

Oscilloscope

Cahier des charges détaillé de l'oscilloscope d'inspection pour carte STARE

Vue d'ensemble

Description générale

A partir de données d'acquisition fournies en continu par un serveur en python, une application web en utilisant le cadriciel Django propose aux utilisateurs des fonctionnalités d'affichage de courbes à la façon d'un oscilloscope :

- sélections en temps et en amplitude,
- options d'affichage,
- transformations mathématiques.

Les langages utilisés sont du Javascript côté navigateur et du Python côté serveur

Contexte de l'expérience :

AGATA (Advanced Gamma Tracking Array) est un projet de recherche européen dont le but est de développer et de construire un spectromètre à rayons gamma 4pi de nouvelle génération. Il sera utilisé dans des expériences utilisant à la fois des faisceaux d'ions intenses, stables et radioactifs, pour étudier la structure des noyaux atomiques en fonction du moment cinétique, de l'isospin et de la température aux limites de leur stabilité. Le spectromètre AGATA complet sera constitué d'un réseau de 180 grands cristaux de germanium de haute pureté (HPGe) encapsulés (9 cm de longueur, 8 cm de diamètre circulaire). Les cristaux sont effilés de forme hexaonique (hexagonale à l'avant, circulaire à l'arrière) et ils sont segmentés électriquement en 36 segments (6 longitudinaux, 6 transversaux).

Cahier des charges fonctionnel

Ce document complète le Cahier des charges fonctionnel [Oscilloscope Software - Specifications](#) (Nécessite d'être logué dans [gitlab.in2p3.fr](#)) écrit par E. Clément, A. Gadea, O. Stézowski, qui rappelle l'objectif d'afficher des flux de données provenant de cartes STARE sous forme de courbes d'Oscilloscope dans un navigateur web. Le logiciel [Perytech Oscilloscope Software](#) est pris comme modèle.

Lien avec le programme de contrôle commande

Le programme s'appuie sur l'interface de contrôle commande ([Slow Control](#)) Il utilise en particulier les fonctionnalités de sélection des cristaux de germanium pilotés par les cartes STARE et de gestion des droits d'accès d'utilisateurs. L'utilisateur après avoir choisi une sélection de cristaux puis de signaux peut lancer l'application Oscilloscope.

Architecture logicielle

Une Application Web Django depuis un navigateur se connecte sur un serveur en Python qui pilote des cartes électronique STARE par réseau :

- protocole [IPbus/IP](#) pour le contrôle commande
- UDP/IP pour les données

Une fonction Javascript de la page web récupère les données par requête HTTP depuis le serveur Python


Coté Javascript	côté python
XMLHttpRequest.send()	récupération des données par UDP
	retour d'une HttpResponse
Dessin dans un canvas HTML	

Une première requête récupère les données générales :

- nb de voies,
- fréquence d'échantillonnage, (fixée à 100 Mhz dans un premier temps),
- nb d'échantillons des traces,
- largeur de l'intervalle des tensions (range voltage) (valeur fixe)
- nb de bits des valeurs (entre 14 et 64)

Le dessin des courbes est effectué en Javascript par la page web. La partie contenant les courbes est un canvas HTML. Les contrôles de la partie droite sont en HTML et leur apparence est fixée en CSS.

En l'absence de carte STARE, celle-ci est remplacée pour un autre programme en python qui répond aux requêtes UDP.

 Il pourra également lire les commandes normalement envoyées par IPBus aux cartes sur un faux serveur IPbus (DummyHardwareUdp.exe). Une acquisition se lancera en mettant une valeur non nulle dans le paramètre N_SLV_aC_PAG.package_trig.CALLautoTime (C'est aussi ce que fait la fonction PACE_QuadLink_start 1 timerDat) et se stoppe en mettant cette valeur à 0

Faisabilité

Tester le nb de dessins par seconde possibles
Tester en regroupant les requêtes pour diminuer le nb d'allers-retours
Tester avec le serveur Django sur une machine différente

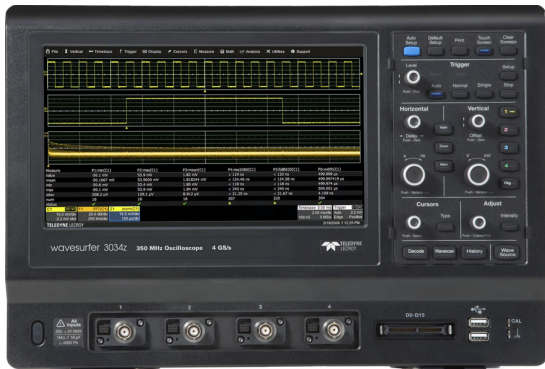
Fonctionnalités

 indique les fonctionnalités supplémentaires à faire dans un deuxième temps.

