

PROJET SysamCampus.py

Programmation de SYSAM CAMPUS avec Python

A. Généralités

Le projet SysamCampus.Py est un ensemble d'outils et d'exemples mis à disposition pour une programmation « facile » avec Python, des interfaces d'acquisition SYSAM CAMPUS (souvent appelées « centrales » dans ce document) de la société Eurosmart. Ces outils sont :

- Des bibliothèques campus.dll (32 bits et 64 bits);
- Le module python sysam_campus_py.py qui met à disposition de nombres fonctions de commande de la centrale CAMPUS
- Le module python sysam_campus_gr.py qui met à disposition des fonctions pour tracés de graphiques sous pygame.
- De nombreux exemples d'utilisation des fonctions, certains très basiques, d'autres plus élaborés.

Le code est conçu de manière à rester aussi simple que possible, et que l'utilisateur n'ait besoin que d'un très petit minimum de connaissance de python pour l'utiliser. Cela pourra encourager les collègues à utiliser ces outils avec leurs élèves.

Lorsque vous créez un programme python, vous l'enregistrez dans un dossier de votre choix. Pour que le script du programme puisse fonctionner, il faut que dans ce dossier se trouvent :

1. Le module sysam_campus_py.py ;
Ce module doit se trouver dans le même dossier que le script du programme python à réaliser. Il utilise une dll (bibliothèque de fonctions en lien avec le pilote de la centrale). Il y a une dll pour les logiciels de codage en python en 32 bits (certaines versions de IDLE Python) et une autre dll pour les logiciels de codage en 64 bits (Edupython et IDLE Python 64 bits), dans deux dossiers distincts.
Le code de ce module ne doit à priori pas être modifié, sauf peut-être avec beaucoup de précautions et en connaissance de cause...
2. Les bibliothèques campus.dll ;
dans le sous-dossier « 32bit » si vous travaillez avec un python 32 bits ;
dans le sous-dossier « 64bit » si vous travaillez sous un python 64 bits (par exemple Edupython)
N.B. : Les campus.dll ne sont pas les mêmes dans ces deux sous-dossiers.
3. Le module sysam_campus_gr.py.
Ce module fournit les outils pour
 - Afficher les résultats des mesures avec un aspect (très sommaire) de multimètre dans une fenêtre pygame.
 - Afficher les résultats sous forme graphique lorsqu'on fait des acquisitions périodiques.
Le code de ce module peut être enrichi, complété, modifié ... par tout connaisseur de Python et pygame.
4. Les exemples

Les exemples de programmes Python fournis montrent l'usage de la majeure partie des instructions disponibles. Toutes les instructions ont été testées dans de nombreuses configurations de paramétrage, mais seulement avec 5 capteurs CAMPUS différents :

SON CAMPUS,
AMP CAMPUS (ampèremètre 4 calibres),

LUX CAMPUS (luxmètre avec deux calibres),
TK CAMPUS (thermocouple) et
ST CAMPUS (thermomètre de - 50°C à +150 °C)

Normalement, tous ces outils devraient fonctionner avec presque tous les capteurs de la gamme CAMPUS.

Malgré la multitude de tests réalisés par l'auteur, la présence d'erreurs est très probable. Prière de signaler toute erreur sur le contact du site <http://www.logisciences.fr> en transmettant le script python et avec une description sommaire du problème et un maximum d'indications sur le montage. Ne transmettre que le code spécifique des exemples ou du module sysam_campus_py.py qui posent problème.

Les compétences de l'auteur en python sont très limitées :

- il ne pourra pas répondre à des questions pointues sur python ;
- on peut certainement coder mieux (plus dans le style python).

La philosophie est de disposer d'outils qui ne suppriment pas la nécessité pour un élève (ou un professeur !) de bien penser la démarche de conception du code en rapport avec l'objectif de l'expérience à réaliser.

Conventions :

- Ce qui est appelé « Valeur numérique » et noté vn est un nombre entier sur 12 bits (entre 0 et 4095). Ce type de nombre résulte d'une conversion analogique numérique (entrées de CAMPUS), ou bien est le point de départ d'une conversion numérique analogique (sorties de CAMPUS)
- Une grandeur physique (notée gp) (qui peut aussi être une grandeur chimique ...), est le résultat de la restitution de la mesure faite, à partir de la vn acquise. Ce sera une tension si aucun capteur n'est branché.
- Le langage python fait la distinction entre majuscules et minuscules (attention à la « casse »).
- Dans la suite de ce document, on appellera « programme » ce qu'on cherche à construire pour faire fonctionner la centrale sysam campus dans une expérience donnée.
- Un certain nombre de procédures ont un nom qui commence par « Def ». Ceci signifie « Définition », c'est-à-dire qu'elles sont prévues pour donner des valeurs à des paramètres. En général, elles effectuent un contrôle d'intervalle : si la valeur passée en paramètre dépasse le domaine de validité, elles donnent au paramètre la valeur limite la plus proche ou une valeur par défaut sécurisée. Néanmoins, prudence...

