

Oxymètre de pouls

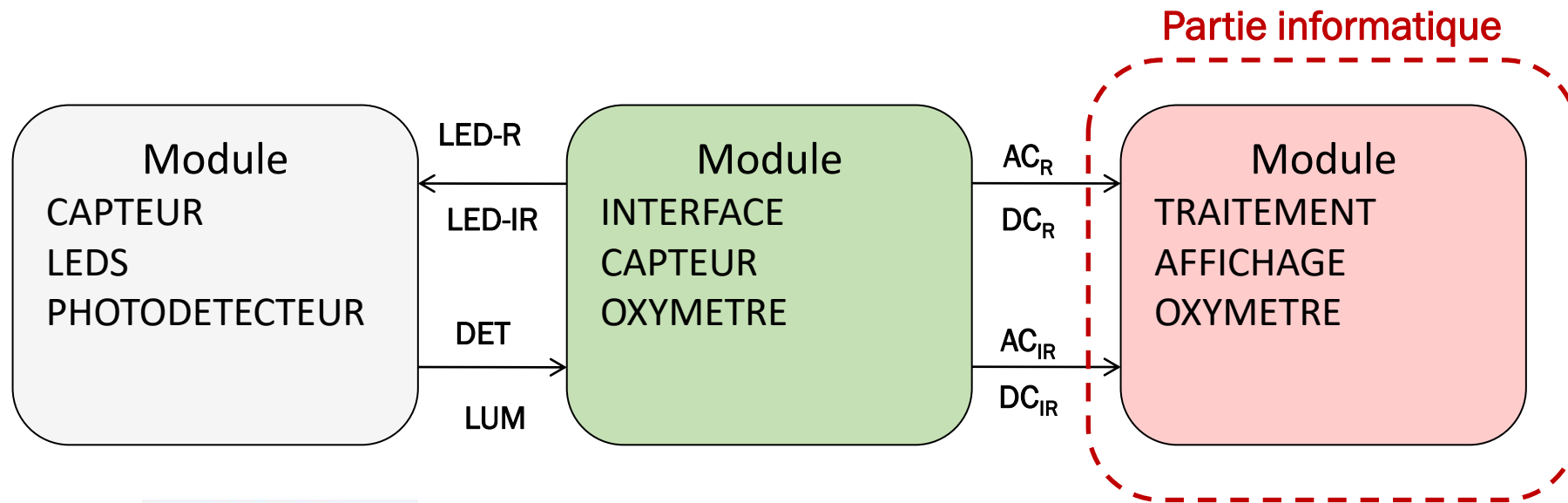
Partie informatique

Pierre-Jean BOUVET



Présentation

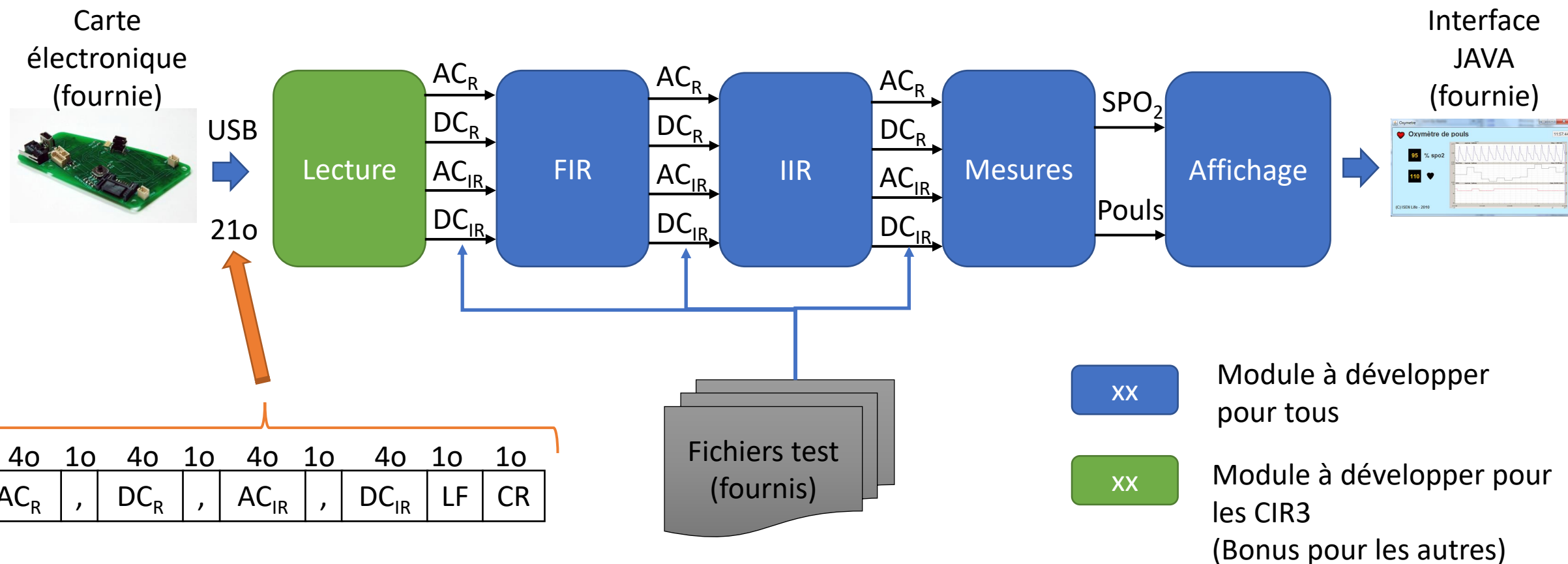
Description générale



Description de la partie informatique

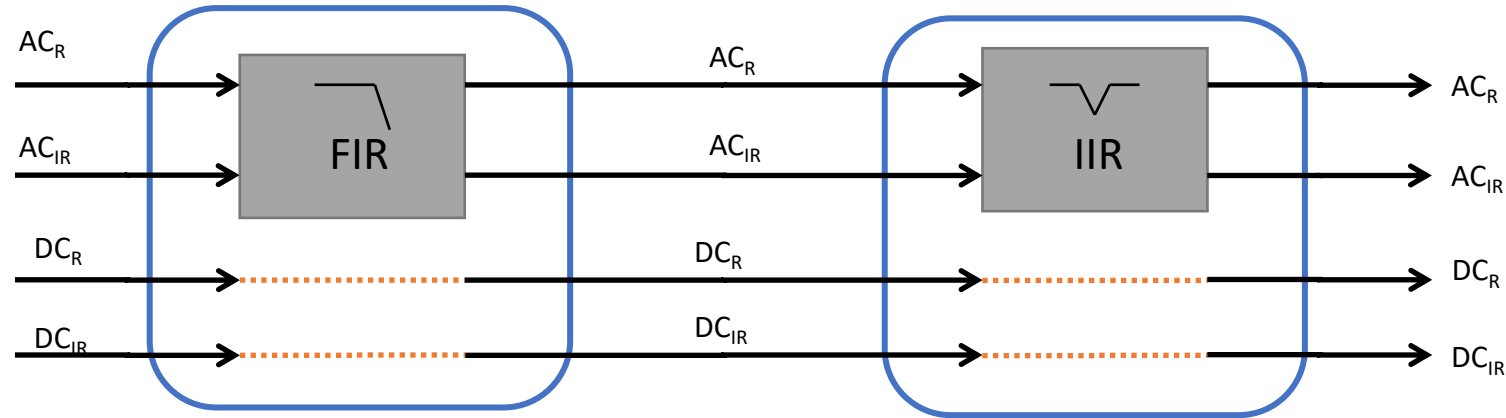
- Récupération des valeurs mesurées de la lumière rouge (ACR et DCR) et de l'infra-rouge (ACIR et DCIR) via liaison USB
- Traitement des informations reçues afin d'éliminer les signaux parasites
- Calcul de l'oxymétrie et de fréquence cardiaque
- Visualisation des résultats via une interface utilisateur

Synoptique de la partie informatique



Les différents blocs

Filtrage (deux blocs : FIR et IIR)



