h e p i a
Haute école du paysage, d'ingénierie
et d'architecture de Genève

Hes·so
Haute Ecole Spécialisée
de Suisse occidentale

Affichage calorimètre

Rapport du WP 5 dans le cadre du projet Calind

Assistant : Schaub Lionel

Professeur : Gérald Litzistorf

En collaboration avec : CHUV

Table des matières

1. Executive Summary

Les objectifs du WP5.1 ont été atteints : affichage temps réel, mode temporel ou respiratoire, décalages paramétrables entre les courbes, estimation de la période respiratoire, sauvegarde automatique, relecture d'une acquisition et prise en compte des exigences cliniques.

Chronologie du WP 5.1 :

- *Étape 1: Phase initiale de test effectuée à partir de données brutes fournies par le Dr. Cotting que Lionel Schaub a utilisées pour la simulation des signaux CO_2, O_2, Flux. Voir figure 1.*
- *Étape 2: Acquisition de données série-USB depuis la carte de développement prêtée par VS.*
- *Étape 3: Adaptation au matériel FR (flux O_2) et validation sur le banc de test CHUV. Voir figure 2.*
- *Étape 4: Adaptation (non-terminée) du logiciel pour fonctionner avec la nouvelle carte d'acquisition pour les paramètres CO_2, Flux.*

Tous les livrables se trouvent sur http://www.tdeig.ch/projet/calind.

*La suppression WP4 (VS) a représenté un **supplément de travail** à ce WP : alors que tous les flux devaient être concentrés sur un seul port USB, Lionel Schaub a dû modifier l'architecture du logiciel pour permettre la prise en compte de N flux USB.*

Le WP5.2 comprend deux parties:

- *Les dossiers patients du CHUV sont gérés par le système Metavision avec une granularité de l'ordre de la minute. L'intégration du projet Calind dans Metavision exige un développement (payant) qui ne peut être effectué que par la société Imd-Soft (http://www.imd-soft.com). Elle n'a de sens qu'avec un WP5.1 totalement terminé (CO_2, O_2, Flux).*
- *Le service de soins intensifs de pédiatrie du Dr Cotting utilise un réseau Phillips dans lequel nous aurions aimé utiliser les affichages. Cette variante semble logique économiquement mais se heurte à des normes d'interconnexion propriétaires (non-publiques) Phillips et à un contrat Phillips-CHUV qui interdit à ce personnel de brancher un équipement non-Phillips.*

gerald.litzistorf@hesge.ch

2. Présentation

Le but de ce projet est de réaliser un logiciel capable d'afficher, sous forme de graphiques, des données en provenance de capteurs médicaux développés par la HES-SO et l'unité de soins intensifs de pédiatrie du CHUV.

Ces données vont ensuite servir pour mesurer la dépense énergétique du corps humain par calorimétrie indirecte.

L'objectif principal de ce logiciel est de faciliter la validation des capteurs.