B. La première phase.

Pour utiliser le module sysam_campus_py.py dans un programme python, il faut le déclarer, par exemple :

```
import sysam_campus_py as camp.
```

Toutes les « fonctions » de ce module pourront ainsi être appelées en les faisant précéder de « camp. ». De même pour les constantes prédéfinies.

Le module contient entre autres le code de chargement de la campus.dll, qui est exécuté dès le lancement du programme.

A la fin du programme, il faut veiller à faire le nécessaire pour que la centrale soit remise dans son état standard. C'est l'instruction « camp.Rangement() » qui effectue ce travail.

Il est nécessaire de s'assurer que la centrale campus est connectée à l'ordinateur avant de chercher à la programmer.

Tout programme devrait donc comporter un code minimal qui ressemble à ce qui suit :

```
1 # Créé par JM, le 20/05/2025 en Python 3.7
2
3 import sysam_campus_py as camp
4 import sys
5
6 print(" TITRE DU PROGRAMME ")
7 print()
8
9 # VERIFICATION DE LA PRESENCE DE LA CENTRALE SYSAM CAMPUS =====
10 if camp.Presente():
11     print("Vérification de la présence de la centrale SYSAM CAMPUS : Centrale CAMPUS connectée")
12 else:
13     print("Vérification de la présence de la centrale SYSAM CAMPUS : Centrale CAMPUS non connectée, programme interrompu")
14     camp.Rangement()
15     exit()
16 print()
17
18
19 # ici, toutes les instructions qui permettront au programme de faire ce qu'il doit faire
20 # dans l'expérience à réaliser.
21 #
22
23
24
25 camp.Rangement()
26
27
```

Ce code peut être copié de n'importe lequel des exemples fournis (ils le contiennent tous).

De nombreuses données sont stockées dans des listes, dont les éléments sont accessibles par leur indice. L'indice du premier élément est toujours 0. Par exemple, l'indice de l'entrée V1 est 0.

C. Architecture des entrées de SYSAM CAMPUS et paramétrage des canaux.

La centrale SYSAM Campus comporte 4 entrées que nous appellerons « Canaux ». Sur chaque canal on peut connecter un capteur, ou pas. Les capteurs fournis par Eurosmart peuvent fonctionner selon des modes différents : de 1 à 4 modes sont disponibles suivant le capteur. Ces modes peuvent correspondre à des grandeurs physiques différentes ou simplement à des calibres différents pour une seule grandeur physique, selon le capteur.

L'instruction de paramétrage d'un canal est :

camp.ActiveCanal (canal, vdc, idxCalMode)

- Le paramètre canal permet d'indiquer le canal à activer. Il est conseillé d'utiliser les constantes prédéfinies camp.C1 (canal 1 en haut à gauche), camp.C2, camp.C3 et camp.C4 (valeurs respectives 0,1,2 et 3).
- Si aucun capteur n'est branché sur le canal donné, seule l'entrée de tension peut être utilisée. Si au contraire, un capteur est branché sur ce canal, alors il faut choisir entre utiliser l'entrée de tension ou le capteur (on n'est pas forcé d'utiliser le capteur, même s'il est branché !). On parlera ici de voie du canal (abréviation utilisée : vdc)

Le paramètre vdc permet donc de choisir entre :

- utiliser l'entrée de tension du canal, valeur prédéfinie camp.vdcTension
- utiliser le capteur du canal, valeur prédéfinie camp.vdcCapteur
- Le troisième paramètre est l'index du mode/calibre choisi (forcément entre 0 et 3). Le module sysam_campus fournit des constantes prédéfinies pour les calibres des entrées de tension.

- Par défaut (initialement), tous les canaux sont désactivés.

