Programmation de SYSAM sous Visual Basic Express avec la bibliothèque DLL *ProgrammationSYSAM_SP5.dll*

Introduction

Cet article a pour objectifs

- de présenter sommairement la mise en œuvre de Visual Basic Express 2010 ;
- d'expliquer en détail chacune des commandes de la DLL « *ProgrammationSYSAM_SP5.dll* » ;
- de décrire la réalisation d'un programme exemple simple utilisant ces diverses commandes à l'aide de Visual Basic Express.

Remarques :

- *Le rédacteur de cet article était il y 15 jours novice en Visual Basic. L'idée du projet de création d'outils de programmation de SYSAM SP5 étant de rendre accessible la programmation de SYSAM au plus grand public possible, l'extension de l'usage de la* ProgrammationSYSAM_SP5.dll *au langage Visual Basic était incontournable. Le contenu de cet article concernant Visual Basic sera donc des plus…basiques. Pour les personnes visant un projet élaboré, rien ne vaut internet et un site comme www.developpez.com pour trouver des informations précises et diverses. Si vous débutez en Visual Basic Express 2010, utilisez l'aide du logiciel pour la prise en main.*

- *Tout ce qui suit n'est valable que sous un système d'exploitation WINDOWS, car l'interface SYSAM SP5 n'a pour l'heure pas de pilote pour d'autres systèmes.*

- *On suppose bien sûr que le pilote de l'interface a été installé.*

- *On suppose également que l'interface reste branchée en permanence. L'effet du débranchement au moment de l'appel des fonctions décrites dans cet article n'a pas été testé…*

- *Les captures d'écran de cet article ont été réalisées sous VB Express 2010. J'imagine que si on travaille avec VB Express 2008, ou avec un Visual Basic antérieur, on saura se débrouiller.*

Première partie : utiliser Visual Basic Express 2010.

La version Visual Basic Express 2010 est une version allégée et gratuite de Visual Basic 2010. L'utilisation est gratuite si on s'enregistre. Le site de Microsoft précise : « Outils gratuits, légers, faciles à utiliser et à comprendre pour les développeurs amateurs, novices ou étudiants. Sur la même page revient le mot « gratuit ». Je suppose donc qu'on peut l'installer sur des postes dans un lycée…

1. Téléchargement de VISUAL BASIC EXPRESS 2010

Visual Basic Express 2010 est un élément d'un ensemble d'environnements de développement intégré (EDI) de Microsoft : la suite Visual Studio Express, disponible à l'adresse :

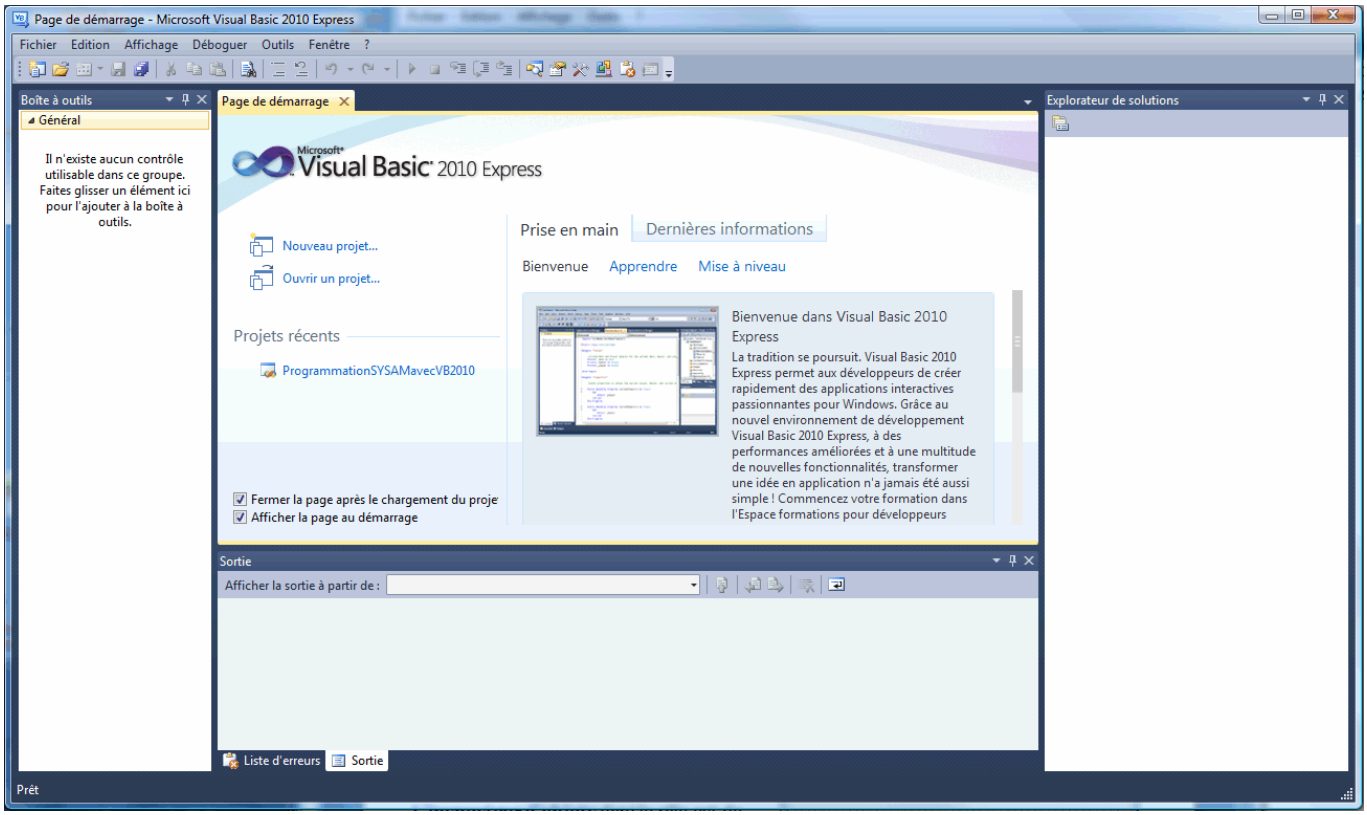
http://msdn.microsoft.com/fr-fr/express/aa975050.aspx

A vous de jouer pour l'installation.

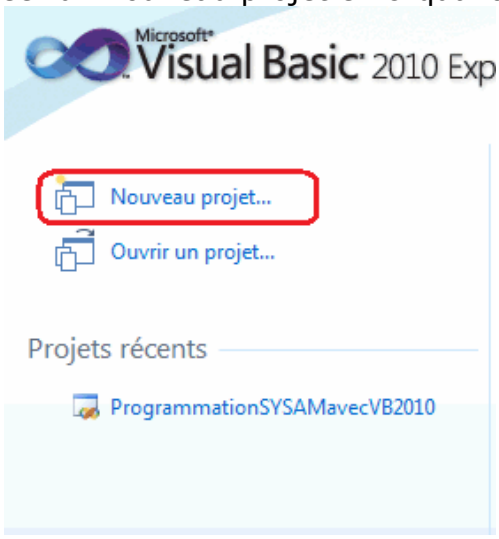
2. Prise en main de Visual Basic Express 2010.

Lorsqu'on veut réaliser un programme, on travaille avec un ensemble de fichiers qui constituent le « projet » et qui doivent être placés dans un dossier unique (un dossier différent pour chaque projet)

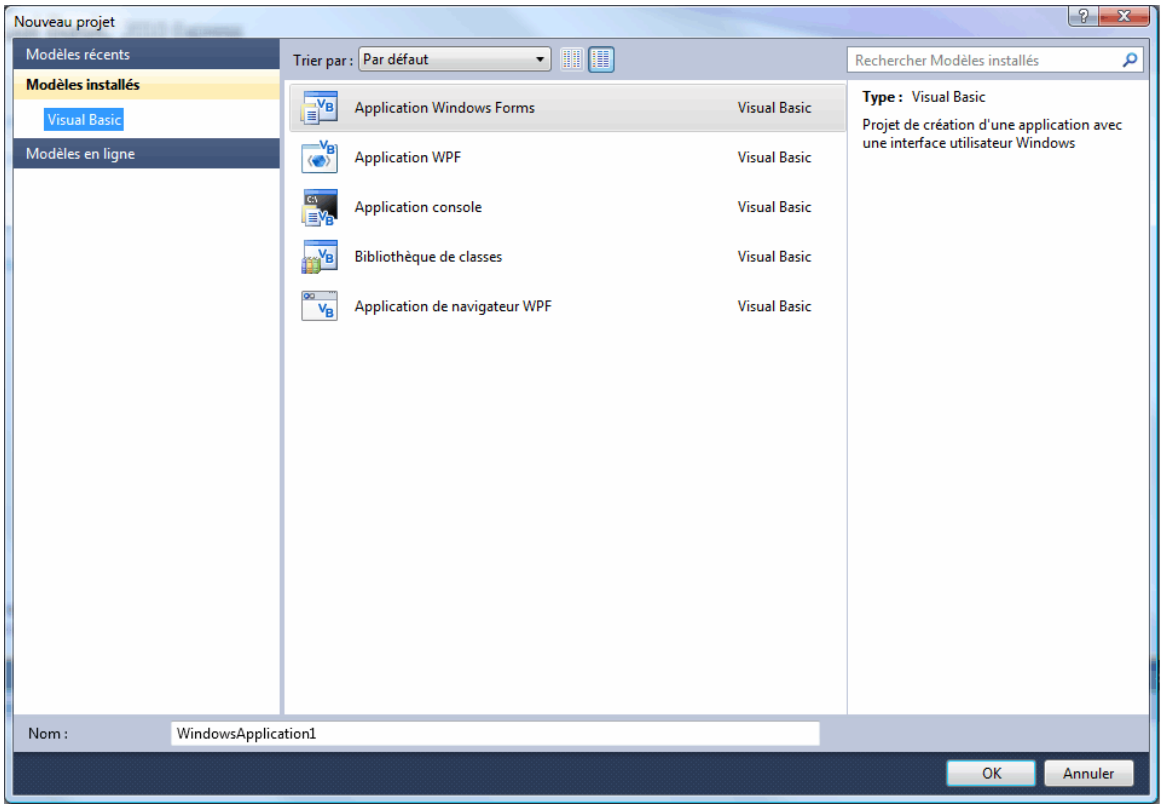
Au lancement on voit apparaître la page de démarrage.



A partir de cette page, créer un nouveau projet en cliquant sur « Nouveau projet »



Dans la boîte de dialogue « Nouveau Projet » sélectionnez la première rubrique : « Application Windows Forms » et cliquez sur OK.

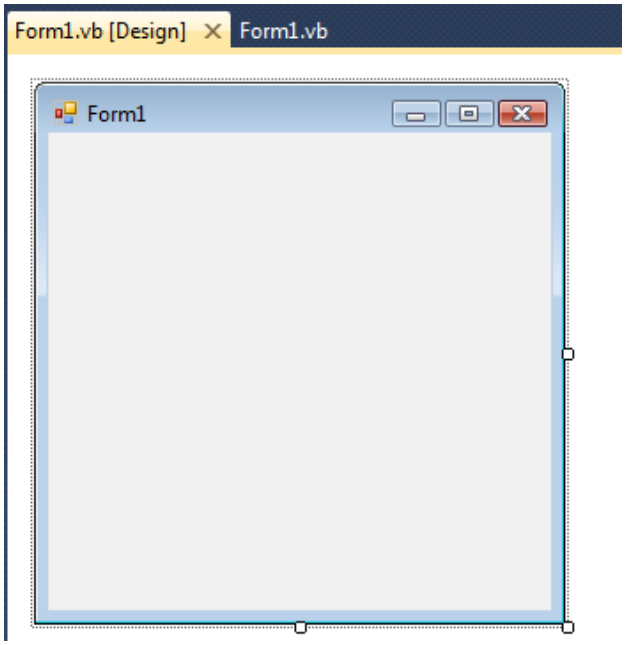


Enregistrez immédiatement le projet : dans la boîte de dialogue, choisissez soigneusement le nom et l'emplacement (utilisez quelque chose de parlant, du genre : Prog_SYSAM_Decouverte).

Les boîtes à outils sont cachées. On peut les faire apparaître en passant la souris, et on peut les fixer en position apparente avec la punaise située à côté de la croix de fermeture.