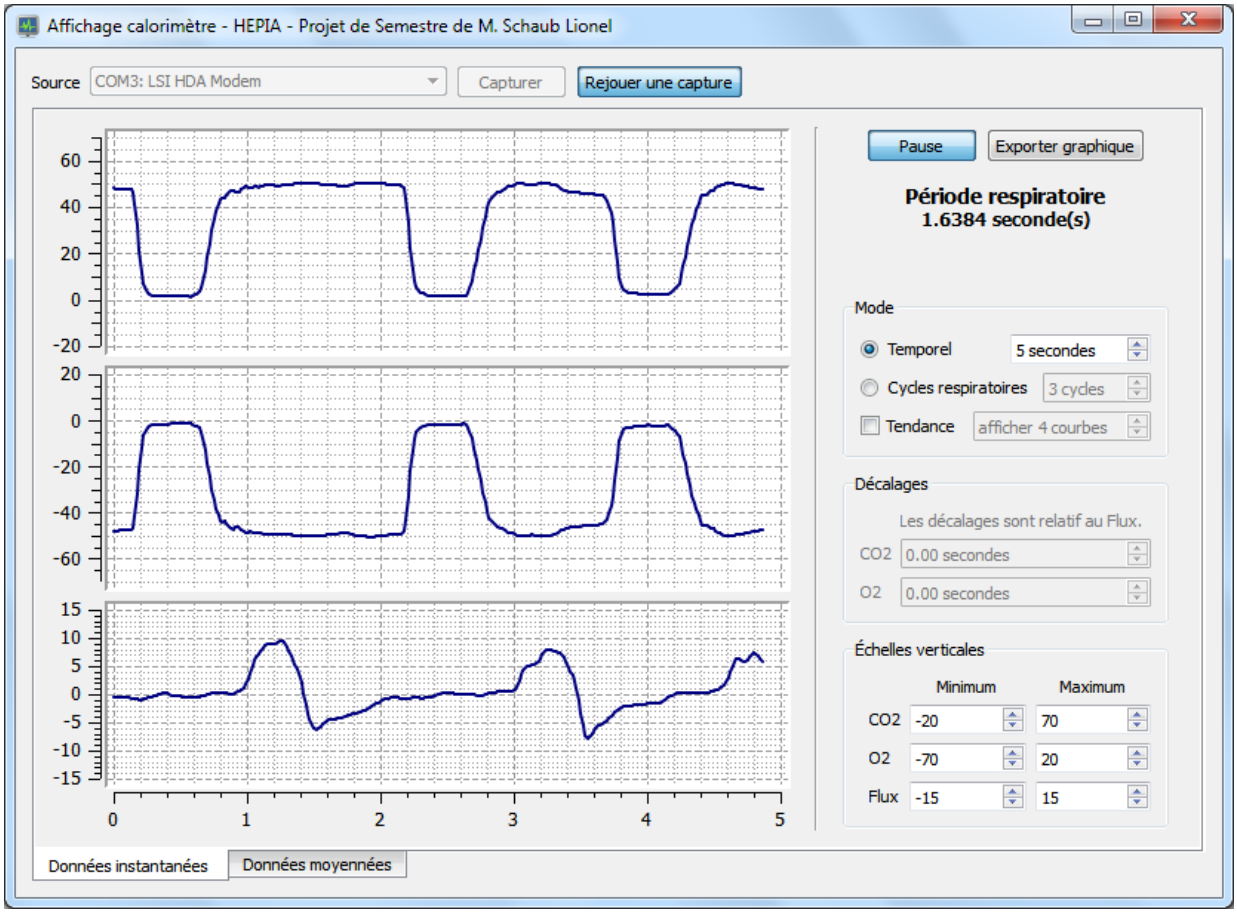


Figure 1

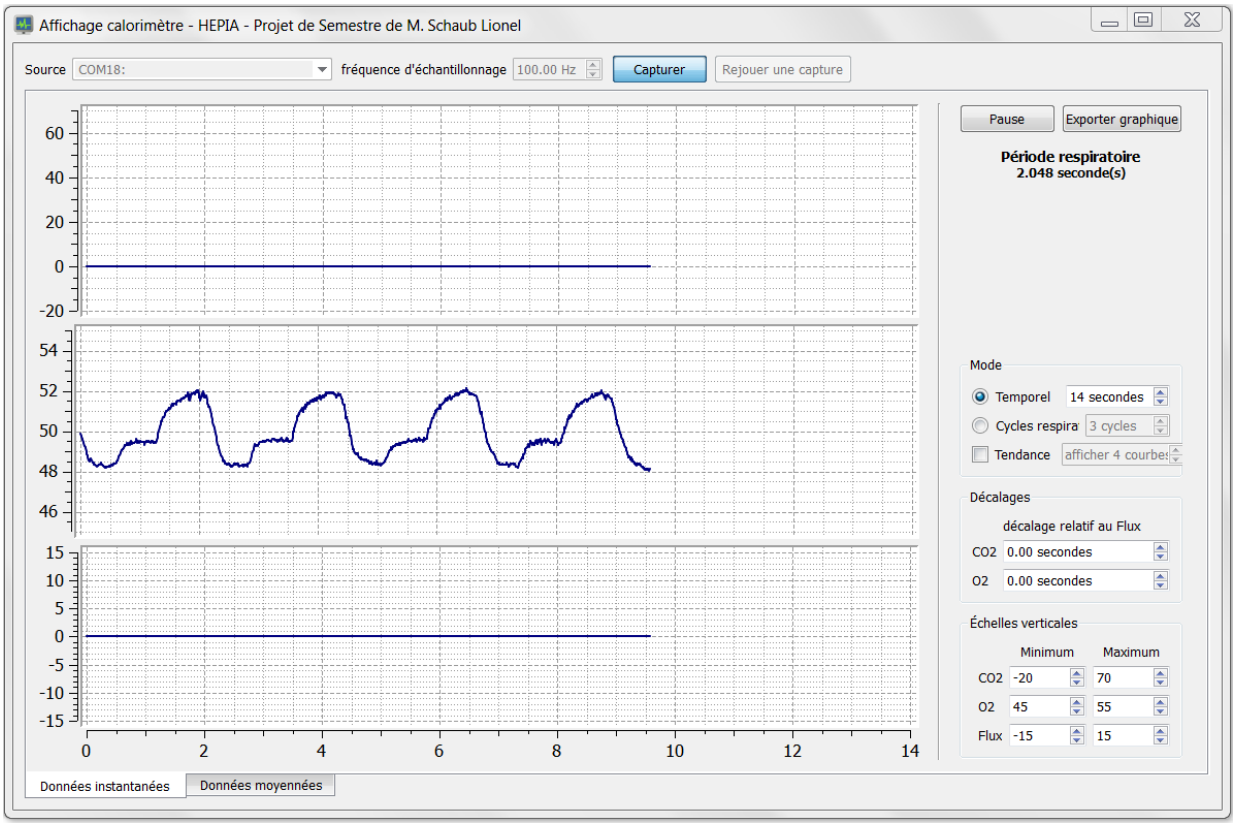


Figure 2

3. Identification des besoins

Pour commencer, il est nécessaire de bien comprendre les besoins des médecins. C'est pour cela que plusieurs rendez-vous aux CHUV ont été organisés.

Il est également essentiel de connaître l'interface et le protocole de communication matériel-logiciel. La HES-SO Valais qui s'occupe de la partie électronique a mis à disposition une carte de développement simulant l'envoi des données du capteur via un câble USB.

Avec ces informations, il a été possible de déterminer le cahier des charges et de choisir les outils appropriés pour le développement du logiciel.

4. Cahier des charges

Réaliser un logiciel s'exécutant sous Microsoft® Windows permettant d'afficher sous formes de courbes les trois valeurs en provenance d'un appareil médical connecté à l'ordinateur via un câble USB.

Les capteurs envoient les données à une vitesse comprise entre 100Hz et 200Hz.
La liaison USB est de type Serial CDC (émulation série).

Un graphique par courbe.
Échelle horizontale commune.
Fonction permettant de geler l'affichage.
Utiliser le maximum de l'espace vertical pour afficher les graphiques.

Deux modes de réglage de l'échelle horizontale : nombre de secondes et nombre de cycles respiratoires.
Fonction de décalage horizontal des signaux pour compenser les différents délais des capteurs.

5. Choix des outils

Comme les outils n'étaient pas imposés, il a été nécessaire d'analyser et de comparer différents outils pour choisir les plus appropriés.

Les seules contraintes étaient le système d'exploitation Microsoft® Windows et le type de liaison matériel-logiciel.