■ Points essentiels

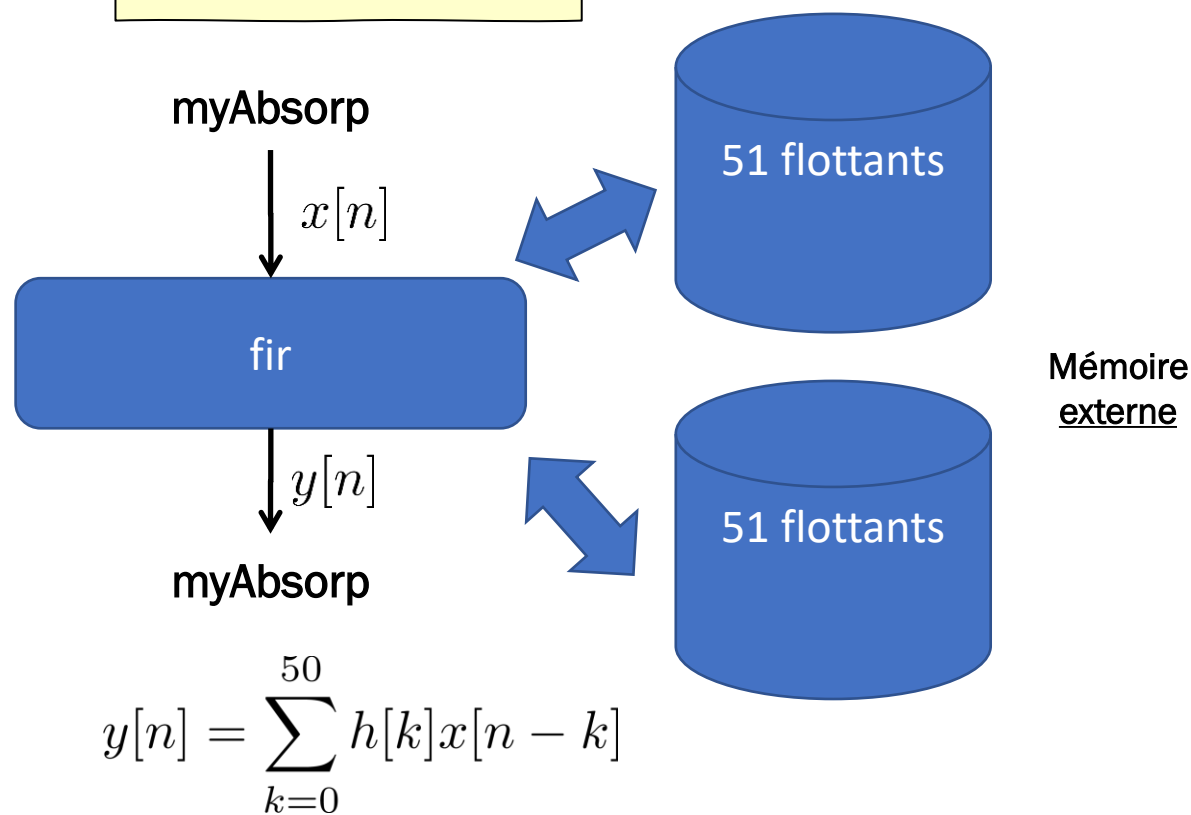
- Filtrage passe-bas (FIR) et passe-haut (IIR) des composantes AC_R et AC_{IR}

■ Points critiques

- Implémentation des filtres

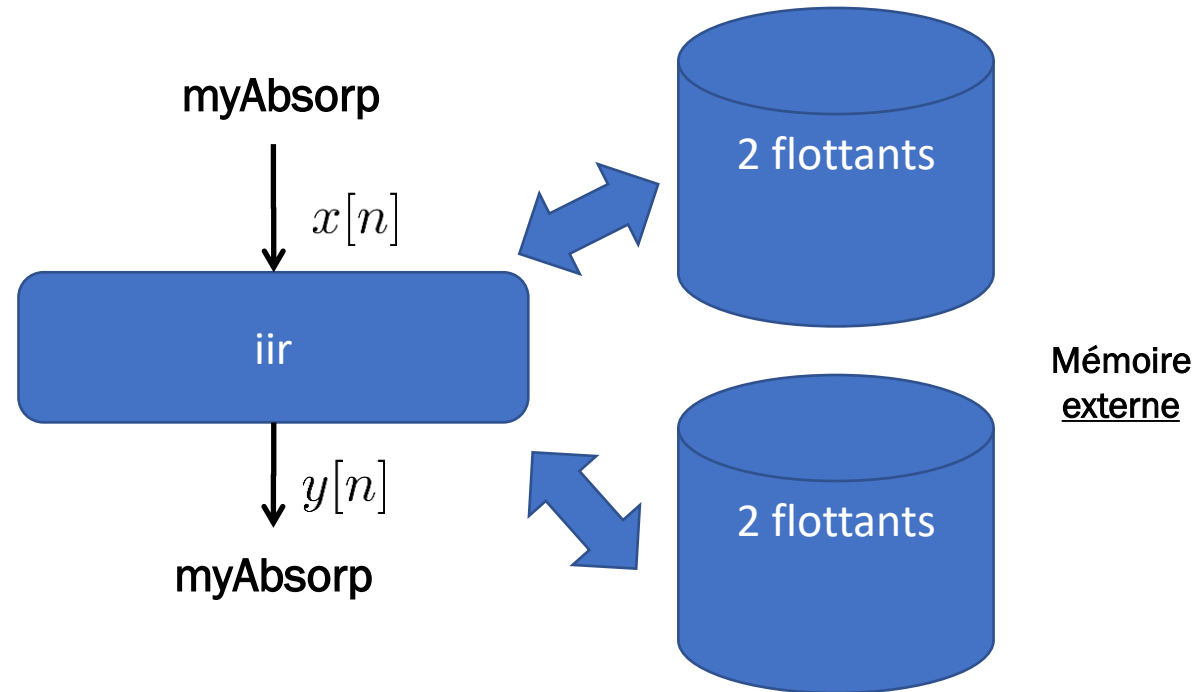
Filtrage FIR

```
typedef struct{
    float acr;
    float dcr;
    float acir;
    float dcir;
} absorp;
```