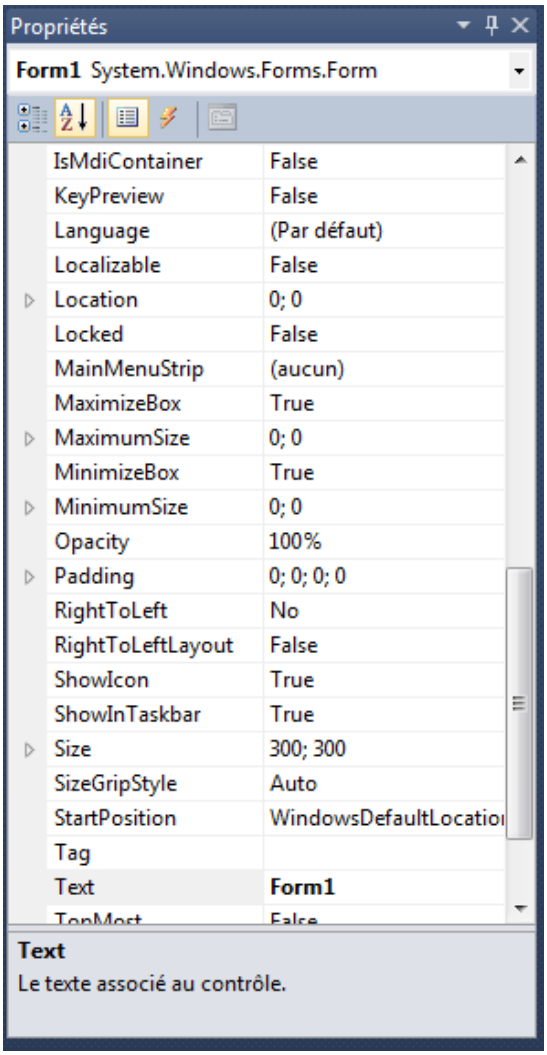
Parmi les éléments de la fenêtre de l'EDI, les suivants sont indispensables

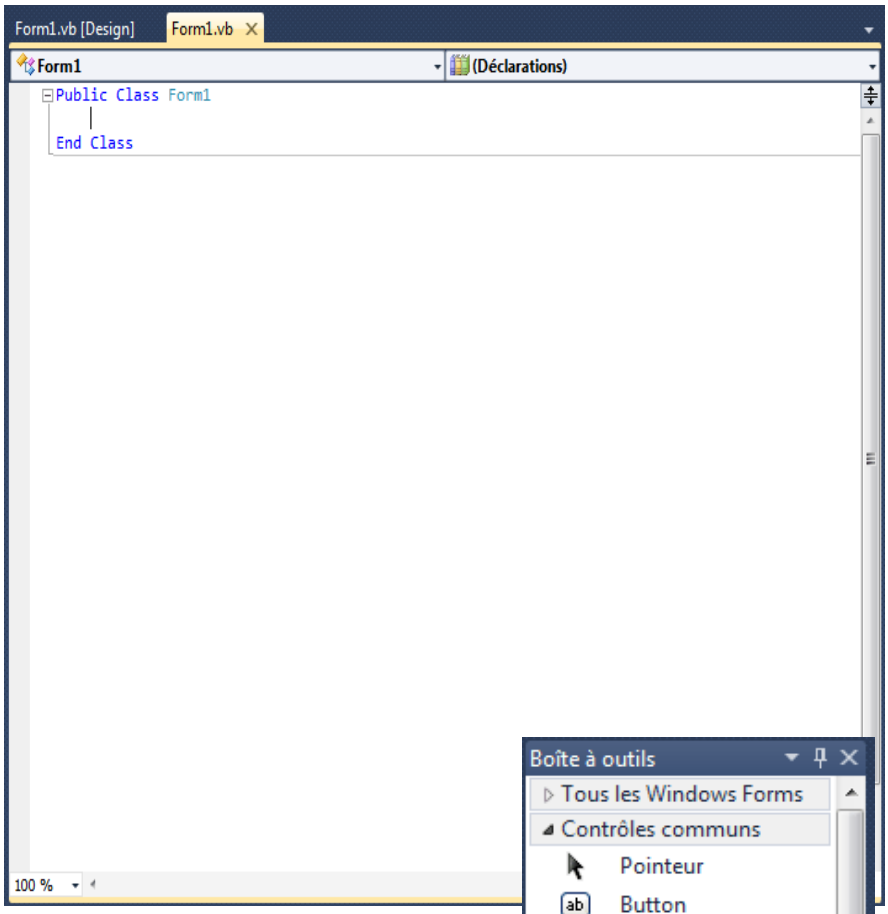
Le concepteur de vues (ou « fiche chantier »), qui est une image de ce que sera la fenêtre du programme, sur laquelle on place des composants « prélevés » dans la **boîte à outils** (palette des composants de la fiche principale). Accès par Maj+F7. Ce concepteur de vues est situé dans un onglet comportant : « Form1.vb[Design] ». Lors de la sauvegarde du projet, les propriétés de la fiche et des composants sont enregistrées dans un des fichiers du projet.

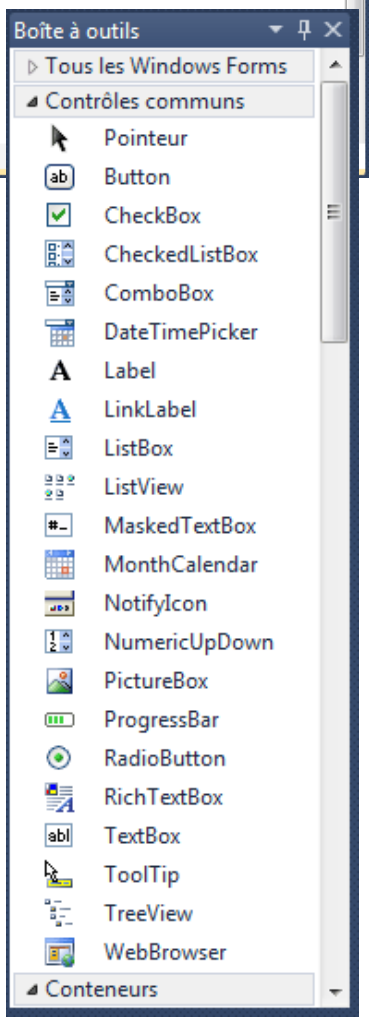


L'éditeur de propriétés (inspecteur d'objets) dont le rôle est de choisir ou régler les propriétés de la fiche et des composants qu'elle contient, au futur démarrage du programme. Pour éditer les propriétés d'un composant, il faut préalablement le sélectionner dans **concepteur de vues.**

L'éditeur de code, où on placera les instructions que devra exécuter le programme. Cet éditeur se trouve dans un second onglet de la même fenêtre que le concepteur de vues.
Lors de la sauvegarde du projet, le contenu de l'éditeur est enregistré dans un des fichiers du projet.







La boîte à outils (palette de composants), dans laquelle on prélève les composants (boutons, mémos, boîtes de saisies, listes déroulantes, menus…) pour les placer sur la fiche chantier pour la réalisation d'un programme original.

3. Touches utiles (il est recommandé de les mémoriser):

F5 : compilation et lancement du programme : Visual Basic Express utilise les informations de l'éditeur de propriétés (inspecteur d'objets) et le code de l'éditeur de code pour créer sur le disque dur un fichier-programme exécutable (.exe) et ensuite lance le fichier-programme. Le fichier exécutable est créé dans un sous-dossier du projet ; par exemple, si le projet se nomme : Programmation_SYSAM_Avec_VB2010, alors le dossier de l'exécutable obtenu est :

C:\...\ Programmation_SYSAM_Avec_VB2010\Programmation_SYSAM_Avec_VB2010\bin\Debug

Ce fichier exécutable contient des informations de débogage (l'exécution du programme peut être suivie instruction après instruction grâce au débogueur intégré)

F7 : affichage en avant-plan de la fenêtre de code (onglet Form1.vb).

MAJ + F7 : affichage en avant-plan du concepteur de vues (onglet Form1.vb[Design])

4. La création d'un programme original.

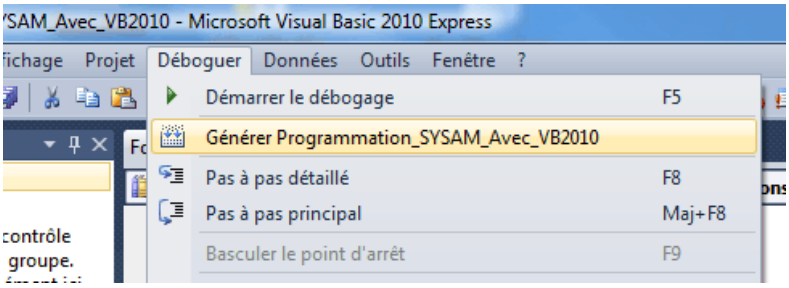
Si vous compilez maintenant le projet (touche F5), vous obtiendrez un programme avec une fenêtre que vous pourrez déplacer, réduire, maximiser et fermer, mais votre logiciel ne fera rien…

Pour créer un programme original, on sélectionne des composants en cliquant avec la souris (Boîte à outils), et on les place sur la fiche chantier (en cliquant sur cette fiche, un composant à la fois). On peut ensuite les « saisir », les placer où on veut sur la fenêtre, et leur donner les dimensions que l'on veut…

Puis on écrit du code (suite d'instructions) qui est exécuté lors d'événements
→ provoqués par le futur utilisateur du programme :
-appuis sur des touches (en rapport avec le composant détenant la focalisation),
-clics avec la souris (en rapport avec les composants sur lesquels on clique)

→ ayant lieu obligatoirement et indépendamment de la volonté de l'utilisateur (celui-ci n'ayant alors rien à faire) pour une bonne gestion de la mémoire et des fichiers :
-démarrage du programme
-fin du programme, pour ne citer que les plus simples.

La quatrième partie de cet article vous guide pas à pas dans toutes ces opérations pour la réalisation d'un exemple.



Lorsque le code est au point, i.e. lorsque le projet se trouve dans un état de finition satisfaisant (si possible sans erreur de programmation), il s'agit de « générer » le véritable fichier exécutable (sans les informations de débogage), dont la taille est minimale, et qui est destiné à être distribué. Pour cela, accéder au menu « Déboguer/Générer… », ce qui a pour effet de créer le fichier « .exe » dans le sous-dossier du projet :

C:\...\ Programmation_SYSAM_Avec_VB2010\Programmation_SYSAM_Avec_VB2010\bin\Release

Deuxième partie : que peut-on faire faire à SYSAM SP5 avec *ProgrammationSYSAM_SP5.dll* ?

Nous appellerons « programme utilisateur » le programme utilisant les fonctions et procédures de la bibliothèque *ProgrammationSYSAM_SP5.dll* pour le contrôle de SYSAM : il s'agit du programme que nous allons créer.

A. Exposé sommaire de la conversion analogique – numérique

Les entrées de SYSAM comportent des convertisseurs analogiques numériques permettant de convertir une tension mesurée (soit directement sur une entrée de SYSAM connectée à un circuit, soit à la sortie d'un capteur) en une valeur numérique entière (abrégée en VNE dans la suite de ce texte) comprise entre 0 et 4095. **La transmission des mesures de SYSAM SP5 vers l'ordinateur et le programme utilisateur se fait exclusivement par des VNE.**

La correspondance entre la valeur numérique et la tension est affine :

$$U = a.\ VNE + b.$$

Les logiciels pilotant SYSAM doivent faire la conversion des valeurs reçues lors de la transmission pour disposer des valeurs de tensions mesurées lors des acquisitions.

Les coefficients a et b dépendent du calibre et de l'entrée utilisés et des réglages précis de SYSAM SP5 en usine. Par exemple, si le calibre est +-1V sur une entrée donnée, on a :

$$-1\ V \ <->\ VNE\ = 0$$
$$+1V \ <->\ VNE\ = 4095$$
$$U \ <->\ VNE1$$

On en tire que

$$U = 2 / 4095 \times VNE1\ - 1$$

et donc

$$a = 2/4095 = 4{,}88.10^{-4} \quad \text{et} \quad b = -1$$

En réalité, les valeurs de a et de b sont légèrement différentes, et la *ProgrammationSYSAM_SP5.dll* utilise les valeurs récupérées dans les données de calibrage de SYSAM (mémorisées à part sur SYSAM au moment du contrôle en usine).

En ce qui concerne les sorties de SYSAM (SA1 et SA2), le processus est symétrique : le programme envoie à SYSAM des VNE judicieusement calculées, qui sont stockées dans sa RAM ; puis SYSAM les convertit en tensions qu'elle fournit à ses entrées, au rythme imposé par le paramétrage temporel des sorties.

B. Les actions possibles

Les instructions de *ProgrammationSYSAM_SP5.dll* permettent de faire faire à la centrale SYSAM SP5 les actions suivantes.