*Le Framework **Qt** de Nokia® a été retenu pour son API C++ multiplateforme de haut niveau qui simplifie grandement l'écriture du code, ainsi que pour ses divers outils (designer de fenêtre, etc.).*

*Qt ne proposant pas nativement de système de gestion de graphique, la bibliothèque **Qwt** a été intégrée au projet. Qwt est une bibliothèque développée avec Qt qui permet d'afficher des graphiques en 2D.*

*La bibliothèque **FFTReal**, permettant d'effectuer des transformées de Fourier, est utilisée pour déterminer la période respiratoire.*

*En ce qui concerne la communication avec les capteurs, la liaison USB est uniquement utilisée pour encapsuler une liaison série. Il existe deux bibliothèques réalisées avec Qt pour gérer une liaison série. Le choix s'est porté sur la bibliothèque **QSerialDevice** qui est stable et mature.*

6. Étapes de réalisation

Principales étapes du projet :

1. *Choisir les outils et créer un logiciel de démonstration permettant de tester les outils choisis et de donner un aperçu des possibilités aux médecins du CHUV.*
2. *Présentation du logiciel au CHUV et détermination, avec les médecins, des spécifications du logiciel final.*
3. *Création du nouveau logiciel (Affichage Calorimètre). Réalisation de l'affichage de courbes fictives.*
4. *Optimisations du logiciel pour qu'il fonctionne avec des données qui arrivent toutes les 5ms (200Hz).*
5. *Déplacement au CHUV pour récupérer des données réelles.*
6. *Implémentation de la bibliothèque de calcul de la transformée de Fourier. Validation avec Matlab®.*
7. *Implémentation des différentes options de configuration des graphiques.*
8. *Développement d'un logiciel (RS232 Reader) qui capture les données en provenance du matériel.*
9. *Intégration de la capture des données au logiciel Affichage Calorimètre.*

7. Fonctionnement du logiciel

Pour expliquer de manière plus simple et plus claire le fonctionnement de ce logiciel, il sera divisé en 3 parties :

7.1. Acquisition et sauvegarde des données

L'acquisition des données s'effectue via une connexion USB en mode Bulk sur un profil Serial CDC.

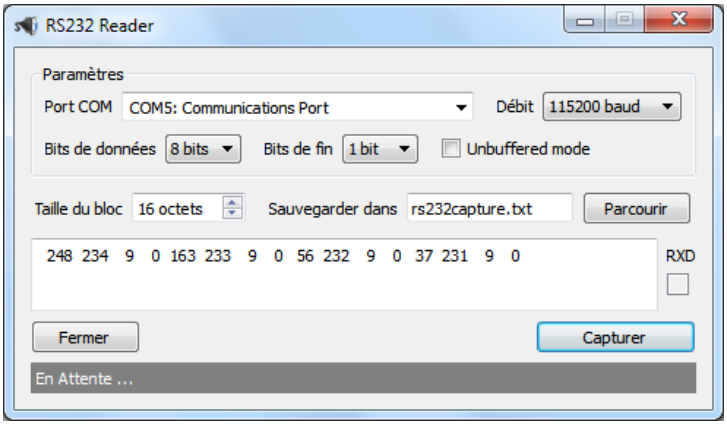
Le profil Serial CDC (Communication Device Class) est un profil qui permet d'encapsuler une liaison série dans une liaison USB. Ceci permet de continuer d'utiliser des liaisons séries malgré la disparition progressive des ports rs232

sur nos ordinateurs. Vu que cette liaison série n'est que virtuelle, elle ne souffre pas de certaines contraintes des liaisons séries physiques, tel que le choix de la vitesse de transmission, le nombre de bits de donnée, etc.

Le pilote Serial CDC est inclus dans le système d'exploitation. Par conséquent, au niveau du logiciel, seule la liaison série sera visible.

16 octets sont transmis toutes les 5 millisecondes soit une fréquence de 200 Hz. Parmi ces 16 octets il y a 4 valeurs de 4 octets (32 bits) chacune.