k	$h[k]$	k	$h[k]$	k	$h[k]$
0	1.4774946e-019	17	3.1294938e-002	34	2.7892178e-002
1	1.6465231e-004	18	3.4578348e-002	35	2.4459630e-002
2	3.8503956e-004	19	3.7651889e-002	36	2.1082435e-002
3	7.0848037e-004	20	4.0427695e-002	37	1.7838135e-002
4	1.1840522e-003	21	4.2824111e-002	38	1.4793934e-002
5	1.8598621e-003	22	4.4769071e-002	39	1.2004510e-002
6	2.7802151e-003	23	4.6203098e-002	40	9.5104679e-003
7	3.9828263e-003	24	4.7081811e-002	41	7.3374938e-003
8	5.4962252e-003	25	4.7377805e-002	42	5.4962252e-003
9	7.3374938e-003	26	4.7081811e-002	43	3.9828263e-003
10	9.5104679e-003	27	4.6203098e-002	44	2.7802151e-003
11	1.2004510e-002	28	4.4769071e-002	45	1.8598621e-003
12	1.4793934e-002	29	4.2824111e-002	46	1.1840522e-003
13	1.7838135e-002	30	4.0427695e-002	47	7.0848037e-004
14	2.1082435e-002	31	3.7651889e-002	48	3.8503956e-004
15	2.4459630e-002	32	3.4578348e-002	49	1.6465231e-004
16	2.7892178e-002	33	3.1294938e-002	50	1.4774946e-019

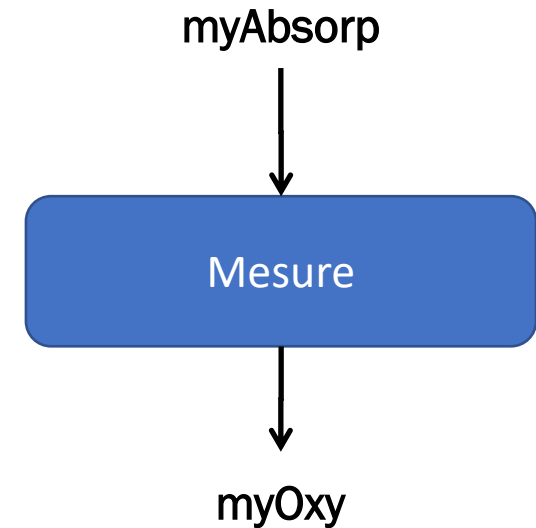
Filtrage IIR



$$y[n] = x[n] - x[n - 1] + \alpha y[n - 1] \quad \text{avec } \alpha = 0.992$$

Mesure

- Prend en entrée les valeurs mesurées de la lumière rouge (AC_R et DC_R) et de l'infrarouge (AC_{IR} et DC_{IR})
- Extraction à partir des différentes composantes des quantités suivantes :
 - Taux de saturation d'oxygène dans le sang (SpO_2)
 - Fréquence cardiaque (pouls)



```
typedef struct{
    int spo2; /*!< SPO2 */
    int pouls; /*!< Pouls */
} oxy;
```