1. Mesure unique sur une de ses entrées analogique. Nous dirons qu'il s'agit d'une **mesure ponctuelle**.

2. Mesure unique presque simultanée sur l'ensemble de ses entrées analogiques actives. Nous dirons qu'il s'agit d'une **acquisition ponctuelle**. De manière générale, nous appellerons « **point** » l'ensemble des valeurs obtenues par une mesure simultanée sur les entrées actives (une valeur par entrée active).

3. Réalisation d'un nombre fini de mesures répétées à intervalle de temps régulier sur l'ensemble des voies actives. Il s'agira d'une **acquisition répétée monocoup**. On obtiendra donc un ensemble limité de points (qui peut être important). L'acquisition cesse dès que le nombre de points donné aura été acquis.

4. Réalisation d'un nombre infini de mesures répétées à intervalle de temps régulier sur l'ensemble des voies actives. Il s'agira d'une **acquisition répétée permanente**. L'acquisition ne cesse que par une action de l'utilisateur du programme pilote. Cette réalisation est plus délicate, car il faut prévoir dans le code le nécessaire pour interrompre la boucle de traitement de l'acquisition.

5. Envoi d'un signal sur chacune des sorties : il s'agit d'une **émission.** On transmet à SYSAM une succession de VNE, qui deviennent par l'intermédiaire du Convertisseur Numérique Analogique (CNA) un signal de tension.

6. Détection des capteurs branchés sur SYSAM, et obtention de leurs numéros d'identifications.

7. Lecture ou paramétrage des états logiques des ports du boitier BOLOGIC.

Pour chacune des actions, on peut effectuer des opérations spécifiques et préliminaires de paramétrage : par exemple pour les acquisitions, activer des entrées au choix, régler un calibre, choisir un type de déclenchement, une tension de seuil, un nombre de points à acquérir ainsi que la durée d'acquisition…

Les instructions de *ProgrammationSYSAM_SP5.dll* permettent de réaliser ces opérations de manière beaucoup plus simple qu'en utilisant les techniques standard données dans la documentation d'Eurosmart, et rendent la programmation de SYSAM accessible aux lycéens et aux étudiants (et aux professeurs !) même novices en programmation.

Pour finir, on notera que les instructions de *ProgrammationSYSAM_SP5.dll* ne permettent pas d'exploiter de manière exhaustive les possibilités de SYSAM SP5. *ProgrammationSYSAM_SP5.dll* a été construite avec l'expérience acquise par la programmation d'OSCILLO5, sans pour autant inclure toute cette expérience. Si un lecteur, après avoir exploité entièrement « *ProgrammationSYSAM_SP5.dll* » souhaite pousser plus avant la programmation de SYSAM, il lui faut contacter l'auteur : contact@logisciences.fr ou bien repartir de la documentation d'Eurosmart.

Troisième partie : le détail du jeu de commandes

A. La bibliothèque « *ProgrammationSYSAM_SP5.dll* ».

Les commandes ou instructions nécessaires au pilotage de SYSAM SP5 sont disponibles dans la bibliothèque « *ProgrammationSYSAM_SP5.dll* ». Il vous faut télécharger cette DLL et la placer

- soit dans le dossier de tout programme distribué qui va l'utiliser, en l'occurrence ici dans les sous-dossiers bin/release et bin/debug du dossier du projet,
- soit dans le dossier C:\Windows\System32

B. Accès aux fonctions.

Pour accéder aux fonctions et procédures, il est nécessaire de déclarer dans le code les fonctions utilisées de la DLL avec le mot réservé Lib. L'emplacement précis sera indiqué dans la quatrième partie de ce document. Le fichier « Déclarations_Visual_Basic.txt » fournit les déclarations qu'il vous suffira simplement de copier dans l'éditeur de code de la fiche Form1 du projet (onglet Form1.vb).

C. Les constantes à déclarer.

Ces constantes dont le but est purement pratique, ne sont pas disponibles dans la *ProgrammationSYSAM_SP5.dll*. Elles doivent être déclarées dans le code du programme utilisateur. Les valeurs qu'elles contiennent sont fixes. Les identificateurs de ces constantes sont en italique gras ; ils peuvent être utilisés tels quels dans le code. Leur utilité est d'obtenir un code plus sympathique et plus lisible : passer à une procédure la constante EA3 au lieu de la valeur 3 rend ce code plus clair. Il y a peut-être d'autres façons de faire, mais mes compétences ne vont pas jusque là. Ces déclarations de constantes sont également données dans le fichier « Déclarations_Visual_Basic.txt ».

Important : dans l'éditeur de VB, il faut laisser l'ensemble de la déclaration sur la même ligne. Cependant, si vous souhaitez la mettre sur deux lignes successives, utilisez le caractère _ isolé par deux espaces comme indicateur de saut de page. Dans le texte qui suit, ce caractère a été omis ; dans le fichier « Déclarations_Visual_Basic.txt », les déclarations sont toutes sur une ligne chacune.

Constantes numériques :

```
'Index des calibres étalonnés de SYSAM
Const cal_10V = 0
Const cal_05V = 1
Const cal_01V = 2
Const cal_002V = 3

'Numéros des voies
Const EA0 = 0
Const EA1 = 1
Const EA2 = 2
Const EA3 = 3
Const EA4 = 4
Const EA5 = 5
Const EA6 = 6
Const EA7 = 7
```

```
'Index des sorties
Const SA1 = 1
Const SA2 = 2

‘Mode des voies
Const mvDiff = True
Const mvNormal = False

'Type de synchro
Const tsAucune = 0
Const tsSeuil = 1
Const tsSynchroExt = 2

'Sens de déclenchement
Const sdMontant = +1
Const sdDescendant = -1

'Entrées/sorties logiques du port B
Const plEntree = 0
Const plSortie = 1
```

Chaînes de caractères :

```
'UTILITAIRES
'Tableau de chaînes de caractères correspondant aux calibres
Dim sCalibresEA() As String = New String(3) {"-10/+10", "-5/+5", "-1/+1",
"0,2/+0,2"}

'Tableau des calibres
Dim CalibresEA() As Double = New Double(3) {10.0, 5.0, 1.0, 0.2}

'Tableau de chaînes de caractères des noms des entrées
Dim NomsEA() As String = New String(7) {"EA0", "EA1", "EA2", "EA3", "EA4", "EA5",
"EA6", "EA7"}
```

Ces trois tableaux sont pratiques pour les boucles de code…

D. Le détail des fonctions

Afin de distinguer les fonctions et procédures de « *ProgrammationSYSAM_SP5.dll* » des procédures et fonctions standard de Visual Basic Express, dans le code du programme utilisateur, tous les noms commencent par « SP5_ »

1. PROCEDURES GENERALES

```
Private Declare Sub SP5_Reinitialisation Lib "ProgrammationSYSAM_SP5.dll" ()
```

Remet l'interface SYSAM SP5 dans son état standard, à savoir:
- arrête les émissions et les acquisitions en cours (s’il y en a)
- remet les calibres des entrées à +/-10V (index 0)
- désactive les voies, etc.

```
Private Declare Sub SP5_FinUtilisation Lib "ProgrammationSYSAM_SP5.dll" ()
```

Libère la mémoire occupée par les structures présentes dans la DLL.
A appeler juste avant la fin du programme utilisateur

```
Private Declare Function SP5_Presente Lib "ProgrammationSYSAM_SP5.dll" () As Boolean
```

Renvoie true si le pilote de SYSAM a été installé et si la centrale est connectée et utilisable. Utile pour sécuriser le programme.

2. TOUT POUR L'ACQUISITION

2.A. Paramètres d'acquisition

```
Private Declare Sub SP5_ActiveVoieA Lib "ProgrammationSYSAM_SP5.dll" (ByVal numVoie As Byte)
```

Active la voie n° numVoie (de EA0 à EA7)

```
Private Declare Sub SP5_DesactiveVoieA Lib "ProgrammationSYSAM_SP5.dll" (ByVal numVoie As Byte)
```

Désactive la voie n° numVoie (de EA0 à EA7)

```
Private Declare Sub SP5_CalibreVoieA Lib "ProgrammationSYSAM_SP5.dll" (ByVal numVoie As Byte, ByVal calibre As Byte)
```

Sélectionne le calibre d'une voie
Paramètres :
- numVoie : numéro de la voie: de EA0 à EA7
- calibre : 0 pour +-10V, 1 pour +-5V, 2 pour +-1V et 3 pour +-0,2V

```
Private Declare Sub SP5_ModeDiff Lib "ProgrammationSYSAM_SP5.dll" (ByVal canal1 As Integer, ByVal mode1 As Boolean)
```

Permet de placer un canal en mode différentiel
Un canal de SYSAM est constitué de deux entrées, par exemple EA0 et EA4 forment le canal 0. Si pour acquérir/mesurer un signal on utilise les deux bornes d'un même canal en mode différentiel, on évite d'utiliser la masse de SYSAM et on mesure la ddp VEA0 - VEA4. Pour que ce soit possible, on active le mode différentiel
Paramètres:
- canal1 : numéro du canal à mettre en mode différentiel (de 0 (pour le canal 0) à 3 (pour le canal 3))
- mode1 : true pour activer le mode différentiel, false pour le désactiver.

```
Private Declare Function SP5_nbVoiesActives Lib "ProgrammationSYSAM_SP5.dll" () As Integer
```

Permet de connaître à tout moment le nombre de voies activées. La valeur renvoyée peut faciliter l'exploitation des acquisitions.

2.B. Acquisition d'un point sur une voie

```
Private Declare Function SP5_AcquisitionVoieUnique Lib "ProgrammationSYSAM_SP5.dll"
(ByVal voie1 As Integer, ByRef mesure1 As UShort) As Integer
```

Cette procédure provoque la mesure d'un point sur la seule voie1 et renvoie la valeur dans le paramètre mesure1 sous forme de valeur entière comprise entre 0 et 4095 (12 bits)

2.C. Acquisition d'un point en simultané sur toutes les voies actives

```
Private Declare Sub SP5_Acquisition Lib "ProgrammationSYSAM_SP5.dll" (ByRef M1 As
UShort)
```

Cette procédure provoque la mesure d'un point sur toutes les voies activées et renvoie les valeurs obtenues dans un tableau (nommé par exemple TabMesures), dont le premier élément (TabMesures(0)) est passé comme paramètre M1 de la procédure.

2.D.a. Préparation des acquisitions de points multiples

```
Private Declare Sub SP5_DureesEtPoints Lib "ProgrammationSYSAM_SP5.dll" (ByVal dur-
eeTotale1 As Double, ByVal nbPts1 As Long)
```

Cette procédure permet de choisir la durée totale de l'acquisition et le nombre de points à acquérir. Cependant, le choix réalisé peut être légèrement modifié par cette procédure de manière à tenir compte du fait que le temps d'échantillonnage (temps séparant la mesure de deux points successifs) doit être un multiple

- de 100 ns si aucune des voies EA4, EA5, EA6, EA7 n'est activée en mode non différentiel
- de 2 µs si une des voies EA4, EA5, EA6, EA7 est activée en mode non différentiel

La modification éventuelle est toujours réalisée dans le sens d'une légère augmentation et n'a généralement lieu que pour les acquisitions très rapides.