Ex. 1 : Capteur luxmètre connecté au canal C2 (en bas à gauche) :

Ce capteur possède les deux modes suivants :
 mode 0 : mesure d'éclairement avec le calibre 250000 lx
 mode 1 : mesure d'éclairement avec le calibre 25000 lx

Pour activer ce capteur sur le calibre 25000 lx, on paramétrera l'instruction comme suit :
camp.ActiveCanal (camp.C2, camp.vdcCapteur, 1)

Ex. 2 : Capteur Ampèremètre connecté au canal C3 (en haut à droite) :

Ce capteur dispose de 4 modes qui sont en fait des calibres :
 mode 0 : calibre 5A ; mode 1 : calibre 1A ;
 mode 2 : calibre 200 mA ; mode 3 : calibre 40 mA

Pour activer ce capteur sur le calibre 40 mA, on paramétrera l'instruction comme suit :
camp.ActiveCanal (camp.C3, camp.vdcCapteur, 3)

Ex. 3 : Utilisation d'une entrée de tension, à savoir activation du canal C1 (V1) avec le calibre 10 V :

camp.ActiveCanal (camp.C1, camp.vdcTension, cal_CA_10V)
 (cal_CA_10V est une constante prédéfinie)

N.B. : si on tente d'activer un canal avec le paramètre vdcCapteur alors qu'aucun capteur n'est connecté au canal, la routine considère que c'est une erreur et par sécurité, active l'entrée de tension du canal avec le calibre 30V (constante prédéfinie : cal_CA_30V).

Si inversement on tente d'activer un canal avec le paramètre vdcTension alors qu'un capteur est connecté, le capteur n'est simplement pas pris en compte.

Propriétés des capteurs.

Pour sélectionner le mode/calibre à utiliser, il faut connaître les modes disponibles du capteur. Pour cela, on pourra utiliser l'exemple « _A_DetectionModesCapteur.py ». Cet exemple utilise l'instruction `camp.AfficherModesDispoCapteurs()`

```

1  # Créé par JM, le 18/05/2025 en Python 3.7
2
3  import sysam_campus_py as camp
4  import sys
5
6  print("RECHERCHE DES CARACTERISTIQUES DES CAPTEURS CONNECTES A SYSAM CAMPUS ")
7  print()
8
9  # VERIFICATION DE LA PRESENCE DE LA CENTRALE SYSAM CAMPUS =====
10 if camp.Presente():
11     print("Vérification de la présence de la centrale SYSAM CAMPUS : Centrale CAMPUS connectée")
12 else:
13     print("Vérification de la présence de la centrale SYSAM CAMPUS : Centrale CAMPUS non connectée, programme interrompu")
14     camp.Rangement()
15     exit()
16 print()
17
18
19 # Recherche des caractéristiques
20 camp.AfficherModesDispoCapteurs()
21 camp.Rangement()
22 |

```

Voici ce qu'on peut obtenir en sortie :

```

Console Python
RECHERCHE DES CARACTERISTIQUES DES CAPTEURS CONNECTES A SYSAM CAMPUS
Vérification de la présence de la centrale SYSAM CAMPUS : Centrale CAMPUS connectée

=====
CANAL 1 : capteur détecté
Designation du capteur  ASON-120DB

Mode 0 -----
Description      : son +/- 2.5V
Unite           : V
Nom de la courbe : Amplitude son
Minimum         : -2.5
Maximum         : 2.5
Symbole de la gp : uSon

Mode 1 -----
Le capteur ne comporte pas de mode 1

Mode 2 -----
Description      : Son 0-120dB
Unite           : dB
Nom de la courbe : Niveau sonore
Minimum         : 0.0
Maximum         : 120.0
Symbole de la gp : Lson

Mode 3 -----
Description      : Son 0-90dB
Unite           : dB
Nom de la courbe : Niveau sonore
Minimum         : 0.0
Maximum         : 90.0
Symbole de la gp : Lson

=====
CANAL 2 : capteur détecté
Designation du capteur  CL250K

Mode 0 -----
Description      : Luxmètre
Unite           : lx
Nom de la courbe : E
Minimum         : 0.0
Maximum         : 250000.0
Symbole de la gp : E

Mode 1 -----
Description      : Luxmètre

```

D. Quelques définitions

Le lecteur ayant des connaissances en acquisition, conversion analogique-numérique et conversion numérique – analogique peut éventuellement passer directement au paragraphe suivant.

- Acquisition : réalisation de mesures par l'intermédiaire de la centrale SYSAM CAMPUS.
- Point : ensemble des mesures réalisées sur toutes les voies actives à un instant donné.
- Acquisition périodique : acquisition de points à intervalle de temps fixe (appelé temps d'échantillonnage T_e)
- Tampon : tableau où sont stockées provisoirement des valeurs résultant de mesures partielles.
- Déclenchement : c'est le fait de commander à la centrale de commencer les mesures périodiques.
- Temps d'échantillonnage T_e : temps séparant deux mesures de points consécutifs lors d'une acquisition périodique.
- Synchronisation : le fait de soumettre le démarrage d'une acquisition périodique à des conditions sur la variation du signal choisi comme référence.

E. Les constantes prédéfinies du module sysam_Campus_py.py

En réalité, sous Python, ce ne sont pas réellement des constantes, mais elles doivent être utilisées comme telles. Leur utilisation est fortement recommandée comme paramètres des fonctions/procédures décrites plus loin.

