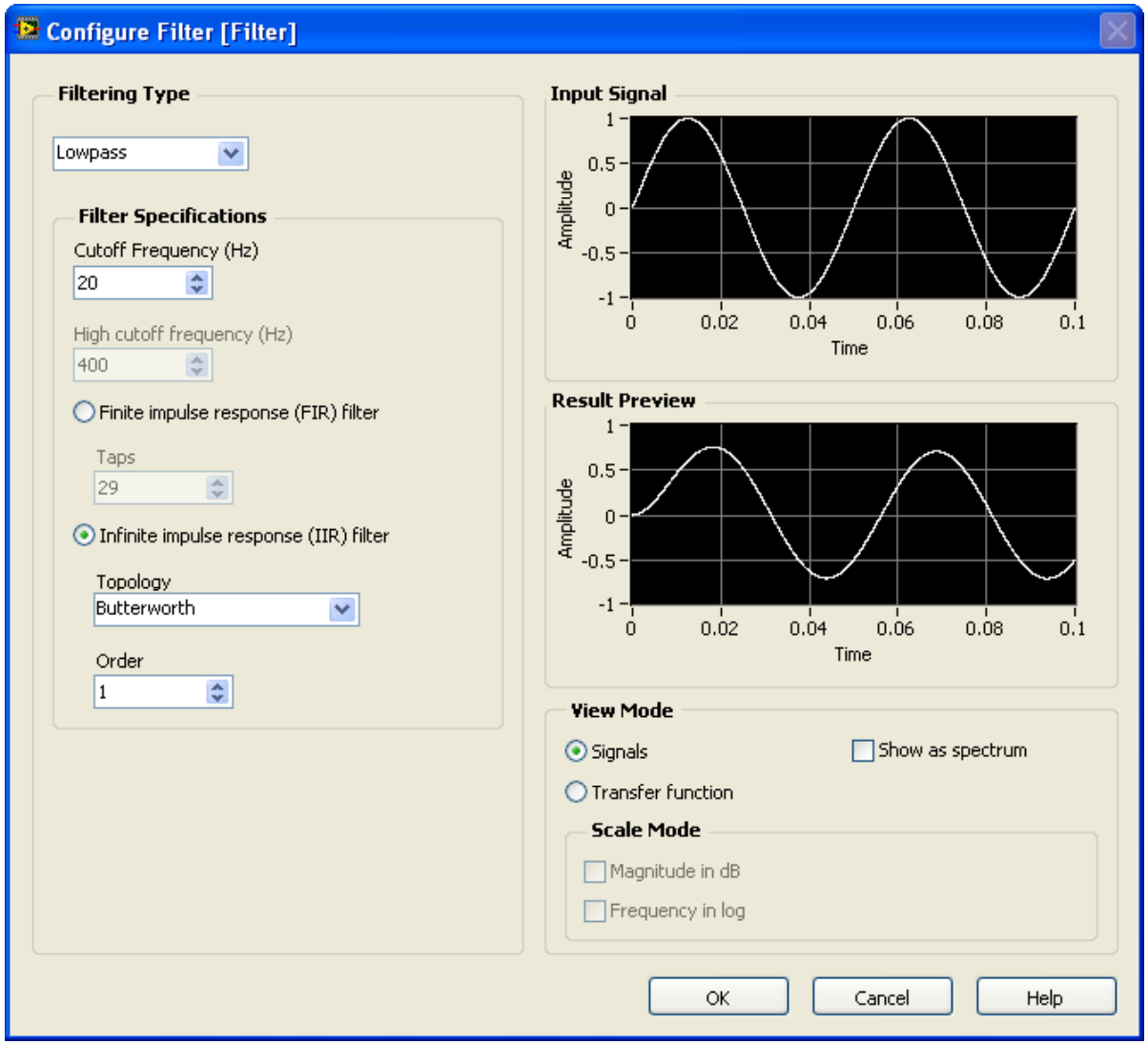
Exercice (suite)