Pour connaître les valeurs effectives de la durée totale et du nombre de points, utiliser les fonctions ci-dessous au §2.D.b.

```
Private Declare Sub SP5_ModeDeclenchement Lib "ProgrammationSYSAM_SP5.dll" (ByVal
modeDecl1 As Integer)
```

Cette procédure permet de sélectionner le mode de déclenchement :

Valeur tsAucune = 0 de modeDecl1 : l'acquisition commence au moment où la procédure ***SP5_DeclencheAcqMonoCoup*** ou ***SP5_DeclencheAcqPermanente*** est appelée

Valeur tsSeuil = 1 de modeDecl1 : synchro sur seuil
Les procédures ***SP5_DeclencheAcqMonoCoup*** ou ***SP5_DeclencheAcqPermanente*** déclenchent l'acquisition, mais SYSAM SP5 ne commence à écrire les valeurs acquises dans sa RAM que lorsque le signal passe par une valeur de seuil donnée (réglable avec SP5_SeuilDeclenchement) dans un sens donné (réglable avec une SP5_SensDeclSeuil)

Valeur tsSynchroExt = 2 de modeDecl1 : synchro externe
Les procédures ***SP5_DeclencheAcqMonoCoup*** ou ***SP5_DeclencheAcqPermanente*** déclenchent l'acquisition, mais SYSAM SP5 ne commence à écrire les valeurs acquises dans sa RAM que

lorsque le signal injecté sur l'entrée spéciale SYNCCHRO EXT. de SYSAM SP5 passe par une valeur proche de 1,6V dans le sens réglable par ***SP5_SensDeclEchelon*** (ci-dessous).

```
Private Declare Sub SP5_VoieSynchro Lib "ProgrammationSYSAM_SP5.dll" (ByVal voie1 As Integer)
```

Cette procédure permet de choisir l'entrée du signal de référence de la synchronisation sur seuil. L'entrée choisie doit avoir été activée AVANT ce choix...

```
Private Declare Sub SP5_Pretrig Lib "ProgrammationSYSAM_SP5.dll" (ByVal idxPretrig1 As Integer)
```

Cette procédure sert à la gestion de la pré-acquisition. Elle règle la durée de la pré-acquisition en % de la durée totale. Lorsqu'il y a pré-acquisition, SYSAM mémorise dans sa RAM les valeurs acquises dès le déclenchement par ***SP5_DeclencheAcqMonoCoup*** ou ***SP5_DeclencheAcqPermanente***, tout en attendant au minimum idxPretig1 % de la durée totale avant de commencer à tester les conditions de déclenchement.
Lorsque ces conditions sont réunies (soit t0 cet instant), elle efface les valeurs précédent l'instant t1 = t0 - idxPretrig /100 * dureeTotale et commence à transmettre les valeurs acquises à partir de t1, qui sont récupérées par le programme utilisateur à l'aide de la fonction ***SP5_LireTamponAcq*** décrite ci-dessous.

```
Private Declare Sub SP5_SensDeclSeuil Lib "ProgrammationSYSAM_SP5.dll" (ByVal sens-Decl1 As Integer)
```

Cette procédure sert à fixer le sens de passage par la valeur de seuil qui va provoquer le déclenchement de l'acquisition (en mode de synchronisation sur seuil uniquement)

Valeur sensDecl1 = sdMontant = + 1 : sens montant
Valeur sensDecl1 = sdDescendant = - 1 : sens descendant.

```
Private Declare Sub SP5_SensDeclEchelon Lib "ProgrammationSYSAM_SP5.dll" (ByVal sensDecl1 As Integer)
```

Cette procédure sert à fixer le sens de passage par 1,6 V du signal injecté sur l'entrée SYNCHRO EXT qui va provoquer le déclenchement de l'acquisition (en mode de synchronisation SYNCHRO EXT.)

Valeur sensDecl1 = sdMontant = + 1 : sens montant
Valeur sensDecl1 = sdDescendant = - 1 : sens descendant.

```
Private Declare Sub SP5_SeuilDeclenchement Lib "ProgrammationSYSAM_SP5.dll" (ByVal UseuilDecl1 As Single)
```

Cette procédure sert à fixer la valeur de seuil de déclenchement de l'acquisition (en mode de synchronisation sur seuil.)

2.D.b. Paramètres effectifs d'acquisition tenant compte des performances de SYSAM

```
Private Declare Function SP5_DtAcq Lib "ProgrammationSYSAM_SP5.dll" () As Double
```

Cette fonction renvoie l'intervalle de temps (Dt = delta t) entre deux points successifs. Cet intervalle est déterminé par la procédure SP5_DureesEtPoints décrite ci-dessus, comme le rapport dureeTotale/ nbPts et est forcément (pour des raisons techniques) un multiple du temps d'échantillonnage de base de SYSAM, à savoir :

- de 100 ns si aucune des voies EA4, EA5, EA6, EA7 n'est activée en mode non différentiel
- de 2 µs si une des voies EA4, EA5, EA6, EA7 est activée en mode non différentiel.

```
Private Declare Function SP5_DureeTotaleAcq Lib "ProgrammationSYSAM_SP5.dll" () As Double
```

Cette fonction renvoie la durée totale effective de l'acquisition, qui peut parfois différer légèrement par excès de la durée totale souhaitée et passée en paramètre de la procédure ***SP5_DureesEtPoints***.

```
Private Declare Function SP5_NbPtsAcq Lib "ProgrammationSYSAM_SP5.dll" () As Integer
```

Cette fonction renvoie le nombre de points effectif de l'acquisition, qui peut parfois différer légèrement par excès du nombre de points souhaité et passé en paramètre de la procédure ***SP5_DureesEtPoints***.

2.D.c. Gestion de la récupération des valeurs acquises

```
Private Declare Sub SP5_NbPtsDansTamponAcq Lib "ProgrammationSYSAM_SP5.dll" (ByVal nbpts1 As Integer)
```

Fixe le nombre de points dans le tampon d'acquisition, ainsi que la taille de ce tampon d'acquisition. Le contenu du tampon ne peut pas dépasser 256 valeurs (à partager entre les voies actives). Le nombre de valeurs dans le tampon est normalement égal au produit du nombre de points dans le tampon multiplié par le nombre de voies actives (une valeur par voie active et par point).

Cette procédure ne doit être appelée qu'après l'activation/désactivation des voies et avant l'acquisition.

Le nombre de points passé en paramètre n'est pas forcément celui qui est retenu. Pour obtenir la valeur retenue, utiliser la fonction suivante. Il est indispensable d'utiliser cette valeur dans l'exploitation des acquisitions.

```
Private Declare Function SP5_NbPtsTampon Lib "ProgrammationSYSAM_SP5.dll" () As Integer
```

Renvoie le nombre de points dans le tampon effectivement retenu après l'appel de la procédure précédente. Cette façon de procéder permet de sécuriser le processus et d'éviter de dépasser les limites du tableau de récupération du tampon. Le nombre de points doit être inférieur ou égal à 256 divisé par le nombre de voies actives SP5_nbVoiesActives.

```
Private Declare Function SP5_LireTamponAcq Lib "ProgrammationSYSAM_SP5.dll" (ByRef M1 As UShort) As Integer
```

Une fois l'acquisition lancée, SYSAM effectue les mesures au rythme régulier imposé et les stocke dans sa propre RAM, de manière indépendante du programme utilisateur. Il s'agit pour le programme utilisateur de récupérer ces valeurs, ce qu'on fait par paquets de points dont le nombre a été fixé par la procédure ***SP5_NbPtsDansTamponAcq(nbPts1: Integer).***

Appelons n le nombre de points dans le tampon.
L'appel de la fonction ***SP5_LireTamponAcq*** provoque la lecture des valeurs des n points dans la RAM et leur placement dans un tableau (nommé par exemple TamponSP5) dont le premier élément (TamponSP5(0)) doit être passé en argument M1 de cette fonction.

La valeur renvoyée par la fonction reste nulle tant que SYSAM n'a pas acquis le nombre de points n nécessaire.
Lorsque le nombre suffisant n de points a été acquis, la fonction retourne une valeur égale au nombre de valeurs présentes dans le tableau. Le programme peut alors accéder à ces valeurs et les traiter (remplir un tableau, tracer une courbe...)
Ensuite, on procède de même pour récupérer les n points suivants, et ainsi de suite jusqu'à ce qu'on ait récupéré toutes les valeurs de la RAM, acquises par SYSAM pendant l'acquisition en cours. La fonction ***SP5_LireTamponAcq*** est donc généralement présente dans une boucle de code.

2.E. Déclenchement de l'acquisition répétée.

```
Private Declare Sub SP5_DeclencheAcqMonoCoup Lib "ProgrammationSYSAM_SP5.dll" ()
```

Prépare physiquement l'interface en tenant compte de tous les choix faits précédemment, puis lance l'acquisition d'une trame (durée finie choisie ci-dessus).

```
Private Declare Sub SP5_DeclencheAcqPermanente Lib "ProgrammationSYSAM_SP5.dll" ()
```

Prépare physiquement l'interface en tenant compte de tous les choix faits précédemment, puis lance une acquisition sans fin (durée infinie).

```
Private Declare Sub SP5_ArreteAcq Lib "ProgrammationSYSAM_SP5.dll" ()
```

Arrête les acquisitions multiples (de durée finie « monocoup » ou permanente)

2.F. Fonctions de conversion pour les entrées

```
Private Declare Function SP5_Ux Lib "ProgrammationSYSAM_SP5.dll" (ByVal voie1 As In-
teger, ByVal mesure1 As Integer) As Double
```

Renvoie la tension entrée sur la voie1, en fonction du calibre et de la valeur numérique. Cette fonction est utilisée pour la conversion de la valeur numérique entière (VNE) résultant d'une mesure (entre 0 et 4095) en la valeur de la tension mesurée.
Paramètres :
Voie1 = entrée analogique concernée (EA0 à EA7). Le calibre courant de cette voie est utilisé pour le calcul, de même que les données de calibrage de SYSAM SP5.
Mesures1 = valeur numérique (entre 0 et 4095) = résultat d'une mesure antérieure
RESULTAT : tension mesurée, tenant compte des paramètres de calibrage de la voie de SYSAM SP5 utilisée.

```
Private Declare Function SP5_ValUx Lib "ProgrammationSYSAM_SP5.dll" (ByVal voie1 As
Integer, ByVal U1 As Single) As Integer
```

Fonction réciproque de la précédente.
Paramètres :
Voie1 = entrée analogique concernée (EA0 à EA7). Le calibre courant de cette voie est utilisé pour le calcul
U1 = tension à convertir en valeur numérique
RESULTAT : valeur numérique (entre 0 et 4095) correspondant à la valeur de tension.

3. TOUT POUR L'EMISSION

```
Private Declare Sub SP5_DeclencheSA Lib "ProgrammationSYSAM_SP5.dll" (ByVal SA As Integer)
```

Démarrage de l'émission.
Valeurs du paramètre: SA
SA = 1 pour SA1
SA = 2 pour SA2
SA = 3 pour SA1 et SA2 en même temps.

```
Private Declare Sub SP5_ArreteSA Lib "ProgrammationSYSAM_SP5.dll" (ByVal SA As Integer)
```

Arrêt de l'émission.
Valeurs du paramètre: SA
SA = SA1= 1 pour SA1
SA = SA2 = 2 pour SA2
SA = SA1 + SA2 pour SA1 et SA2 en même temps

```
Private Declare Sub SP5_DeclencheSAetAcq Lib "ProgrammationSYSAM_SP5.dll" (ByVal SA As Integer)
```

Déclenche simultanément l'émission et l'acquisition
Valeurs du paramètre: SA
SA = SA1= 1 pour SA1
SA = SA2 = 2 pour SA2
SA = SA1 + SA2 pour SA1 et SA2 en même temps

```
Private Declare Sub SP5_ArreteSAetAcq Lib "ProgrammationSYSAM_SP5.dll" ()
```

Arrête toutes les émissions et l'acquisition en une seule opération.

```
Private Declare Function SP5_ValUxSA Lib "ProgrammationSYSAM_SP5.dll" (ByVal SA As Integer, ByVal U1 As Single) As Integer
```

Renvoie la valeur numérique correspondant à la tension U1, en tenant compte des paramètres d'étalonnage de SYSAM.
Paramètres :
SA: sortie concernée (valeurs: SA1 = 1 ou SA2 = 2)
U1: tension en V
Résultat: valeur numérique comprise entre 0 et 4095 correspondant à la tension U1

Fonction utilisée pour obtenir la valeur à passer en paramètre de la procédure ***SP5_EmissionValContinue*** ci-dessous.

```
Private Declare Sub SP5_EmissionUContinue Lib "ProgrammationSYSAM_SP5.dll" (ByVal SAi As Integer, ByVal U1 As Single)
```

Provoque l'émission sur la sortie SAi (SAi = SA1 = 1 ou SAi = SA2 = 2) d'une tension continue de valeur U1.

```
Private Declare Sub SP5_EmissionValContinue Lib "ProgrammationSYSAM_SP5.dll" (ByVal SAi As Integer, ByVal vNum As Integer)
```

Provoque l'émission sur la sortie SAi (SAi = SA1 = 1 ou SAi = SA2 = 2) d'une tension correspondant à la valeur entière vNum comprise entre 0 (-10V) et 4095 (+10V).

Emission de signaux créés de toutes pièces

Pour créer un signal de toutes pièces, il faut utiliser les procédures suivantes, dans l'ordre, avant de déclencher l'émission.
Le principe est de réserver une portion de la RAM de SYSAM devant recevoir un nombre donné de VNE, et de transmettre ces VNE à la RAM. Ces VNE sont converties par le CNA de SYSAM en signal de tension appliqué sur la sortie (SA1 ou SA2) concernée.