Index des calibres	cal30V = 0	cal15V = 1	cal10V = 2	cal01V = 3
Numéros des sorties	SA1 = 1 (borne verte 1 de Campus) SA2 = 2 (borne verte 2)			
Numéros des canaux	C1 = 0	C2 = 1	C3 = 2	C4 = 3
Voies des canaux	vdcTension = 0 vdcCapteur = 1			

SYNCHRONISATION	
Modes de déclenchement	mdAucun = 0 pas de synchro mdSeuil = 1 déclenchement sur seuil mdExterne = 2 déclenchement sur signal injecté dans l'entrée Trig (borne jaune de Campus)
Sens de déclenchement	sdMontant = 0 sdDescendant = 1

F. Les variables globales prédéfinies du module `sysam_campus_py.py`

IMPORTANT : Ces variables sont utilisables à tout instant, mais ne pas modifier directement leurs valeurs : il y a trop de risques d'incohérence. Pour leur donner une valeur, utiliser systématiquement les routines prévues à cet effet (cf §G). Utiliser leur valeur ne présente aucun problème (elles sont même faites pour ça)

MesureVN	Recueille la valeur numérique unique lue sur un canal par la fonction <code>AcquisitionUnPointSurUnCanal</code>
MesureGP	Recueille la valeur de la grandeur physique unique, qui doit être calculée à partir de <code>MesureVN</code> (pour des raisons pédagogiques, ce n'est pas fait automatiquement...)
TabMesuresVN = [4 valeurs]	Tableau des mesures pour la procédure pour recueillir les valeurs numériques obtenus par l'utilisation de la fonction : <code>MesureDirecte</code>
TabMesuresGP = [4 valeurs]	Tableau des grandeurs physiques correspondantes
gpM = [[], [], [], []]	Tableau dans lequel seront consignées les grandeurs physiques acquises lors des acquisitions périodiques (y compris les tensions si pas de capteur). Une « colonne » par canal, indexation par canaux réels, il peut donc y avoir des colonnes vides correspondant aux canaux non activés.
CanauxActifs = []	Tableau listant les numéros des canaux actifs. Contient 4 valeurs comprises entre C1 et C4 (soit entre 0 et 3), ou bien -1 (canal non activé). Il est automatiquement cohérent avec l'activation et la désactivation des canaux avec les routines définies plus loin. Cette variable ne doit pas être modifiée directement, mais utilisée en lecture seule.
nbCanauxActifs	Nombre de canaux actuellement actifs. Même remarque : en lecture seule.
nbCapteursActifs	Nombre de capteurs actuellement connectés et activés. En lecture seule.
DureeTotale	Durée totale d'acquisition pour une acquisition périodique. Valeur par défaut : 1.000 s. Valeur minimale de l'ordre de la microseconde. A définir avec la routine : <code>DureesEtPoints(dureeTotale1, nbPts1)</code>
NbPts	Nombre de points à acquérir lors d'une acquisition périodique. Valeur par défaut : 10 ; valeur minimale : 10. Valeur maximale : 4000. A définir avec la routine <code>DureesEtPoints(dureeTotale1, nbPts1)</code>
Te	Temps d'échantillonnage de l'acquisition périodique. Ne pas le définir directement. Est défini par la routine : <code>DureesEtPoints(dureeTotale1, nbPts1)</code>
NbPtsBlocAcq	Nombre de points dans le bloc/tampon d'acquisition, pour les acquisitions périodiques longues avec affichage en temps (quasi) réel. Voir plus loin.
DureeBlocAcq	Durée du tampon d'acquisition (durée correspondant au nombre de points du bloc/tampon)

TeSA	Temps d'échantillonnage des sorties (tableau à deux valeurs, une par sortie)
NbValSA	Nombre de valeurs pour un cycle d'émission sur une sortie (tableau à deux valeurs, une par sortie)
DureesTotalesSA	Durée d'un cycle d'émission (tableau à deux valeurs, une par sortie)
DernierPtAcquis	Dernier point acquis lors d'une acquisition par blocs.
PremierPtAcquis	Premier point acquis lors d'une acquisition par blocs.
AcquisitionTerminee	Comme son nom l'indique.
NbTotalPtsLus	Technique
SYNCHRONISATION	
ModeDecl	Mode de déclenchement courant.
SensDeclSeuil	Sens de déclenchement sur seuil
ValSeuilDecl	Valeur de seuil de déclenchement
CanalSynchro	Canal sur lequel on synchronise
TauxPretrig	Taux de pré-trig entre 0 et 100 % (explications plus bas)
NbPtsPretrig	Nombre de points correspondant au pretrig
DureePretrig	Durée du prétrig correspondante. Ces trois valeurs sont cohérentes. Ne pas les affecter directement.