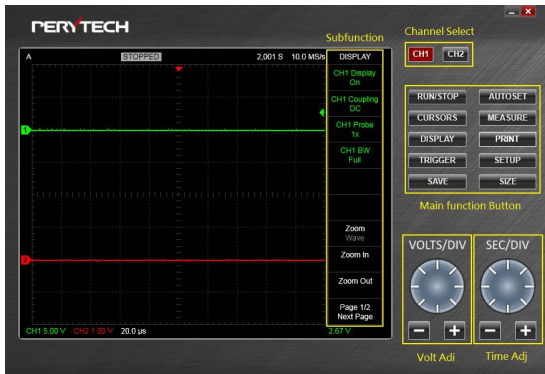
Ecran de l'Oscilloscope

Exemples d'IHM

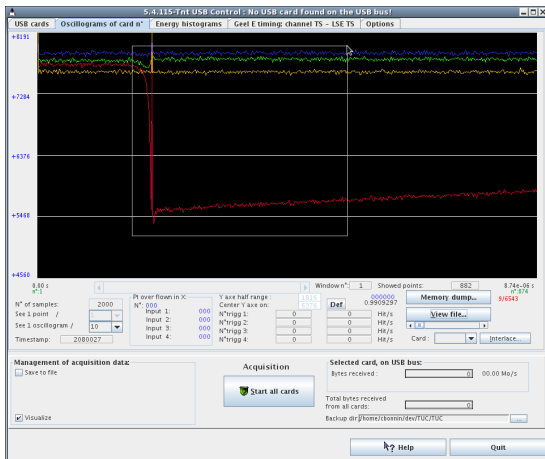
- Oscilloscope Lecroy



- Interface graphique du boîtier d'acquisition Perytech



- Logiciel lourd de visualisation d'oscillogrammes TUC



- Exemple d'oscilloscope en Javascript utilisant le micro comme signal d'entrée [Virtual Oscilloscope](#)

Le modèle sera l'interface de Perytech avec 10 boutons de sélection des voies, sans les boutons + et -, avec la partie Subfunction invisible par défaut

Sélection des voies

Le nombre de voies sélectionnables est donné par le nombre de voies demandées depuis l'écran de sélection des voies oscillo ([exemple](#))

- Un clic sur bouton de voie le sélectionne (bordure plus épaisse), sa voie s'affiche si ce n'était pas le cas
- Un seul bouton est sélectionné à la fois
- Un deuxième clic sur le bouton sélectionné dont la voie est affichée, la cache.

- Les boutons des voies affichées sont en couleur, les autres en gris

Zoom

Vertical

Le bouton Volts/Division permet de régler le zoom vertical. Ce zoom est individuel pour chaque voie. Un clic simple remet la valeur par défaut. Un glisser-déplacer démarré sur le bouton le fait tourner dans le sens du déplacement en fonction de la position de début du clic par rapport au centre. L'image du bouton pivote pour visualiser le sens de rotation et réglage de la valeur. Le nombre de Volts par division est écrit en bas de l'écran dans la couleur de chaque voie

Horizontal

Le bouton Secondes/Division permet de régler le zoom horizontal, valable pour toutes les voies. Il fonctionne de la même manière que le vertical. La durée d'une division est écrit en blanc en bas de l'écran (entre 1 μ s et 1s).


Sélection d'une zone à la souris

Il est aussi possible de sélectionner une zone dans l'écran à l'aide du pointeur. Un rectangle blanc visualise la zone en cours de sélection comme dans TUC. Les dimensions du rectangle sont affichées (hauteur en Volts, T=largeur en Secondes, Fréquence = 1/T en Hz)

Unzoom

La sous fonction Unzoom accessible par le bouton Display remet le zoom par défaut.

Molette

 zoom vertical avec la molette en fonction de la position du pointeur et zoom vertical avec la molette et la touche Majuscule enfoncée (Shift)

Déplacements

Verticaux

A gauche de l'écran un repère permet d'effectuer à la souris un déplacement vertical (offset) pour chaque voie.

Horizontaux

En haut au centre de l'écran, un repère triangulaire permet déplacer horizontalement (delay) l'ensemble des voies.