Etape 1 : paramétrage de l'émission.

```
Private Declare Sub SP5_ParametreEmission Lib "ProgrammationSYSAM_SP5.dll" (ByVal
SAi As Integer, ByVal deltat As Single, ByVal nbVal As Long)
```

On paramètre
- la sortie utilisée,
- l'intervalle de temps entre deux valeurs (temps d'échantillonnage, qui sera arrondi au multiple de te, temps d'échantillonnage minimal de SYSAM)
- le nombre de valeurs (indirectement la mémoire de SYSAM réservée aux valeurs du signal). Dans le contexte de l'outil « *ProgrammationSYSAM_SP5.dll* », ce nombre est limité à 100000 par défaut.

```
Private Declare Sub SP5_MonocoupSA Lib "ProgrammationSYSAM_SP5.dll" (ByVal SA As In-
teger, ByVal monocoupSA0 As Boolean)
```

Positionne la sortie SA sur monocoup si monocoupSA0 vaut true. Une émission monocoup s'arrête dès que la série de valeurs transmises à la RAM de SYSAM a été émise (une seule fois) sur la sortie concernée.
En mode permanent, l'émission est cyclique: l'émission de la série de valeurs reprend à son début sans interruption.
Pour émettre un signal périodique, il faut que monocoupSA0 soit sur false, ce qui est le cas par défaut (au démarrage et après l'appel de ***SP5_Reinitialisation***).

Etape 2 : préparation du tableau des valeurs.

```
Private Declare Sub SP5_PlaceValeur Lib "ProgrammationSYSAM_SP5.dll" (ByVal SAi As
Integer, ByVal position As Integer, ByVal valeur As Single)
```

Cette procédure permet de placer la valeur « valeur » à la position « position » dans le tableau des valeurs de sortie de la sortie SA1. « Valeur » est compris entre -10V et + 10V (résolution 12 bits). On répète cette procédure jusqu'à ce que le nombre nbVal de cases soit correctement rempli (position doit varier de 0 à nbVal - 1).

Etape 3 : transmission des valeurs à la RAM de SYSAM.

```
Private Declare Sub SP5_TransmetValeurs Lib "ProgrammationSYSAM_SP5.dll" (ByVal SAi
As Integer)
```

Provoque la transmission des valeurs numériques à la RAM de SYSAM. Ne doit être appelée que si les nbVal ont été préparées par la procédure SP5_PlaceValeur, et avant le déclenchement de l'émission.

Etape 4 : utiliser ***SP5_DeclencheSA*** (ci-dessus) pour déclencher l'émission.

4. PORT LOGIQUE B.

L'accès physique au port logique B de SYSAM nécessite le boîtier BOLOGIC.

Signification des notations:
plb : port logique B
vpb : valeurs du port logique.

```
Private Declare Sub SP5_EmetPLB0_3 Lib "ProgrammationSYSAM_SP5.dll" (ByVal B1 As Byte)
```

Emet sur le port logique B, bits 0 à 3, les 4 premières valeurs (0 ou 1) d'un tableau (par exemple déclaré sous vpb), dont seule la première case (vpb(0)) est passée par valeur dans B1 lors de l'appel de cette procédure
Exemple d'appel : SP5_EmetPLB0_3(vbp(0))
Autrement dit: fixe l'état des bits 0 à 3 du port logique aux 4 premières valeurs du tableau vpb.

```
Private Declare Sub SP5_EmetPLB4_7 Lib "ProgrammationSYSAM_SP5.dll" (ByVal B1 As Byte)
```

Emet sur le port logique B, bits 4 à 7, les 4 dernières valeurs (0 ou 1) d'un tableau (par exemple déclaré sous vpb), dont seule la première case vpb(0) est passée par valeur dans B1 lors de l'appel de cette procédure.

```
Private Declare Sub SP5_LitPLB0_3 Lib "ProgrammationSYSAM_SP5.dll" (ByRef B1 As Byte)
```

Lit les bits 0 à 3 du port logique B et les place dans un tableau (par exemple vpb) aux quatre premières places.
Le paramètre B1 sera la première case de vpb (vpb(0))
Exemple d'appel : SP5_LitPLB0_3(vpb(0))

```
Private Declare Sub SP5_LitPLB4_7 Lib "ProgrammationSYSAM_SP5.dll" (ByRef B1 As Byte)
```

Lit les bits 4 à 7 du port logique B et les place dans un tableau (par exemple vpb) aux quatre dernières places.
Le paramètre B1 sera la première case de vpb (vpb(0)).
Exemple d'appel : SP5_LitPLB4_7(vpb(0))

```
Private Declare Sub SP5_EmetPLB Lib "ProgrammationSYSAM_SP5.dll" (ByVal numBit As Byte, ByVal B1 As Byte)
```

Emet (fixe) individuellement le bit de valeur val1 (0 ou 1) sur port logique B

```
Private Declare Function SP5_LitPLB Lib "ProgrammationSYSAM_SP5.dll" (ByVal numBit As Byte) As Byte
```

Renvoie l'état du bit numBit (0 à 7) de valeur 0 ou 1 sur port logique B.

5. CAPTEURS

```
Private Declare Function SP5_Capteur Lib "ProgrammationSYSAM_SP5.dll" (ByVal canal As Byte) As Byte
```

Renvoie le numéro du capteur sur le canal « canal ».

Quatrième partie : créer un programme.

Rappels :

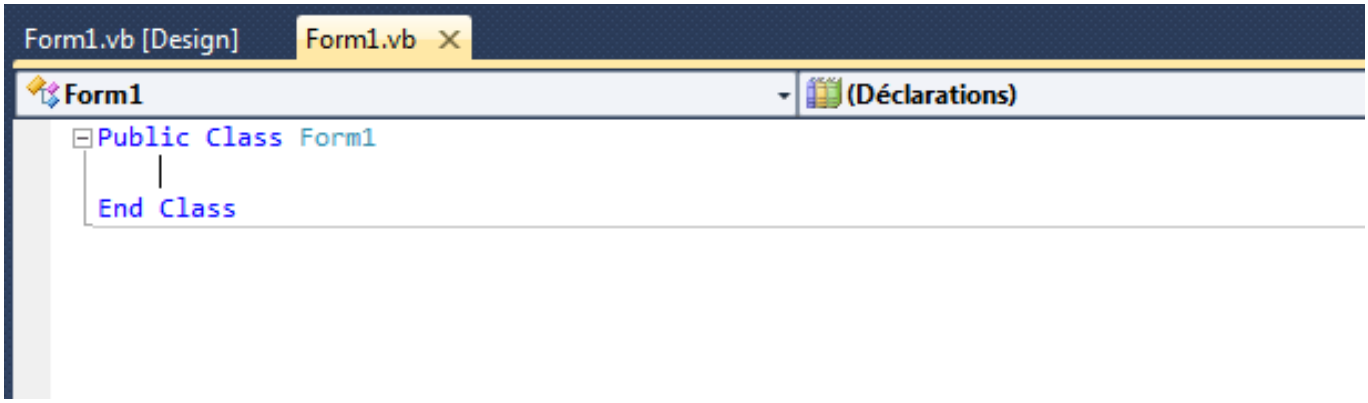
A. On compose la fenêtre (fiche) du programme en y plaçant des composants que l'on prend dans la boîte à outils (palette des composants). L'éditeur de propriétés (inspecteur d'objet) permet de fixer les propriétés initiales de ces composants (couleurs, taille, position sur la fenêtre, texte qu'ils contiennent...)

B. On fait de la "programmation orientée objet et événementielle": le programme lancé réagit à des événements (clics de souris sur des boutons, appuis de touches...) en exécutant les procédures correspondantes appelées gestionnaires d'événements. Tout l'art consiste à mettre dans ces gestionnaires un code bien adapté, ce qui reste assez facile pour des petits programmes.

Commençons.

1. Déclarations.

Dans le projet vierge créé dans la première partie, mettre en avant plan l'éditeur de code.



Après les mots réservés « Public Class » apparaît la déclaration Form1, qui est le type de la fiche-fenêtre originale que nous créons en ajoutant des composants.

Copier dans la ligne vide (à la position du curseur ci-dessus) l'intégralité du contenu du fichier « Déclarations_Visual_Basic.txt » fourni avec la DLL « ProgrammationSYSAM_SP5.dll). On remarquera la présence des types décrits plus haut ainsi que celle des constantes.

Examinons les déclarations copiées. Prenons l'exemple de la procédure :

```
Private Declare Sub SP5_DureesEtPoints Lib "ProgrammationSYSAM_SP5.dll" (ByVal dur-
eeTotale1 As Double, ByVal nbPts1 As Long)
```

Chacune des déclarations commence par les mots réservés **Private Declare** puis le mot **Sub** pour une procédure ou **Function** pour les fonctions (qui renvoient un résultat). Suit le mot réservé **Lib** qui indique que la procédure est accessible dans la DLL dont le nom suit (ici, **"ProgrammationSYSAM_SP5.dll").** Suit le nom de la procédure, puis éventuellement le ou les paramètres avec leur type, et s'il s'agit d'une fonction, le type du résultat.

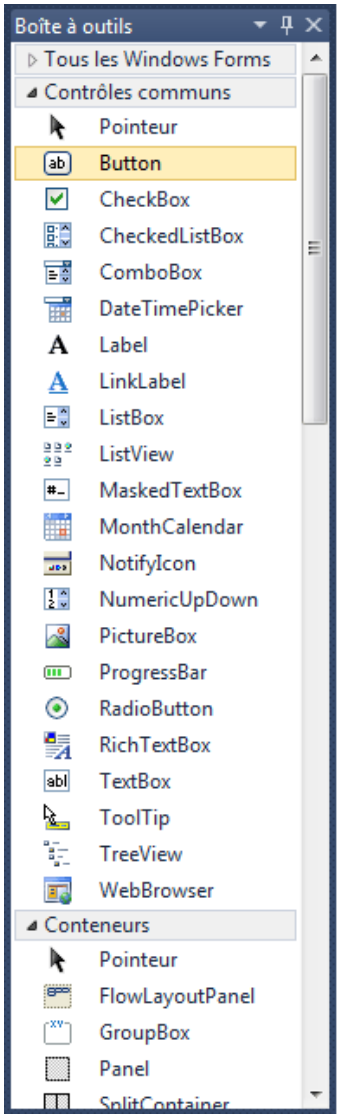
Cette façon de déclarer une procédure ou une fonction d'une DLL se nomme « chargement statique » de la DLL. Le programme utilisateur vérifie au démarrage (automatiquement) la présence de la DLL dans le dossier du programme ou dans le dossier System32 de

Windows, et « charge » les procédures ainsi déclarées. Si la DLL n'est pas présente, le programme ne peut pas fonctionner.

On peut ne déclarer que les procédures dont on a besoin.

2. Première acquisition (ou : réalisation d'un voltmètre).

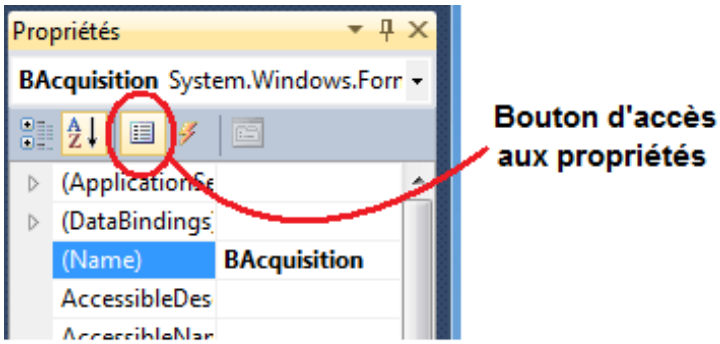
Réalisons un voltmètre en faisant mesurer par SYSAM une tension injectée sur son entrée EA0. Le déclenchement de la mesure peut être provoqué suite à l'événement de clic sur un bouton de la fenêtre du programme (action la plus courante des utilisateurs de logiciels…). Nous allons donc ajouter sur notre fiche-chantier un bouton et utiliser son gestionnaire d'événement de clic pour provoquer une acquisition.



•Remettez en avant plan la fiche chantier (MAJ + F7). Dans la boîte à outils, allez dans l'onglet « Contrôles communs ». Cliquez sur le bouton du composant Button (ci-contre)

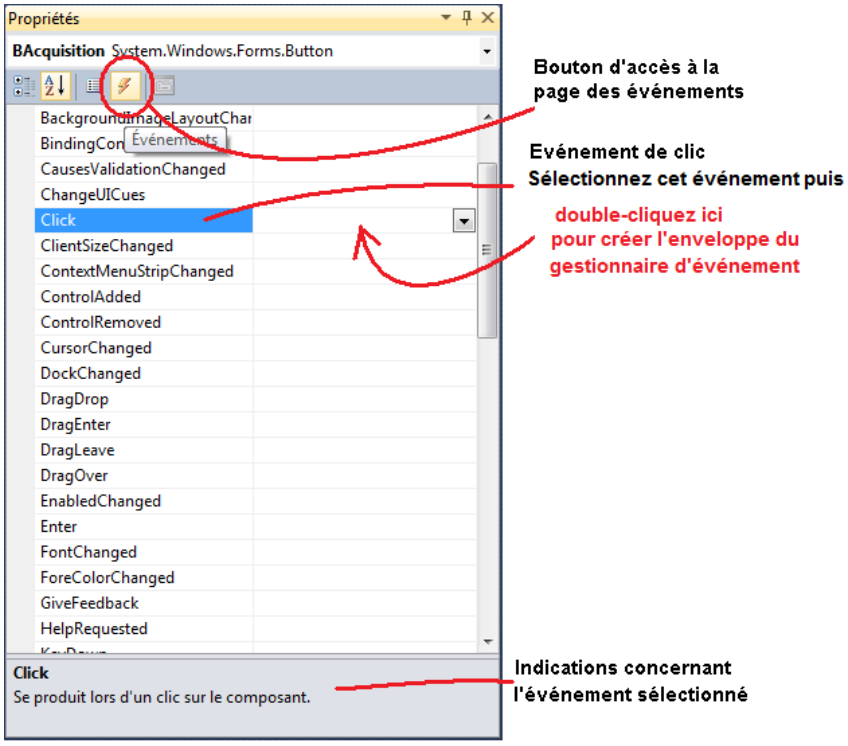
•Cliquez ensuite sur la fiche chantier, là où vous voulez déposer ce composant. En principe, à ce stade, le composant est géré dans l'éditeur de propriétés.