G. Fonctions générales utilitaires

Le préfixe fm signifie format.

Fonction fmt (t1, nbCS)	<p>Renvoie une chaîne de caractères représentant le temps t1 avec nbCS chiffres significatifs, en écriture ingénieur, avec l'unité s.</p> <p>Paramètres :</p> <p>t1 : temps</p> <p>nbCS : nombre de chiffres significatifs</p>
Fonction fmf (f, nbCS)	<p>Renvoie une chaîne de caractères représentant le temps t1 avec nbCS chiffres significatifs, en écriture ingénieur, avec l'unité Hz</p> <p>Paramètres :</p> <p>f : fréquence</p> <p>nbCS : nombre de chiffres significatifs.</p>
Fonction fmlnge (v, n, nMin)	<p>Renvoie une chaîne de caractères représentant la valeur v avec n chiffres significatifs, en écriture ingénieur (utilisation des préfixes kilo, méga, ...), sans unité</p> <p>Paramètres :</p> <p>v : grandeur quelconque ;</p> <p>n : nombre de chiffres significatifs</p> <p>nMin : nombre minimal de chiffres significatifs.</p>
Fonction fmlnge1 (v, n, nMin)	<p>Renvoie</p> <ul style="list-style-type: none"> - une chaîne de caractères représentant la valeur v avec n chiffres significatifs, en écriture ingénieur, sans les préfixes

	<ul style="list-style-type: none"> - le préfixe (utilisation des préfixes kilo, méga, ...) , sans unité Paramètres : v : grandeur quelconque ; n : nombre de chiffres significatifs nMin : nombre minimal de chiffres significatifs.
Fonction ArronditReel(v1, n)	Arrondit un réel au nombre de chiffres significatifs n. Aucun formatage n'est effectué.

H. Fonctions de contrôle de SYSAM CAMPUS pour l'acquisition

Ces routines ont globalement été pensées pour ne faire qu'une seule chose à la fois.

ROUTINES GENERALES	
Presente ()	Test de présence de la centrale. Renvoie True si centrale est connectée et alimentée et False sinon.
Rangement ()	Libération des mémoires, des dll, remise en état initial de SYSAM CAMPUS. A appeler systématiquement en fin de programme.

TOUT POUR L'ACQUISITION	
ActiveCanal (Canal, vdc, idxCalMode)	Vu précédemment §C. Met à jour les variables CanauxActifs[], nbCanauxActifs et quelques autres.
DesactiveCanal (Canal)	Désactivation d'un canal. Met à jour les variables CanauxActifs[], nbCanauxActifs et quelques autres.
ValCanal (canal, vn)	Fonction de calcul de la grandeur physique ou de la tension à partir de la valeur numérique mesurée vn, avec ou sans utilisation de capteur. Pour que cette fonction donne un résultat correct, il faut que le canal soit actif.
AcquisitionUnPointSurUnCanal (canal)	Acquisition sur canal unique - Renvoie la vn acquise et l'affecte également à la variable prédéfinie MesureVN. Il faut ensuite la convertir en une tension ou une grandeur physique, de préférence avec la fonction fournie ValCanal (qui tient compte de l'étalonnage des capteurs et de SYSAM CAMPUS)
MesureDirecte ()	Acquisition unique simultanée sur toutes les canaux actifs (acquisition d'un et un seul point). Les résultats sont placés dans les tableaux TabMesuresVN [] et TabMesuresGP [] (une valeur par canal)
Description(canal, mode)	Envoie une chaîne de caractères décrivant le mode du capteur branché sur le canal.
UniteGP (canal)	Récupération de l'unité de la grandeur du canal
NomGP (canal)	Récupération du nom officiel de la grandeur physique du canal
RefCapteur (canal)	Récupération de la référence du capteur sur le canal
NomCapteur (canal)	Récupération du nom du capteur sur le canal