Mesure SpO₂

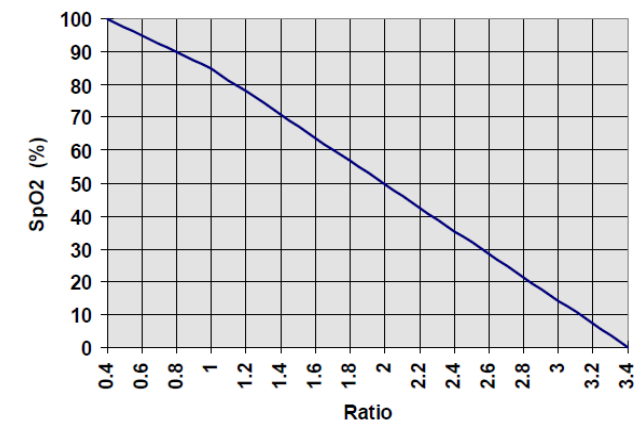
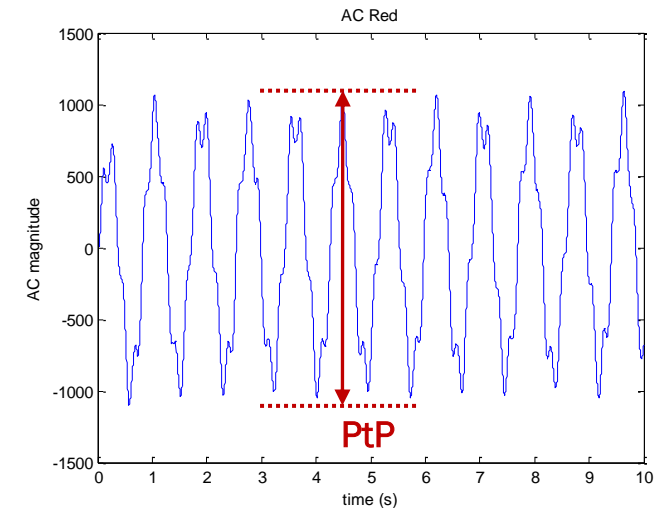
■ Points essentiels

- Calcul de la valeur crête de chaque composante AC
- Calcul de RsIR puis SpO₂ via une table de correspondance

$$R_{sIR} = \frac{\frac{PtP(AC_R)}{DC_R}}{\frac{PtP(AC_{IR})}{DC_{IR}}}$$

■ Points critiques

- Calcul de SpO₂ au "fil de l'eau"



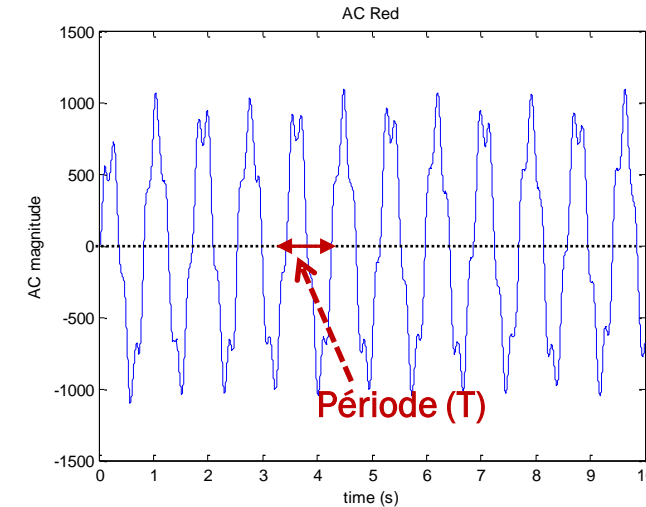
Mesure de la fréquence cardiaque

■ Points essentiels

- Calcul de la fréquence de l'onde de pouls (sur AC_R et AC_{IR})

■ Points critiques

- Calcul de la fréquence cardiaque au "fil de l'eau"