•Renommez le bouton : recherchez dans les « Propriétés » la rubrique « Name », et changer « Button1» en « BAcquisition ». Cette propriété doit respecter la syntaxe du Visual Basic, à savoir



- le premier caractère est une lettre (pas un chiffre)
- ni espace ni caractères accentués ou spéciaux. Le caractère _ peut être utilisé.

•Texte du bouton : recherchez la propriété « Text » et changez-la en « Acquisition ponctuelle ».



•Allez dans la page des événements de l'éditeur de propriétés (cliquez sur l'éclair)

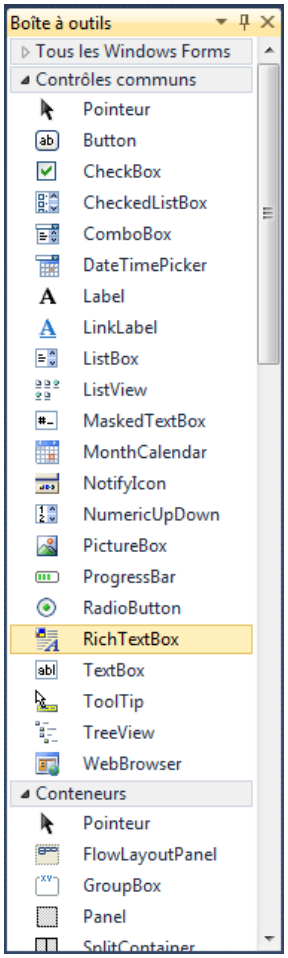
•Cliquez sur la partie à droite de l'événement Click (voir ci-contre), ce qui fait apparaître dans le code l'enveloppe de la procédure-gestionnaire de l'événement de clic sur le bouton.

```
    Private Sub BAcquisition_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles BAcquisition.Click

    End Sub
End Class
```

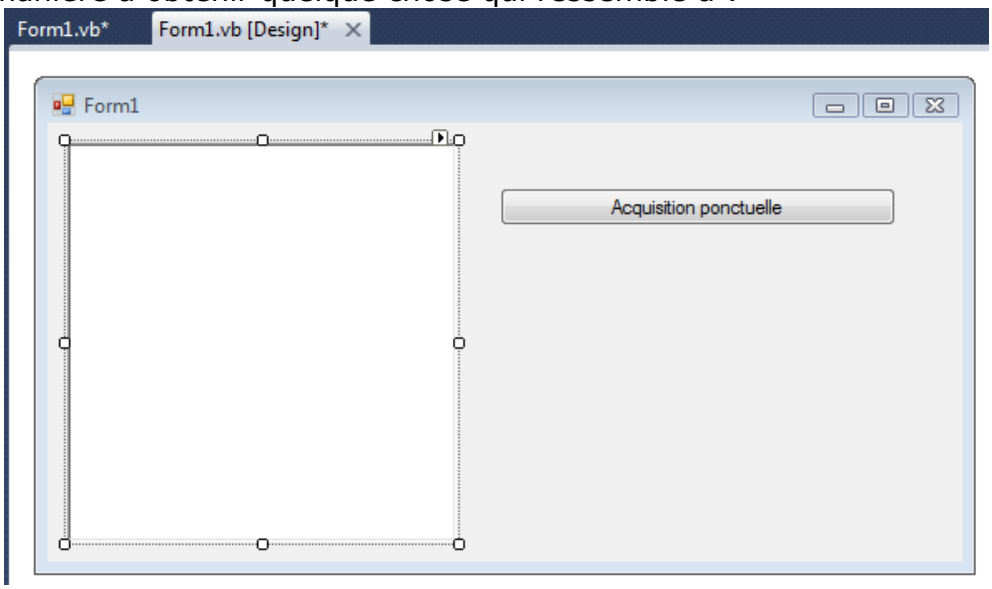
Entre la déclaration « Private… » et le « End Sub », ajoutez le code suivant :
(les textes précédés d'une apostrophe sont des commentaires : le compilateur n'en tient pas compte. Chaque commentaire indique ce que fera l'instruction qui suit lorsque ce code est appelé, i.e. lorsque l'événement correspondant a lieu.)

```
‘Déclarations de variables locales
'Variable entière sur 2 octets destinée à recevoir le résultat de la mesure
Dim Mesure1 As UShort
'Numéro de la voie utilisée'
Dim Voie As Integer

'Instructions du gestionnaire d'événements'
'Choix de la voie activée'
Voie = EA0

‘Activation de l'entrée analogique EA0 de SYSAM'
SP5_ActiveVoieA(Voie)

‘Réglage de son calibre
SP5_CalibreVoieA(voie, cal_10V)

'Mesure
SP5_AcquisitionVoieUnique(Voie, Mesure1)
```

Si à ce stade vous compilez puis exécutez le programme (touche F5), et si vous cliquez sur le bouton « Acquisition ponctuelle »), la VNE (valeur numérique entière) correspondant à la tension appliquée à l'entrée EA0 de SYSAM est mesurée et « stockée » dans la variable entière mesure1. Il faut maintenant ajouter les composants et les instructions qui permettent d'afficher cette valeur à l'écran. Un composant adapté à cet usage (parmi d'autres) est le RichTextBox, affichant du texte, auquel on rajoute des lignes de texte au fur et à mesure des besoins.



Remettez en avant plan la fiche chantier (MAJ + F7). Dans la boîte à outils, allez dans l'onglet « Contrôles communs ». Cliquez sur le bouton du composant « RichTextBox » (voir ci-contre), puis sur la fiche chantier. Positionnez et dimensionnez ce composant avec la souris de manière à obtenir quelque chose qui ressemble à :

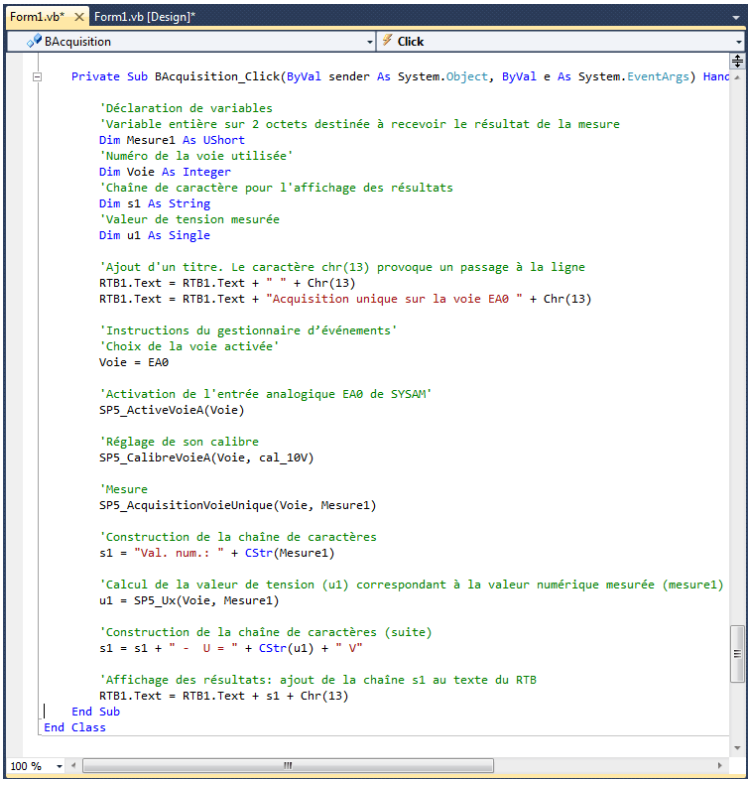


Dans l'éditeur de propriétés, remplacez la propriété « Name » par « RTB1 » (plus pratique pour le code).

Dans l'éditeur de code (F7), remplacez le code précédent du gestionnaire par :

```
'Déclaration de variables
 'Variable entière sur 2 octets destinée à recevoir le résultat de la mesure
 Dim Mesure1 As UShort
 'Numéro de la voie utilisée'
 Dim Voie As Integer
 'Chaîne de caractère pour l'affichage des résultats
 Dim s1 As String
 'Valeur de tension mesurée
 Dim u1 As Single

 'Ajout d'un titre. Le caractère chr(13) provoque un passage à la ligne
 RTB1.Text = RTB1.Text + " " + Chr(13)
 RTB1.Text = RTB1.Text + "Acquisition unique sur la voie EA0 " + Chr(13)

 'Instructions du gestionnaire d'événements'
 'Choix de la voie activée'
 Voie = EA0

 'Activation de l'entrée analogique EA0 de SYSAM'
 SP5_ActiveVoieA(Voie)

 'Réglage de son calibre
 SP5_CalibreVoieA(Voie, cal_10V)

 'Mesure
 SP5_AcquisitionVoieUnique(Voie, Mesure1)

 'Construction de la chaîne de caractères
 s1 = "Val. num.: " + CStr(Mesure1)

 'Calcul de la valeur de tension (u1) correspondant à la valeur numérique mesurée (mesu-
re1)
 u1 = SP5_Ux(Voie, Mesure1)

 'Construction de la chaîne de caractères (suite)
 s1 = s1 + " -  U = " + CStr(u1) + " V"

 'Affichage des résultats: ajout de la chaîne s1 au texte du RTB
 RTB1.Text = RTB1.Text + s1 + Chr(13)
```

Ci-contre, l'éditeur de code avec ce code :



•Le premier essai :

Reliez un générateur continu à l'entrée EA0, ainsi qu'un voltmètre, réglez sur une tension comprise entre –5V et +5 V, puis compilez et lancez le programme (F5). Cliquez sur le bouton « Acquisition ».

Votre première mesure est réalisée et s'affiche dans le RTB; comparez avec l'indication du voltmètre.

Changez le réglage du générateur ; recommencer l'acquisition…

3. Une autre acquisition (ou : réalisation d'un voltmètre multiple)

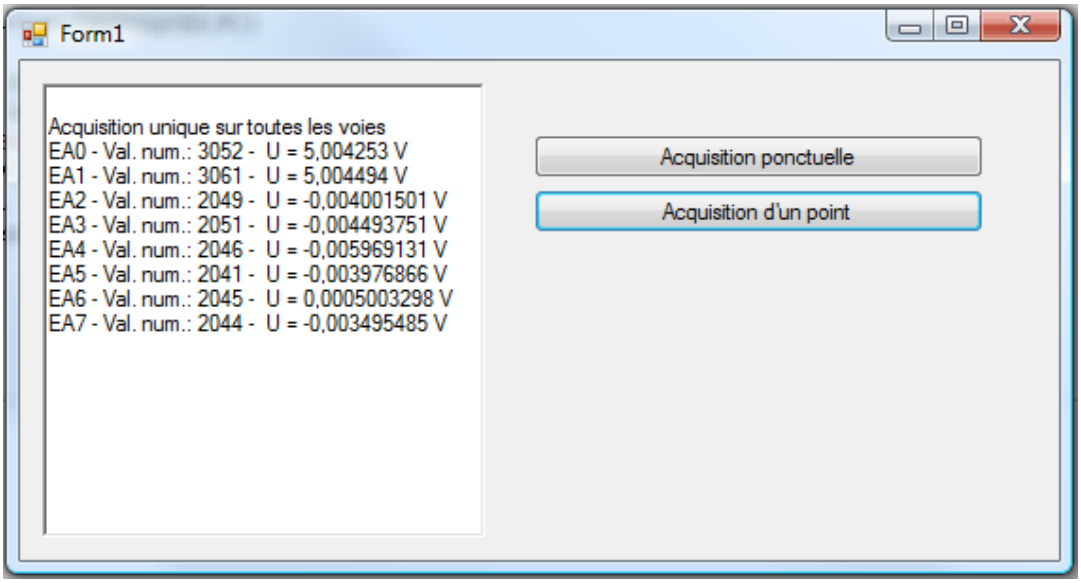
L'acquisition ne remet pas à zéro les paramètres. Si par exemple, on a activé une voie pour une acquisition, et qu'on ne souhaite pas l'utiliser lors de l'acquisition suivante, il faut la désactiver avant de lancer l'acquisition suivante.