Affichage

- Un quadrillage discret (gris) permet d'évaluer les grandeurs. Il peut être retiré par le menu Display

- Par défaut les couleurs des voies sont les suivantes :
 - Sur fond noir : `lime red cyan yellow green orange pink blue fuchsia white`
 - Sur fond blanc : `green, red, gray, olive, green, orange, maroon, blue, purple, black`



Possibilité pour l'utilisateur de modifier les couleurs.



Les préférences d'affichage sont conservées pour chaque utilisateur

Marche / Arrêt

Le bouton Run / Stop arrête ou reprend le défilement des courbes

Impression

Le bouton Print génère une image au format PNG et la télécharge par le navigateur

Exportation

Le bouton Save exporte les données au format choisi :

- format CSV avec autant de colonnes que de voies, un entête, et une ligne par point échantillonné.
- format PNG
- format JPEG
- image dans presse-papier

Le résultat est téléchargé

Réglage automatique

Le bouton Autoset effectue un réglage automatique du zoom vertical pour chaque voie de sorte que toutes les valeurs soient sur l'écran. Il tente de déterminer la fréquence de la première voie affichée et s'arrange pour afficher 5 périodes sur l'écran

Fonctions mathématiques

(5.3.8. Wave pattern calculation function) Un sous-menu permet de choisir

- une voie non affectée.
- Une première voie opérande parmi celle dont le numéro est inférieur pour éviter les boucles
- Une opération (+, -, *, /, ², dérivée, intégration)
- Une deuxième voie si l'opération a 2 opérandes
- Possibilité d'effacer la voie
- Possibilité de déplacer vers un autre numéro de voie compatible avec les opérandes

Autres fonctions

Fonctions décrites dans le cahier des charges fonctionnel :

- Curseur

- Horizontal pour toutes les voies
- Vertical pour la voie sélectionnée
- Vue générale (5.3.9. Overview function)
- Mesures automatiques (5.5. Automatic Measurement Function)
- Changement de taille de l'écran (5.9. Adjust Screen Size)

Simulateur de carte STARE

Nom: fakeStare.py

Arguments

- size : Taille des paquets de données en octets
- delay : Durée entre deux envois (en ms)
- file : chemin du fichier où lire les données
- ip : adresse ipv4 où envoyer les données
- port : Numéro de port
- repeat : Nombre d'envois de paquets (0=illimité par défaut)
- channels : Nombre de voies (1 par défaut)
- group : 💡 Nombre de paquets regroupés dans un envoi (supplémentaire, 1 par défaut)

Formats de fichiers de données

- .dat : données brutes envoyées directement
- .osc : fichiers oscillogrammes [Format](#)

Fonctionnalités supplémentaires

💡 Lecture régulière (1 / s) du paramètre IPBus N_SLV_aC_PAG.package_trig.CALLautoTime pour modifier la taille des envois ou les stopper (si 0)

Logiciel

Outils

Le logiciel est développé en python avec Django. Les données sont stockées dans une base de données MySQL. Une [instance de démonstration](#) est hébergée par l'IPHC (mot de passe par défaut : agataguest).

Code Source

Dépôt des sources git : `git@gitlab.in2p3.fr:cbonnin/starecontrol.git` .

Installation de développement

Utiliser la branche git "opichot" dérivée de la branche master.

Environnement de développement (hors base de données)

Le serveur Django contenant les applications slowControl et oscillo doit être lancé dans un conteneur ayant tous les paquets et modules python nécessaires déjà installés

```
git clone -b opichot git@gitlab.in2p3.fr:cbonnin/starecontrol.git
docker/dockerRun.sh
```

Depuis un autre terminal, copier les sources dans le conteneur avec

```
docker/dockerCp.sh
```

Dans le conteneur, lancer le serveur Django

```
./runserver.sh
```

L'application est maintenant utilisable dans un navigateur à l'adresse <http://localhost:8000>

Les comptes admin (mot de passe : admin) et guest (agataguest) sont présents dans la base

Premières étapes du développement

1. Affichage d'une courbe fixe
2. Animation de la courbe
3. Affichage de plusieurs voies
4. Déplacement vertical d'une courbe
5. Déplacement horizontal des courbes
6. Zoom vertical
7. Zoom horizontal
8. ...

Tests automatiques

Les fonctions du serveur python peuvent être testées automatiquement. Pour cela les fonctionnalités de test de Django sont utilisées : `python manage.py test`

Licence

<http://www.cecill.info/>