11. Vous devez être capable d'ajouter des données à votre fichier lors d'une nouvelle exécution du programme.
12. Paramètres de configuration du VI Simulate Signal
    - Il faut générer au moins un cycle qui comprend un minimum de 180 points afin d'évaluer le déphasage avec une précision de ±1 degré (on peut avoir plus de points si on veut augmenter la précision). Pour faire cela il faut que le paramètre `Samples per second` soit égal à 180 x fréquence du signal et que le paramètres `Number of samples` soit égal à 180.
    - Pour éviter d'arrêter continuellement le programme afin d'ajuster ces paramètres, vous pouvez transformer le VI-Express en un sous-VI et modifier ensuite son diagramme de façon à ce qu'il calcule lui-même les bons paramètres (voir `Simulate Signals 2.vi` dans le fichier zip du cours no. 5)
    - Attention, ne pas dépasser 880 KHz pour le paramètre `Samples per second` car c'est la limite de la carte d'acquisition de données que vous allez utiliser au laboratoire pour générer le signal.

Configuration du VI-Express Simulate Signal



Configuration du VI-express `Filter`



Exemple de VI pour la caractérisation d'un filtre (présenté en classe)

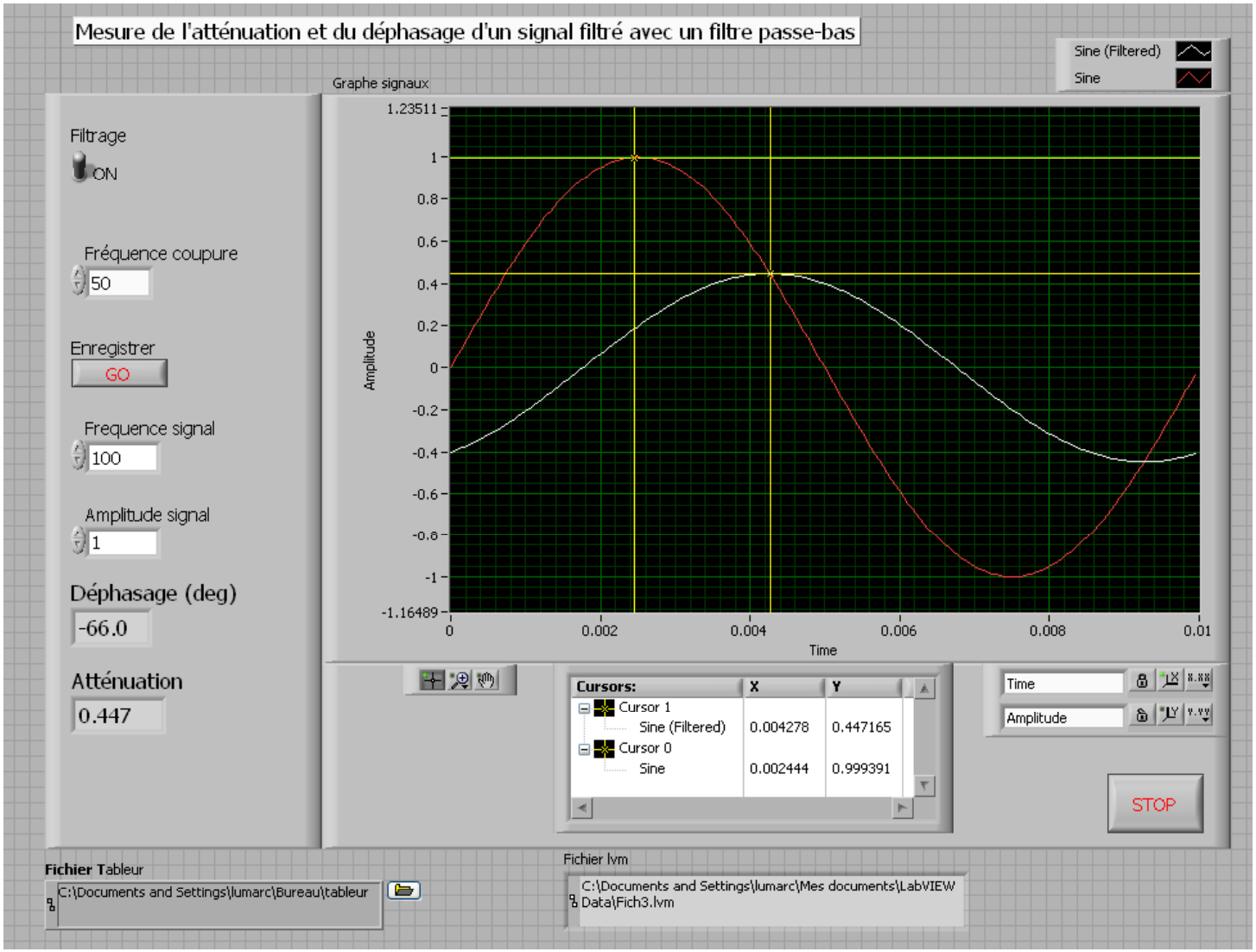
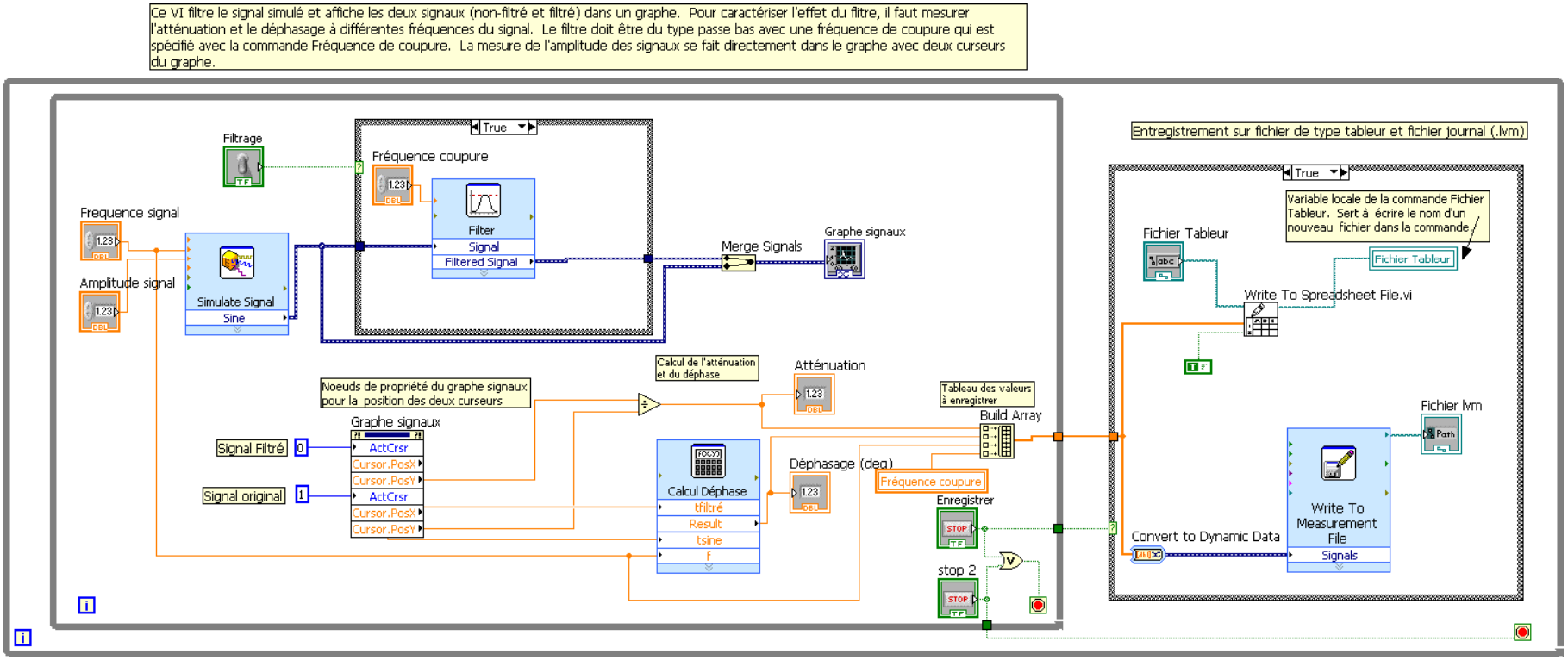


Diagramme du VI pour la caractérisation d'un filtre (face-avant sur diapo précédente)



Configuration du VI-Express Convert to Dynamic Data