Dans ce second exemple, nous allons réaliser une acquisition sur les huit entrées analogiques en même temps, en utilisant le calibre +/-10 V sur les huit entrées.

Cette seconde acquisition sera déclenchée par un autre bouton que vous placerez à votre convenance sur la fiche-chantier. Assurez-vous que le bouton est bien dans l' éditeur de propriétés. Renommez ce bouton (propriété « Name » = BAcquisitionPoint), changez son « text » en « Acquisition d'un point » ; affichez la page des événements de l'éditeur de propriétés ; double cliquez à droite de la rubrique onClick. Vous obtenez dans le code l'enveloppe du gestionnaire de l'événement de clic sur ce bouton.
Placez à cet endroit (entre « Private… » et « End Sub » le code suivant :

```
'GESTIONNAIRE DU CLIC SUR LE BOUTON BAcquisitionUnique

        'Variables locales

        'Tableau destiné à recueillir les valeurs des mesures effectuées
        'par l'appel à la procédure SP5_Acquisition (acquisition ponctuelle
        'sur toutes les entrées actives)
        '8 valeurs numérotées de 0 à 7, constituées d'entier positifs
        'compris entre 0 et 65535 (2 octets = 16 bits)
        'Dans la pratique, les valeurs du tableau sont comprises
        'entre 0 et 4095 (12 bits seulement)
        'On passe par référence la première case de ce
        'tableau (TabMesures(0)) à la procédure SP5_Acquisition
        Dim TabMesures(7) As UShort

        'Variables auxiliaires :
        'Chaîne de caractères pour l'affichage de résultat
        Dim s1 As String
        'valeur numérique du résultat
        Dim U1 As Single

        'Facultatif : SP5_Reinitialisation()
        'Effacement des lignes de la RTB1
        RTB1.Text = ""
        'Ajout d'un titre. Le caractère chr(13) provoque un passage à la ligne
        RTB1.Text = RTB1.Text + " " + Chr(13)
        RTB1.Text = RTB1.Text + "Acquisition unique sur toutes les voies " + Chr(13)


        'Activation des 7 entrées analogiques de SYSAM, calibre 10V'
        For voie = EA0 To EA7
            SP5_ActiveVoieA(voie)
            SP5_CalibreVoieA(voie, cal_10V)
        Next

        'Acquisition
        SP5_Acquisition(TabMesures(0))

        'Affichage des résultats
        For voie = EA0 To EA7
            s1 = "EA" + CStr(voie) + " - Val. num.: " + CStr(TabMesures(voie))
            U1 = SP5_Ux(voie, TabMesures(voie))
            s1 = s1 + " -  U = " + CStr(U1) + " V" + Chr(13)
            RTB1.Text = RTB1.Text + s1
        Next
```

Compilez et lancez (F5). Voici une capture d'écran après un clic sur « Acquisition d'un point » et une tension non nulle sur EA0 et EA1, on obtient :



Les mesures réalisées avec cette technique ne sont pas parfaitement simultanées. La technique ne convient donc pas aux mesures répétées trop rapidement sur des tensions variant vite. Pour cela, d'autres fonctions sont utilisées ; c'est l'objet du paragraphe suivant.

4. Une acquisition répétée (ou : réalisation d'un enregistreur)

Nous allons maintenant réaliser une acquisition répétée monocoup sur les entrées EA0 et EA3.
Donc, dans l'ordre, nous commandons :
a. l'activation des voies EA0 et EA3 et la désactivation de toutes les autres ;
b. le calibre 5 V sur ces deux entrées utilisées
c. 301 mesures réparties sur une durée de 100 ms.

Ajoutez un nouveau bouton sur la fiche, changez sa propriété « Name » en « BAcqPointsMultiples», son « text » en « Acquisition multiple monocoup » et créez l'enveloppe de son gestionnaire d'événement OnClick. Dans ce gestionnaire, placez le code suivant :

```
'================================================================================
'GESTIONNAIRE DU CLIC SUR LE BOUTON BAcquisitionMonocoup
'================================================================================
'Déclaration des variables locales (utilisées dans cette procédure)

'Taleau utilisé pour la récupération des valeurs du tampon d'acquisition
Dim TamponSP5(256) As UShort

'Paramètres d'acquisition
Dim dureeTotale As Single   'Durée totale de l'acquisition
Dim nbPts As Integer        'Nombre de points acquis
Dim nbPT As Integer         'Nombre de points dans le tampon d'acquisition

'Auxiliaires
Dim numPt As Integer        'Numéro du point en cours de traitement
Dim tp As Integer
Dim kp As Integer           'Compteur des points d'un tour
Dim posTp As Integer        'Position dans le tampon de la valeur en cours de traitement
Dim AcquisitionTerminee As Boolean 'Indicateur de fin d'acquisition
```

```
Dim s1 As String
Dim t1, u1 As Single
Dim vNum As UShort


'Facultatif :
'SP5_Reinitialisation()

'Effacement des lignes de la RTB1
RTB1.Text = ""

'Ajout d'un titre. Le caractère chr(13) provoque un passage à la ligne
RTB1.Text = RTB1.Text + " " + Chr(13)
RTB1.Text = RTB1.Text + "Acquisition Monocoup " + Chr(13)

'Choix des paramètres, à faire varier pour les tests
dureeTotale = 0.1
nbPts = 301
nbPT = 50 ' minimum : 1 ;  maximum :  256 / nombre de voies actives
'Le contenu du tampon correspond à un temps d'acquisition égal
'à  :  dureeTotale / nbPts * nbPtsDansTampon
'Cette valeur conditionne le temps réel: pour avoir un vrai temps réel,
'il faut que ce nombre de points puisse être traité par le programme pendant
'la durée correspondante, de manière à être traité pendant l'acquisition
'du tour suivant. Avec la RTB1, ce n'est pas le cas...

'Remise à l'état standard de SYSAM SP5 (facultatif: arrête l'émission,
'remet tous les calibres 10V, désactive les voies)
SP5_Reinitialisation()

'Désactivation de toutes les entrées
For voie = EA0 To EA7
    SP5_DesactiveVoieA(voie)
Next

'Activation de EA0 sur le calibre +-5V)
SP5_ActiveVoieA(EA0)
SP5_CalibreVoieA(EA0, cal_05V)

'Activation de EA3 sur le calibre +-5V)
SP5_ActiveVoieA(EA3)
SP5_CalibreVoieA(EA3, cal_05V)


'Réglage de la durée totale et du nombre de points à faire
'mesurer par SYSAM et à recueillir
SP5_DureesEtPoints(dureeTotale, nbPts)

'Choix du nombre de points dans le tampon
SP5_NbPtsDansTamponAcq(nbPT)

'Préparation diverses
AcquisitionTerminee = False
'Mise à zéro du compteur de points
numPt = 0

'Déclenchement
SP5_DeclencheAcqMonoCoup()
'On tente en boucle de lire le contenu du tampon de transmission
'des valeurs de SYSAM vers le programme utilisateur
Do Until AcquisitionTerminee
    tp = SP5_LireTamponAcq(TamponSP5(0))
    'à l'instant où tp est supérieur à  0, on sait que le
    'tampon a été rempli entièrement de mesures et que
    'ces mesures ont été renvoyées au programme dans le
    'tableau TamponSP5. Elle peuvent donc être traitées.
    'La phase de traitement sera nommée "tour"

    'TRAITEMENT DES VALEURS récupérées en temps quasi- réel.
```

```
If tp > 0 Then
    'Comme exemple, un simple ajout des valeurs récupérées dans
    'le memo1, avec une mise en forme sommaire

    'Mise à zéro du compteur des points du tour
    kp = 0

    'Boucle des points d'un tour
    Do While kp < SP5_NbPtsTampon And numPt < SP5_NbPtsAcq

        'Construction d'une chaîne de caractères correspondant au point traité
        s1 = CStr(numPt) 'CStr convertit un nombre en chaîne de caractères

        'Instant de la mesure du point
        t1 = numPt * SP5_DtAcq
        'Construction de la chaîne d'affichage de l'instant
        s1 = s1 + " - t = " + CStr(t1) + "s" + Chr(13)

        'Ecriture de la chaîne dans le RTB1
        RTB1.Text = RTB1.Text + s1


        'Position dans le tampon de la première valeur du point (EA0)
        posTp = kp * SP5_nbVoiesActives

        'Valeur numérique acquise sur EA0, lue dans le tampon
        vNum = TamponSP5(posTp)

        'Construction de la chaîne d'affichage du résultat
        s1 = " - EA0  --> " + CStr(vNum)

        'Valeur de la tension mesurée (numéro de voie = EA0 =  0)
        u1 = SP5_Ux(EA0, vNum)

        'Ajout de la valeur de tension dans la chaîne
        s1 = s1 + " ---- u = " + CStr(u1) + "V" + Chr(13)

        'Ecriture de la chaîne dans le RTB1
        RTB1.Text = RTB1.Text + s1



        'Position dans le tampon de la seconde valeur du point (EA3)
        posTp = posTp + 1

        'Valeur numérique acquise sur EA3
        vNum = TamponSP5(posTp)

        'Construction de la chaîne d'affichage du résultat
        s1 = " - EA3  --> " + CStr(vNum)

        'Valeur de la tension mesurée (numéro de voie = EA3 = 3)
        u1 = SP5_Ux(EA3, vNum)

        'Ajout de la valeur de tension dans la chaîne
        s1 = s1 + " ---- u = " + CStr(u1) + "V" + Chr(13)

        'Ecriture de la chaîne dans le RTB1
        RTB1.Text = RTB1.Text + s1

        'Compteur de points global : on passe au numéro suivant
        numPt = numPt + 1

        'Compteur de points du tour : on passe au numéro suivant
        kp = kp + 1

        'fin du traitement d'un point du tampon
    Loop 'Do while
```

```
        End If ' tp > 0
        'Test de fin d'acquisition. On se base sur le nombre de points souhaité  (nbPts) car
        '1.  le tampon peut ne pas être rempli à nbPtsTampon Points au dernier tour,
        '    si le nombre total de points n'est pas un multiple entier de nbPtsTampon
        '2.  le nombre de points renvoyé par SP5_nbPtsAcq et sur lequel est paramétré SYSAM
        '    peut être supérieur au nombre souhaité pour des raisons techniques
        If numPt >= nbPts - 1 Then AcquisitionTerminee = True
    Loop  'until acquisitionTerminee

    'Facile à deviner…
    SP5_ArreteAcq()

    'Désactivation des voies
    SP5_DesactiveVoieA(EA0)
    SP5_DesactiveVoieA(EA3)
```

Lors du lancement du programme, un clic sur « Acquisition multiple monocoup » provoque donc ici l'affichage des VNE acquises et converties en tensions dans le RTB1. C'est l'écriture dans le RTB1 qui prend su temps ; si on fait dessiner une courbe au programme, les choses seraient quasi instantanées.

Branchez un GBF sur les entrées EA0 et EA3, faire les réglages et lancer une acquisition. Testez en faisant varier les paramètres (dureeTotale, nbPts, nbPT). Ne poussez pas trop sur le nombre de points et la durée totale, car dans l'état actuel du code, on peut difficilement interrompre une acquisition trop longue.