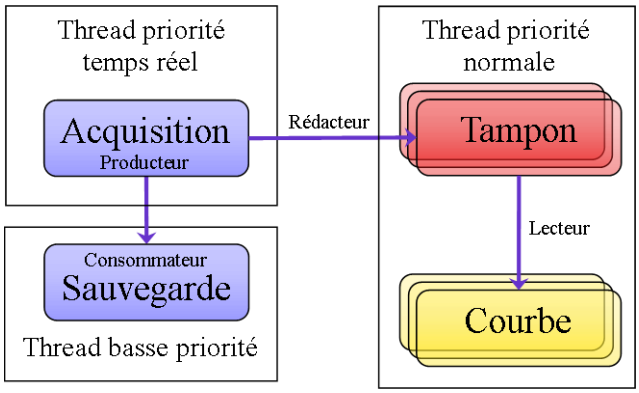
Pour tester le bon fonctionnement de la liaison série ainsi que de la bibliothèque QSerialDevice, un logiciel de test a été créé (cf. capture ci-contre). Ce logiciel a également été testé avec une liaison série physique pour valider les fonctionnalités qui n'existent pas dans une liaison série virtuelle via USB CDC.



Cette acquisition est effectuée dans un thread de haute priorité pour éviter qu'un autre thread, utilisant trop de ressources systèmes, retarde l'acquisition.

Chaque fois qu'une donnée est acquise, elle est transmise dans un tampon (pour être affichée sur le graphique). Cette donnée est également sauvegardée dans un fichier au format csv. L'écriture s'effectue à la fin de l'acquisition pour éviter de ralentir l'acquisition à cause des latences du disque.

La communication entre le thread d'acquisition et celui contenant le tampon utilise un modèle lecteur-rédacteur tandis que la communication entre le thread d'acquisition et celui de sauvegarde utilise un modèle producteur-consommateur.



La classe d'acquisition est également capable d'acquérir les données depuis un fichier csv. Cette fonction permet de rejouer une précédente acquisition.

Les méthodes d'acquisitions ont changées plusieurs fois au cours du projet. L'annexe Changement intervenus au cours du projet donne plus de détails sur les différentes méthodes utilisées.

7.2. Interface graphique et affichage des données

L'interface graphique se compose d'une seule fenêtre qui contient deux onglets pour les deux modes principaux d'affichage.

7.2.1. Mode instantané

Dans ce mode, les données des trois capteurs (O_2, CO_2 et Flux) sont affichées au fur et à mesure de leur réception. L'échelle horizontale peut être réglée, soit pour afficher un certain nombre de secondes, soit pour afficher un certain nombre de cycles respiratoires. Dans ce dernier mode, il faut être capable de détecter instantanément le début de chaque nouvelle période pour connaitre le moment où il faut effacer le graphique et dessiner une nouvelle courbe. Pour cela nous utilisons la transformée de Fourier ainsi que l'intégration du signal [1].
Une option permet de conserver à l'écran un certain nombre de courbes précédentes, ceci permet de voir la tendance.
[1]Comme les capteurs O_2 et CO_2 ont des délais plus longs que le capteur de flux, il y a deux champs qui permettent de décaler dans le temps les valeurs des courbes O2 et CO2 pour les synchroniser au flux.

7.2.2. Mode mesure

Ce mode affiche des données calculées (fréquence respiratoire, VO_2, VCO_2, etc.) à partir des données des capteurs.

Il était prévu de développer ce mode à la fin du projet si le temps le permettait. Ce mode n'a pas été développé.

7.3. Calculs

Plusieurs calculs doivent être effectués tout au long de l'acquisition, dont certains à chaque réception d'une donnée.