MinCanal (canal)	Récupération de la valeur minimale mesurable sur le canal.
MaxCanal (canal)	Récupération de la valeur maximale mesurable sur le canal
ExpCalibreCanal (canal)	Récupération de l'expression du calibre du canal (avec les unités)
ExpRes(canal, v, nbcs)	Renvoie l'expression de la grandeur de valeur v avec le symbole et l'unité, et le nombre de chiffres significatifs nbcs précisé.
NbPtsDansBlocAcq (nbPts1)	Définition du nombre de points des blocs d'acquisition. Cette fonction n'est utilisée que pour les acquisitions « longues » où on souhaite un affichage en temps quasi réel des résultats (pour ne pas s'impatienter...) Cette fonction renvoie la valeur retenue, i.e. éventuellement corrigée pour assurer la cohérence des réglages et pour contraintes techniques.
DureeBloc (DureeBloc1)	Définition de la durée des blocs d'acquisition. Renvoie la valeur retenue, i.e. éventuellement corrigée en fonction des contraintes techniques et pour assurer la cohérence des réglages (avec le nombre de points dans les blocs).
DureesEtPoints (dureeTotale1, nbPts1)	Sélection de la durée d'acquisition et du nombre de points souhaité pour une acquisition périodique. Les valeurs retenues peuvent être différentes des valeurs souhaitées du fait de contraintes techniques de la SYSAM CAMPUS. Ce processus détermine le temps d'échantillonnage Te. Vous ne pouvez pas le choisir...
SYNCHRONISATION DE L'ACQUISITION	
DefModeDeclenchement (modeDecl1)	Définition du mode de déclenchement. Trois possibilités : mdAucun : pas de synchronisation, l'acquisition démarre dès la commande (DeclencheAcqMonocoup ci-dessous) mdSeuil : Une fois déclenchée (avec DeclencheAcqMonocoup) l'acquisition ne démarre réellement qu'après le passage du signal de référence par une valeur donnée (à définir) dans un sens donné (à définir) mdExterne : Une fois déclenchée (avec DeclencheAcqMonocoup) l'acquisition ne démarre réellement que lorsque le signal de tension injecté dans l'entrée externe (Trig, borne jaune de SYSAM CAMPUS) passe par une valeur fixe positive (que j'ai oubliée et qui n'est pas réglable)
DefCanalSynchro (canal)	Définition du canal de synchronisation. Ce canal doit être activé préalablement pour assurer la cohérence des réglages. Si le canal choisi n'est pas actif, le processus sélectionnera le premier canal actif disponible.
DefDureePretrig (DureePretrig1)	Lorsque la durée du prétrig est supérieure à zéro, la centrale commence les acquisitions dès le déclenchement, pendant un temps minimal égal à la durée de ce prétrig, et seulement à ce moment-là, cherche à voir si les conditions de déclenchement sont réunies. La centrale renvoie donc aussi les valeurs précédentes le moment où ces conditions sont réunies et correspondant à la durée de prétrig choisie. Utile pour des expériences dont il faut voir l'instant initial (charge / décharge du condensateur par exemple)
DefNbPtsPretrig (NbPtsPretrig1)	Définition du nombre de points de préacquisition (prétrig). Confiné entre 0 et le nombre total de points à acquérir (NbPts).

DefTauxPretrig (TauxPretrig1)	Ce taux est en cohérence avec les notions de durée de prétrig et de nombre de points de prétrig. Définition du taux de préacquisition (pré-trig). Valeur en %, confinée entre 0 % et 100 % (de la durée d'acquisition)
DefSensDeclSeuil (sensDecl1)	Définition du sens de déclenchement sur seuil. Valeurs possibles : sdMontant ou sdDescendant. On choisit si les mesures doivent commencer lorsque le signal (ou la gp) passe par le seuil en augmentant ou en diminuant.
DefSeuilDeclenchement (ValSeuilDecl1)	Définition de la valeur de seuil (pour la synchro sur seuil).
DECLENCHEMENT DES ACQUISITIONS et RECUPERATION DES DONNEES	
DeclencheAcqMonocoup ()	Déclenche une acquisition monocoup, i.e. un cycle d'une acquisition périodique.
ArreteAcq ()	Arrête l'acquisition, que la récupération soit terminée ou non.
LireAcq()	Récupération de la totalité des points (en une fois), après la fin de l'acquisition. Explications supplémentaires dans le code des exemples.
LireBlocAcq ()	Lecture/récupération des valeurs acquises d'un bloc d'acquisition – Utiles uniquement pour les acquisitions longues où on souhaite voir les résultats en temps quasi réel. Fonction technique : ne pas modifier, sauf avec des connaissances approfondies de la programmation de la centrale SYSAM CAMPUS.