5. Emission d'une tension continue.

Ajoutez un nouveau bouton sur la fiche, changez sa propriété « name » en « BEmissionUContinue », son « caption » en « Emission tension continue » ; créez l'enveloppe de son gestionnaire d'événement OnClick, puis placez dans ce gestionnaire les instructions :

```
RTB1.Text = RTB1.Text + "Emission d'une tension continue de 2,0 V" + Chr(13)
SP5_EmissionUContinue(SA1, 2.0)
```

Lancez le programme, cliquez sur le bouton ; la sortie SA1 de l'interface doit alors fournir une tension de 2.0 V. Vérifiez avec un voltmètre…

Cependant cette procédure n'a peut-être pas une portée pédagogique suffisante. On peut donc la remplacer par les instructions suivantes (en gras), après avoir déclaré V1 comme single et vNum1 comme word dans l'entête du gestionnaire :

```
    Dim v1 As Single
    Dim Vnum1 As UShort

    RTB1.Text = RTB1.Text + "Emission d'une tension continue de 2,0 V" + Chr(13)

    'Choix de la valeur de la tension continue à émettre
    v1 = 2.0

    'Calcul de la valeur entière correspondante (VNE)
    Vnum1 = CInt(2047 * (1 + v1 / 10))

    'Emission de la valeur :
    SP5_EmissionValContinue(SA1, Vnum1)
```

Il ne reste plus qu'à tester…

6. Emission d'une tension sinusoïdale.

Principe : pour la sortie utilisée :

A. On prépare les paramètres d'émission :

a. Intervalle de temps entre deux valeurs successives : deltat. Cet intervalle est forcément un multiple du temps d'échantillonnage de SYSAM en sortie Te = 200 ns.
b. Le nombre de valeurs à « émettre », nbVal , limité à environ 100000 (environ 2/5èmes de la RAM de SYSAM)

Procédure utilisée :

```
Private Declare Sub SP5_ParametreEmission Lib "ProgrammationSYSAM_SP5.dll" (ByVal SAi As Integer, ByVal deltat As Single, ByVal nbVal As Long)
```

où SAi = SA1 ou SA2.

B. On prepare un tableau (caché dans « *ProgrammationSYSAM_SP5.dll* ») de valeurs

de tensions à émettre. Ce tableau est dimensionné par l'instruction précédente. Toutes les valeurs doivent être comprises entre –10 V et + 10 V. (calibre unique)

Procédure utilisée :

```
Private Declare Sub SP5_PlaceValeur Lib "ProgrammationSYSAM_SP5.dll" (ByVal SAi As Integer, ByVal position As Integer, ByVal valeur As Single)
```

C. On envoie les valeurs du tableau a la RAM de SYSAM

```
Private Declare Sub SP5_TransmetValeurs Lib "ProgrammationSYSAM_SP5.dll" (ByVal SAi As Integer)
```

Il faut avoir fait au moins une fois les trois opérations précédentes, dans l'ordre avant d'utiliser la suivante.

D. On declenche l'emission

```
Private Declare Sub SP5_DeclencheSA Lib "ProgrammationSYSAM_SP5.dll" (ByVal SA As Integer)
```

SAi = SA1 pour faire émettre la sortie analogique SA1
SAi = SA2 pour faire émettre la sortie analogique SA2
SAi = SA1 + SA2 pour commencer l'émission simultanée sur les deux entrées.

```
Private Declare Sub SP5_ArreteSA Lib "ProgrammationSYSAM_SP5.dll" (ByVal SA As Integer)
```

Si on veut changer le signal émis, il faut préalablement arrêter l'émission, et au minimum changer les valeurs du tableau.

On peut arrêter et re-émettre à loisir sans forcément re-préparer les valeurs ou les autres paramètres d'émission.

Lorsqu'on émet un signal périodique, il faut veiller à ce que la durée totale de l'émission soit un multiple entier de la période du signal.

Exemple de code pour l'émission d'un sinus

On cherche à émettre sur SA1 un signal sinusoïdal d'amplitude 9 V, de période 2,0 ms et de phase pi/9 radians.

Ajoutez un nouveau bouton sur la fiche, changez sa propriété « Name » en « BEmissionSinus », son « text » en « Emission d'une tension sinusoïdale » ; créez l'enveloppe de son gestionnaire d'événement OnClick, puis placez dans ce gestionnaire les instructions :

```
'================================================================================
 'GESTIONNAIRE DU CLIC SUR LE BOUTON BEmissionSinusSA1 provoquant l'émission sur SA1
 'd'un signal sinusoïdal
 '================================================================================

 Dim k As Integer
 Dim t, u As Single

 Dim per, ampl, phase As Single
 Dim sortieUtilisee As Integer


 'Paramètres du signal
 per = 0.002 's
 ampl = 9.0 'V
 phase = 3.1415927 / 9 'rad

 'Effacement des lignes de la RTB1
 RTB1.Text = " "
 'Ajout d'un titre. Le caractère chr(13) provoque un passage à la ligne
 RTB1.Text = RTB1.Text + " " + Chr(13)
 RTB1.Text = RTB1.Text + "Emission d''un signal sinusoïdal" + Chr(13)
 RTB1.Text = RTB1.Text + "Fréquence: " + CStr(1 / per) + " Hz" + Chr(13)
 RTB1.Text = RTB1.Text + "Amplitude: " + CStr(ampl) + " V" + Chr(13)

 'Choix de la sortie à utiliser
 sortieUtilisee = SA1
 '1.  On fixe les paramètres d'émission : 100 valeurs à transmettre , correspondant à
 'exactement une période du signal
 SP5_ParametreEmission(sortieUtilisee, per / 100, 100)
 'L'intervalle de temps entre deux valeurs émises est le 100eme de la période, arrondi
 'au multiple de 200ns le plus proche
 '2.  Remplissage du tableau des valeurs
 For k = 0 To 100 - 1
     t = k * per / 100
     u = ampl * Math.Cos(2 * 3.1415927 * t / per + phase)
     SP5_PlaceValeur(sortieUtilisee, k, u)
 Next
 '3. Transmission des valeurs à SYSAM
 SP5_TransmetValeurs(sortieUtilisee)
 '4. Déclenchement de l'émission
 SP5_DeclencheSA(sortieUtilisee)
```

Pour arrêter l'émission, placer un autre bouton sur la fiche chantier et créer son gestionnaire de clic. Placer dans le gestionnaire l'instruction destinée à arrêter l'émission :

```
SP5_ArreteSA(SA1)
```

Exercice : émettre une tension en créneaux, en dents de scie…
A Tester avec un oscilloscope…

7. Le port logique B et le boîtier BOLOGIC

Le boîtier BOLOGIC permet d'accéder individuellement aux huit bits du port logique B.

a. Emission sur les sorties logiques.

Ajoutez un bouton, renommez-le, choisissez un « text » significatif, créez l'enveloppe de son gestionnaire d'événement de clic.

Voici le code du gestionnaire du bouton déclenchant l'émission de la valeur 1 sur les ports 0 et 3.

```
'================================================================================
'GESTIONNAIRE DU CLIC SUR LE BOUTON BEmissionPortB provoquant grâce au code suivant
'la mise à l'état 1 (haut ou +5V ) des ports logiques B0 et B3
'================================================================================

'Effacement des lignes de la RTB1
RTB1.Text = " "
'Ajout d'un titre. Le caractère chr(13) provoque un passage à la ligne
RTB1.Text = RTB1.Text + " " + Chr(13)
RTB1.Text = RTB1.Text + "Mise à l'état 1 (haut ou 5V) des ports logiques B0 et B3" + Chr(13)
SP5_EmetPLB(0, 1)
SP5_EmetPLB(3, 1)
```

Testez avec un voltmètre et vérifiez l'allumage des diodes du boîtier BOLOGIC.

b. Lecture de l'état des ports logiques B4 et B7 :

Ajoutez un bouton, renommez-le, choisissez un « caption » significatif, créez l'enveloppe de son gestionnaire d'événement de clic, déclarez les variables suivantes :

```
'================================================================================
'GESTIONNAIRE DU CLIC SUR LE BOUTON BEmissionPortB provoquant grâce au code suivant
'la mise à l'état 1 (haut ou +5V ) des ports logiques B0 et B3
'================================================================================

Dim s1 As String

'Effacement des lignes de la RTB1
RTB1.Text = " "
'Ajout d'un titre. Le caractère chr(13) provoque un passage à la ligne
RTB1.Text = RTB1.Text + " " + Chr(13)
RTB1.Text = RTB1.Text + "Lecture de l'état des ports logiques B4 et B7" + Chr(13)
s1 = "   B4 : " + CStr(SP5_LitPLB(4)) + Chr(13)
RTB1.Text = RTB1.Text + s1
s1 = "   B7 : " + CStr(SP5_LitPLB(7)) + Chr(13)
RTB1.Text = RTB1.Text + s1
```

Imposez avec un générateur ou un fil un état logique à B4 et B7, puis testez.

8. Pour aller plus loin…

a. Les autres fonctions et procédures de SYSAM

A vous de tester et d'essayer les fonctions supplémentaires disponibles dans la *ProgrammationSYSAM_SP5.dll*… : synchronisation de l'acquisition…

b. Autres propriétés des composants

Pour améliorer l'aspect visuel de votre programme, vous pouvez, à l'aide de l'inspecteur d'objets, changer les couleurs des composants, la police de caractères utilisée, la taille de la fiche et des composants. Peut-être également remplacer le « text »(libellé) par défaut de la fiche-chantier (« Form1 ») par quelque chose de plus parlant.
Il suffit de cliquer sur le composant dans la fiche-chantier pour le faire « entrer » dans l'inspecteur d'objet, puis d'éditer les propriétés. On peut aussi les fixer dans l'événement « Load » de la fiche principale (fiche chantier)

c. Ajout de composants de réglage de la tension de sortie en émission.

Ou : comment fabriquer un générateur continu réglable ?

Utilisez un composant TrackBar (onglet « Tous les Windows Forms », « name » par défaut TrackBar1) et son événement ValueChanged, qui a lieu à chaque déplacement de son curseur par l'utilisateur.

Mettez sa propriété « maximum » à 100 et sa propriété « minimum » à –100 ;

Utilisez la correspondance

value = 100 <---> U = +10 V
value = -100 <---> U = -10 V

ce qui se traduit par l'équation :

```
u1 = value / 10
```

Ajoutez également un composant d'édition de type TextBox (nom par défaut TextBox1), pour l'affichage de la tension obtenue par le réglage du curseur .

Ce qui donne le code (avec une déclaration de variables) suivant pour le gestionnaire :

```
Dim u1 As Single

u1 = TrackBar1.Value / 10

'Affichage de la tension émise
TextBox1.Text = CStr(u1)


' émission
SP5_EmissionUContinue(SA1, u1)
```

Lancez le programme, branchez un voltmètre sur SA1, déplacez le curseur de la TrackBar…

d. Utiliser les événements de composants pour régler les paramètres d'acquisition et d'émission.

En prolongement et généralisation de l'esprit du paragraphe précédent, pour créer une application utilisable par un public (non programmeur) scolaire ou autre, le réglage des paramètres d'acquisition ou d'émission doit être réalisable pendant l'utilisation du programme. Il faut donc dans un projet EAXO placer les composants adéquats et utiliser les événements associés, en clair faire de la véritable programmation événementielle.

e. Travailler avec un capteur

Pour utiliser un capteur, il faut commencer par déterminer son numéro d'identification, ensuite faire la mesure (acquisition) comme s'il n'y avait pas de capteur, puis convertir la valeur de tension en valeur de grandeur physique en utilisant les indications de conversion données sur la notice du capteur par Eurosmart.

Exemple du capteur de température : la notice donne
dans la rubrique « Fonction de transfert » : Température(°C) = Tension(V) x 25
dans la rubrique « Etendue de mesure » : de -25°C à 125°C

On en déduit que les tensions extrêmes sont
-1 V (pour -25°C) et +5V pour (+125°C)
Il est judicieux d'utiliser le calibre -/+5V de SYSAM pour une acquisition.

Il suffit d'ajouter dans le code (après la mesure de la VNE et le calcul de la tension U1 correspondante) l'instruction teta = 25*U1, (après avoir déclaré la variable teta comme single) puis de faire afficher cette température dans le RTB1 ou ailleurs pour disposer d'un thermomètre…

On perçoit ici la multiplicité des réalisations possibles…

Conclusion

L'auteur de cet article se tient à la disposition du lecteur pour toute question concernant l'utilisation des instructions présentées dans cet article, mais seulement pour ces instructions et leur rapport avec SYSAM. Il n'est pas compétent en visual basic, la DLL ayant été créée avec DELPHI.

Pour les passionnés, les éléments de cet article suffisent pour créer des applications d'EXAO assez performantes et utilisables en classe avec des élèves, si on maîtrise Visual Basic par ailleurs. On peut imaginer d'inclure dans une application simple le dessin progressif d'un graphe…

Si votre projet finalisé doit être distribué (installé sur plusieurs machines dans un but d'utilisation collective , en classe par exemple), il faut veiller à fournir aux utilisateurs la DLL « *ProgrammationSYSAM_SP5.dll* » et à ce qu'elle se trouve dans le dossier du programme exécutable que vous avez créé, ou dans le dossier « C:\Windows\System32 » des ordinateurs des utilisateurs. L'absence de la DLL plante le programme dès le démarrage, du fait du chargement statique de la DLL.

Pour des suggestions de développements nouveaux, une seule adresse : contact@logisciences.fr , site de l'auteur.

30 août 2010

Jean-Marie THOMAS
Professeur de Sciences Physiques
Lycée Jean-Monnet - Strasbourg.