7.3.1. Fréquence respiratoire

Pour calculer la fréquence respiratoire plusieurs méthodes ont été envisagées :

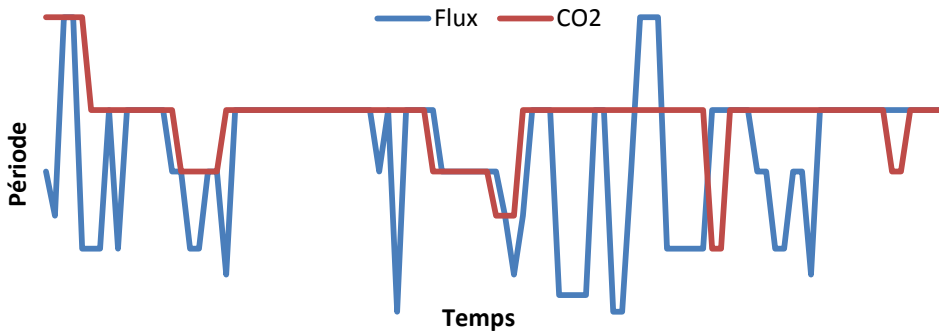
- *Détecter les passages par zéro (tous les deux passages à zéro = une période).*
- *Détecter les changements de flancs (montant <-> descendant).*
- *Utiliser la transformée de Fourier pour trouver la fréquence fondamentale.*

La première méthode peut poser problème si le signal subit un décalage vertical car il pourrait ne plus passer par zéro.

Les deux premières méthodes ne fonctionneront pas si le signal n'est pas parfait. En effet, si une petite perturbation ou un changement de flancs local se produit, la période serait faussée.

Il faut donc utiliser la troisième méthode qui ne souffre pas des problèmes ci-dessus.
La transformée de Fourier est effectuée grâce à la bibliothèque FFTReal.

Après avoir effectué des tests (cf. graphique ci-dessous), il s'avère que la détection de la période donne de meilleurs résultats si on utilise le signal CO_2 ou O_2 plutôt que le signal Flux. Le calcul est donc effectué sur le signal CO_2. Le signal CO_2 est mieux adapté à ce calcul car c'est un signal carré et, par conséquent, la variation de niveau est plus marquée.



7.3.2. Détection du début d'une période

La détection du début d'une période respiratoire est réalisée en utilisant conjointement le résultat de la transformée de Fourier ainsi que le résultat de l'intégration du signal CO_2 [1].
[1]Dès que le taux d'augmentation de la valeur de l'intégration du signal devient inférieur à un certain seuil, une nouvelle période est détectée. [1]Ensuite, le temps passé depuis la dernière détection est comparé au résultat de la transformée de Fourier pour vérifier que la détection n'est pas un faux-positif.

7.3.3. Volume O_2 et Volume CO_2

Pour calculer les valeurs VO_2 et VCO_2, il faut effectuer l'intégration des données des capteurs O_2 et CO_2. Pour cela il faut détecter en temps le début d'un cycle respiratoire. Cette détection ne peut pas se faire avec une transformée de Fourier car cette dernière ne permet pas d'effectuer une détection instantanée. Une autre méthode, détectant les passages proches de zéro et les changements de flancs, a donc été imaginée. Cette méthode n'a pas été implémentée suite aux changements intervenus à la fin du projet.

8. Notions temporelles et optimisation

Comme ce programme acquière, traite et affiche des données à une fréquence relativement élevée (~200 Hz), il est vital que toutes les opérations soient effectuées en moins de 5 millisecondes.

Même avec un ordinateur récent, ces opérations (acquisition, transformée de Fourier, dessin du graphique, etc.) ne sont pas réalisables en si peu de temps, il faut donc trouver des méthodes pour les optimiser.

Il n'est pas nécessaire d'actualiser la courbe à chaque réception d'une donnée car l'œil humain a une fréquence d'environ 25 Hz. La courbe est par conséquent actualisée toute les 25 ms (40 Hz) pour être certain que l'animation sera fluide.