I. Fonctions de contrôle de SYSAM CAMPUS pour l'émission.

TOUT POUR L'EMISSION	
DeclencheSA (SA)	Déclenche l'émission cyclique sur la ou les sorties. Trois cas : SA = camp.SA1 : sur la sortie 1 de CAMPUS SA = camp.SA2 : sur la sortie 2 de CAMPUS SA = camp.SA1 + camp.SA2 = 3 : sur les deux sorties en même temps.
ArreteSA (SA)	Arrête l'émission sur la ou les sorties (même paramétrage que l'instruction précédente).
EmissionUContinue (SA, U)	Emission d'une tension continue U (entre -10 V et + 10 V) sur une sortie analogique
ArreteSAetAcq ()	Arrête l'émission sur toutes les sorties SA et l'acquisition en même temps.
CONSTRUCTION D'UN SIGNAL A EMETTRE SUR UNE SORTIE	
ParametreEmission (SA, DureeTotale1, NbVal1)	Choix des paramètres d'émission SA = sortie où « émettre », valeurs prédéfinies possibles : camp.SA1 ou camp.SA2 DureeTotale1 = durée totale d'un cycle NbVal1 = nombre de valeurs (de tension) du cycle.

	<p>Cette fonction détermine le temps d'échantillonnage retenu pour l'émission, en tenant compte des contraintes techniques de campus.</p> <p>Elle doit impérativement être appelée avant les deux instructions suivantes.</p>
PlaceValeur(SA, position, u)	<p>Préparation des valeurs du signal (en V). Ces valeurs sont stockées dans un tableau provisoire caché. Paramètres :</p> <p>SA : sortie concernée (camp.SA1 ou camp.SA2)</p> <p>position : position dans le tableau provisoire.</p> <p>u : valeur de tension à placer.</p>
TransmetValeurs(SA)	<p>Transmission du tableau des valeurs à la centrale.</p> <p>Cette fonction doit être appelée APRES la fin de la mise en place des valeurs et AVANT le déclenchement de l'émission.</p>

Procédure à suivre dans l'ordre pour créer un signal de toutes pièces :

1. Préparer les caractéristiques avec ParametreEmission.
Ce réglage prépare un tableau de valeurs avec le nombre de cases égal à NbVal1, et détermine le temps d'échantillonnage TeSA [SA] le plus approprié pour la durée d'émission DureesTotalesSA[SA] souhaitée en tenant compte des contraintes techniques de SYSAM CAMPUS. Cette durée totale peut être égale à la période du signal (ou à un nombre entier de périodes), car SYSAM CAMPUS émet les valeurs de manière cyclique.
2. Indiquer les valeurs du signal à émettre, une à une avec PlaceValeur ; il faut les calculer judicieusement (voir les exemples).
3. Transférer ces valeurs vers la RAM d'émission de la centrale SYSAM CAMPUS avec TransmetValeurs
4. Déclencher l'émission.

J. Cas particulier de l'émission de salves (d'ultrasons).

Les contraintes liées à l'échantillonnage et au faible nombre maximal de points (2000) de la RAM d'émission de SYSAM CAMPUS sont trop fortes pour utiliser directement un code accessible utilisant les étapes 1, 2 et 3 décrites juste au-dessus.

Nous proposons donc une fonction qui fait le travail de préparation en une fois de manière transparente :

camp.PrepareEmissionSalve

Paramètres:

camp.PrepareEmissionSalve(SA(sortie), freq (en Hz), periodicite (en s), amplitude (en V), taux (en%))

1. SA est la sortie utilisée (valeurs : camp.SA1 ou camp.SA2)
2. La fréquence est prioritaire : c'est le paramètre qui ne sera pas modifié, mais limité à 2.5 MHz eu égard au fait que la fréquence d'échantillonnage maximale de campus est de 10 MHz)
3. periodicite = intervalle de temps de répétition des salves

Les contraintes techniques imposent que cette valeur soit d'autant plus faible que la fréquence est élevée : pour une fréquence donnée, il y a une valeur maximale de la periodicite. La routine corrige donc la valeur en cas de dépassement. La valeur corrigée se trouve dans la variable globale camp.PeriododiciteSalves

4. L'amplitude maximale est de 10V (correspondant au calibre -10V/+10V des sorties de campus).
5. Le taux (en % entre 0 et 100) correspond à la fraction de la camp.PeriodiciteSalves pendant laquelle la salve est émise. La salve comporte un nombre entier de périodes (minimum une - mais est-encore une salve dans ce cas ?)

BON A SAVOIR (pour ne pas être surpris par la forme des courbes dans certains cas)

Le faible nombre de points (2000) de la RAM d'émission des sorties de campus limite les possibilités pour l'émission de salves et oblige à des choix contradictoires.

- a. Si on souhaite que la périodicité (temps séparant le début de l'émission de deux salves successives) soit aussi grande que possible, il faut accepter un faible nombre de points par période. Nous avons décidé que le minimum de ce nombre est de 4, afin que le signal, même en marche d'escalier, ressemble (de loin) à une sinusoïde.
Pour obtenir ce résultat, passer au paramètre « periodicite » une valeur élevée (par exemple 1 s), la valeur de périodicité qui en résulte sera automatiquement réduite à la plus grande valeur compatible avec la fréquence. Si la fréquence est élevée, ce sera assez fortement réduit...
- b. Si on souhaite un signal au caractère sinusoïdal plus marqué, il faut plus de points par période, mais en contrepartie, on réduit la périodicité maximale. Pour obtenir ce résultat, on passera comme paramètre « periodicite » une valeur plus faible que la periodicite maximale possible, par exemple 100 fois la période des salves (ex : avec des salves de 40 kHz, prendre 2.5 ms, ce qui fera 20 points par période-sinusoïde)

Quand on alimente un transducteur/émetteur d'ultrasons avec un signal comportant un faible nombre de points (au moins 4, mais pas beaucoup plus, donc « sinusoïde » avec des paliers), étant donné la bande passante très limitée de ces transducteurs, seul l'harmonique fondamental (parfaitement sinusoïdal) est émis sous forme d'onde ultrasonore, les ultrasons émis correspondants sont donc quand même sinusoïdaux (enfin, je suppose...) (le transducteur se comporte comme un filtre passe bande très étroit).

La synchronisation de salves est souvent difficile et/ou aléatoire : en général, comme chacun sait, on utilise des salves d'ultrasons pour mesurer la vitesse du son ou bien des distances entre émetteur et récepteur ou entre deux récepteurs. Dans tous les cas, il faut mesurer le décalage temporel entre le début de deux salves qui se correspondent. Or la synchronisation peut déclencher l'acquisition en plein milieu d'une salve. Pour pallier cet inconvénient, on pourra avantageusement remplacer la procédure

camp.PrepareEmissionSalve

par

camp.PrepareEmissionSalveAvecSignalSynchro (paramétrage inchangé)

Cette procédure fournit (en même temps que les salves sur SA) sur l'autre sortie de campus le signal enveloppe de la salve émise sur la sortie SA : à savoir un signal en créneau d'une valeur égale à l'amplitude tant qu'il y a émission, une valeur nulle pendant les « silences ». On peut se servir de ce signal pour la synchronisation : sur seuil, à condition de relier la sortie à une entrée disponible et de synchroniser sur cette entrée ; ou bien en synchro externe, en reliant la sortie à l'entrée spéciale « TRIG » de campus (des exemples sont disponibles).

K. Capteurs

CAPTEURS	
DetectionCapteur(canal)	Détecte le capteur actuellement connecté sur le canal.
DetectionCapteurs()	Détecte les capteurs actuellement connectés à la centrale sur les quatre canaux. Utilise la fonction précédente
AfficherModesDispoDeCapteur(canal)	Affiche les modes disponibles sur le capteur connecté sur le canal. Cette fonction sert à faire connaissance avec le capteur pour sélectionner en connaissance de cause le mode à utiliser pour l'expérience qu'on veut réaliser.
AfficherModesDispoCapteurs()	Affiche les modes disponibles pour tous les capteurs connectés sur les canaux de la centrale. Utilise la fonction précédente.

L. Affichage des résultats des acquisitions : matplotlib et pygame

On rappelle que les résultats des acquisitions sont stockés dans le tableau à deux entrées gpM, dans l'ordre réel des canaux. Cela peut suffire pour de nombreuses exploitations, que chacun devra programmer lui-même en fonction de ses objectifs.

Le projet SysamCampus.Py fournit des exemples montrant comment afficher les résultats sous forme de graphiques avec matplotlib ou pygame.

Cependant, je n'ai pas réussi à obtenir un temps réel satisfaisant lors d'acquisition un peu longues avec matplotlib (si quelqu'un qui s'y connaît a une solution ...). L'utilisation de pygame permet ce suivi en temps quasi-réel de l'affichage des courbes. Dans cette optique, le module sysam_campus_gr.py fournit quelques outils sommaires pour agrémenter l'affichage des résultats.

Deux parties :

1. Ce qu'il faut pour afficher les résultats des mesures uniques (obtenues par AcquisitionUnPointSurUnCanal et MesureDirecte) avec un aspect rappelant vaguement un contrôleur universel, dans une fenêtre.
2. Ce qu'il faut pour afficher un graphique en temps réel ou non (avec le temps en abscisse) pour des acquisitions qui durent un peu.

Lorsqu'on veut représenter une grandeur acquise en fonction d'une autre (sans intervention du paramètre temps), l'usage de matplotlib est suffisant. Dans l'état actuel des choses, le module sysam_campus_gr.py ne fait pas le travail, mais tout volontaire peut rajouter ce qu'il faut en s'inspirant de l'existant.

A découvrir dans les exemples.

Janvier 2026

Jean-Marie THOMAS
Professeur de physique-chimie retraité
Auteur du logiciel Oscillo5