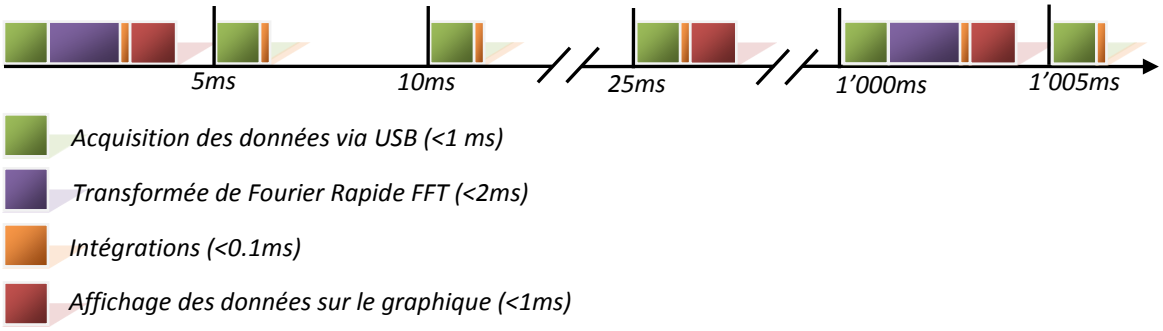
Par défaut Qwt redessine tout le graphique à chaque ajout de point. On peut donc considérablement améliorer la vitesse de dessin en ne traçant que les points ajoutés depuis la dernière actualisation de la courbe.

La transformée de Fourier est un calcul dont la complexité s'exprime en $O(n^2)$. Cet algorithme est donc particulièrement lent et coûteux en ressources. Il existe cependant un algorithme de calcul rapide de la transformée de Fourier appelé FFT (Fast Fourier Transform) dont la complexité s'exprime en $O(n \cdot log(n))$. Grâce à cet algorithme, il est possible d'effectuer la transformée en un temps beaucoup plus court (exemple : 340 fois plus court pour n=1024 données).

De plus, il n'est pas nécessaire d'effectuer ce calcul à chaque réception de données. Il est donc raisonnable d'effectuer ce calcul environ une fois par seconde.

Pour éviter de manquer des données, l'acquisition est effectuée dans un thread avec une priorité élevée. En effet, si les traitements prennent trop de temps, le programme n'arrivera pas à temps pour récupérer les données. Cependant, l'ajout d'un thread nécessite de gérer les communications inter-thread en ajoutant un modèle de lecteurs rédacteurs.

Vous trouverez ci-dessous un chronogramme représentant les différentes étapes exécutées par le programme à chaque réception de données.



9. Problèmes rencontrés

Durant la réalisation de ce projet, il y a eu trois problèmes majeurs :

1. *Respect des contraintes de temps.*
   Au début du projet, il était prévu d'acquérir 1 point par minute. Par la suite, les spécifications ont changé et il fallait 200 points par seconde, soit un facteur de 12'000. L'ordinateur utilisait 100% des capacités du processeur et n'arrivait pas à afficher les données à temps. Il a donc été nécessaire de trouver plusieurs méthodes pour optimiser le temps de traitement.

2. *Décalages des octets lors de l'acquisition des données via la liaison série.*
   Peu après le début de l'acquisition, les 4 octets qui composent le nombre sur 32 bits subissaient un décalage circulaire suite à la perte d'un octet. Par conséquent, l'intégrité du nombre était perdue. Le problème a été résolu en activant le buffer du port série. Ceci permet d'éviter la perte d'un octet et par conséquent d'éviter le décalage.

3. *Fonction de décalage des signaux en temps réel*
   Le décalage des signaux O_2 et CO_2 en fonction du signal est un problème non-trivial. Il faut adapter et décaler les données dans les buffers tout en recevant des nouvelles données.

4. *Intégration et utilisation de la bibliothèque UniversalLibrary au projet*
   La bibliothèque UniversalLibrary est prévue pour fonctionner avec le compilateur de Microsoft et non avec le compilateur MinGW utilisé pour ce projet. Il a fallu trouver un moyen pour convertir la bibliothèque dans un format adapté.

10. Mesures de performances

Les mesures de performance ont été effectuées sur un ordinateur équipé d'un processeur Intel Core2 Duo T8300 2.4 GHz. Pour ce test, un seul cœur a été utilisé et la fréquence du processeur a été limitée à 500 MHz (20%).

Après une heure d'exécution, l'application n'a jamais utilisé plus de 5% de la capacité du processeur.
Son empreinte mémoire est d'environ 30 Mo.

11. Conclusion

On constate qu'un simple programme d'acquisition et d'affichage de graphiques cache un certain nombre de difficultés tel que le traitement du signal en temps réel, l'optimisation et le respect des contraintes de temps.

La fonction de décalage des signaux en temps réel est également un problème non-trivial.

A. Annexes

A.1. Codes sources et autres fichiers

Les codes sources et les exécutables sont disponibles sur le site web du Laboratoire de transmission de données hepia à l'adresse http://www.tdeig.ch/projet/calind.

A.2. Auteurs et Licences

***Qt** : Nokia, LGPL*
***Qwt** :Uwe Rathmann, LGPL*
***FFTReal** : Laurent de Soras, WTFPL*
***QSerialDevice**:Denis Shienkov , GPL*

A.3. Liens utiles

Voici une liste de liens utiles lors de développements avec Qt.

Site officiel de Qt : http://qt.nokia.com/
Documentation Qt 4.7 : http://doc.qt.nokia.com/4.7/index.html
Documentation Qwt : http://qwt.sourceforge.net/classes.html
Tutoriel sur Qt : http://www.siteduzero.com/tutoriel-3-11406-apprenez-a-programmer-en-c.html#part_11407
Section Qt sur Developpez : http://qt.developpez.com/
Faq Qt sur Developpez : http://qt.developpez.com/faq/
Liste des outils (bibliothèques, EDI, …) pour Qt sur Developpez : http://qt.developpez.com/outils/

A.4. Changements intervenus au cours du projet

Ce projet a subi quelques changements de spécifications au cours de son avancement.

Au départ, les données devaient être acquises depuis une carte développée par la HES-SO Valais. Cette carte devait réunir et convertir les signaux en provenance des différents capteurs puis les transmettre au PC grâce à une émulation série sur USB (USB CDC Serial).

Puis les données d'oxygène devaient être acquises directement depuis la carte de Fribourg qui implémente également une émulation série sur USB. Le logiciel a donc été modifié en conséquence.

Vers la fin du projet les spécifications ont changées et une nouvelle carte d'acquisition a été utilisée. Cette carte devait être utilisée pour effectuer l'acquisition des données de CO_2 et de Flux. Cette nouvelle carte n'implémente pas l'émulation série, il faut utiliser leur bibliothèque UniversalLibrary. Le logiciel a donc été modifié pour permettre l'acquisition des signaux de CO_2 et de Flux via cette carte tout en effectuant l'acquisition du signal de O_2 via la carte de Fribourg.

Un dernier changement est survenu dans le projet avant que l'étape ci-dessus ne soit terminée. La répartition de charge a été changée et le développement de la partie calcul du logiciel a été transféré à la HEIG-VD.

A.5. Visibilité du projet

En février 2012, j'ai été contacté par un ancien étudiant de l'Institut National Polytechnique de Grenoble. Il était intéressé par le développement effectué pour Calind car il travaillait sur un projet de domotique et devait développer un système similaire. Son application devait afficher (avec Qt) des données en temps réel acquises via des liaisons séries, puis les sauvegarder.

Je lui ai donc transmis les sources du logiciel RS232Reader pour l'aider à développer ses connexions séries.

On remarque donc que le site web www.tdeig.ch a permis de fournir une bonne visibilité au projet.