$$F = \frac{1}{T}$$

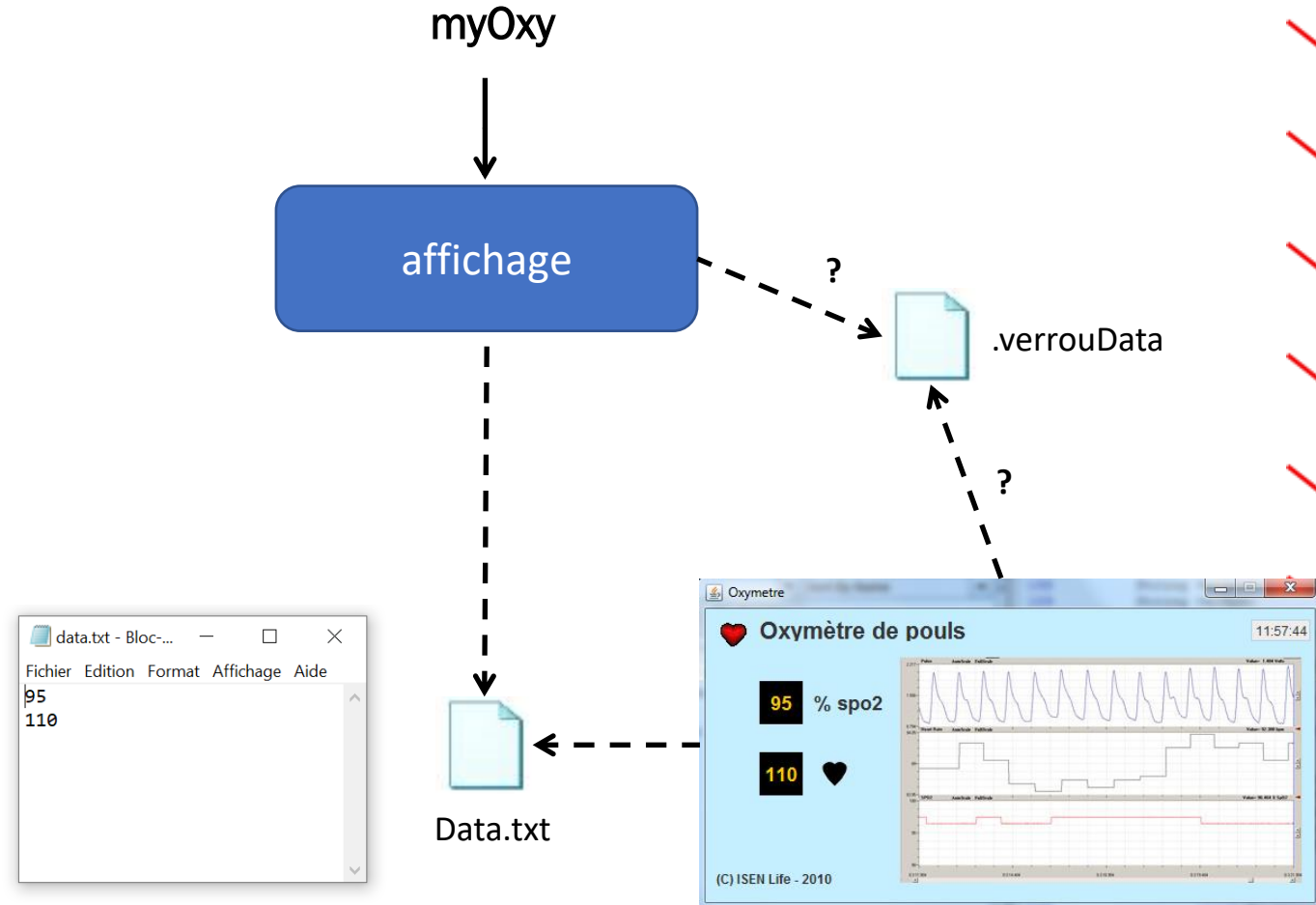
Affichage

■ Points essentiels

- Ecriture des informations dans le fichier Data.txt

■ Points critiques

- Exclusion mutuelle à l'aide d'un verrou



Affichage

■ Test existence d'un fichier

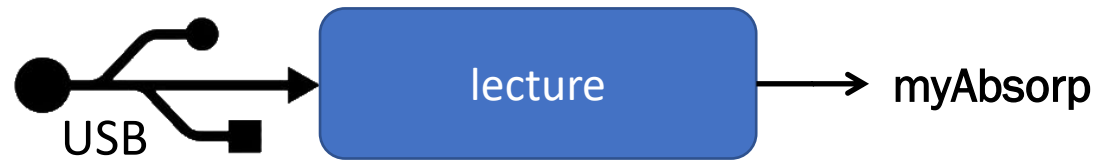
```
#include <unistd.h>

if( access( ".verrouData", F_OK ) != -1 )
{
    // Fichier existe
}else{
    // Fichier n'existe pas
}
```

■ Effacer un fichier

```
remove( ".verrouData" )
```

Lecture



■ Points essentiels

- Lire chacun des champs
- Recentrer les composantes AC autour de 0

■ Points critiques

- Etablir la synchronisation afin de récupérer les valeurs
- Driver FTDI

Du fait du confinement due à l'épidémie COVID-19, l'accès à la carte électronique est potentiellement rendu impossible. Dans ce cas, la lecture sur le port USB sera simulée. A cet effet nous avons créé un fichier nommé `record1_bin.dat` contenant les données à lire sous format binaire.

Lecture

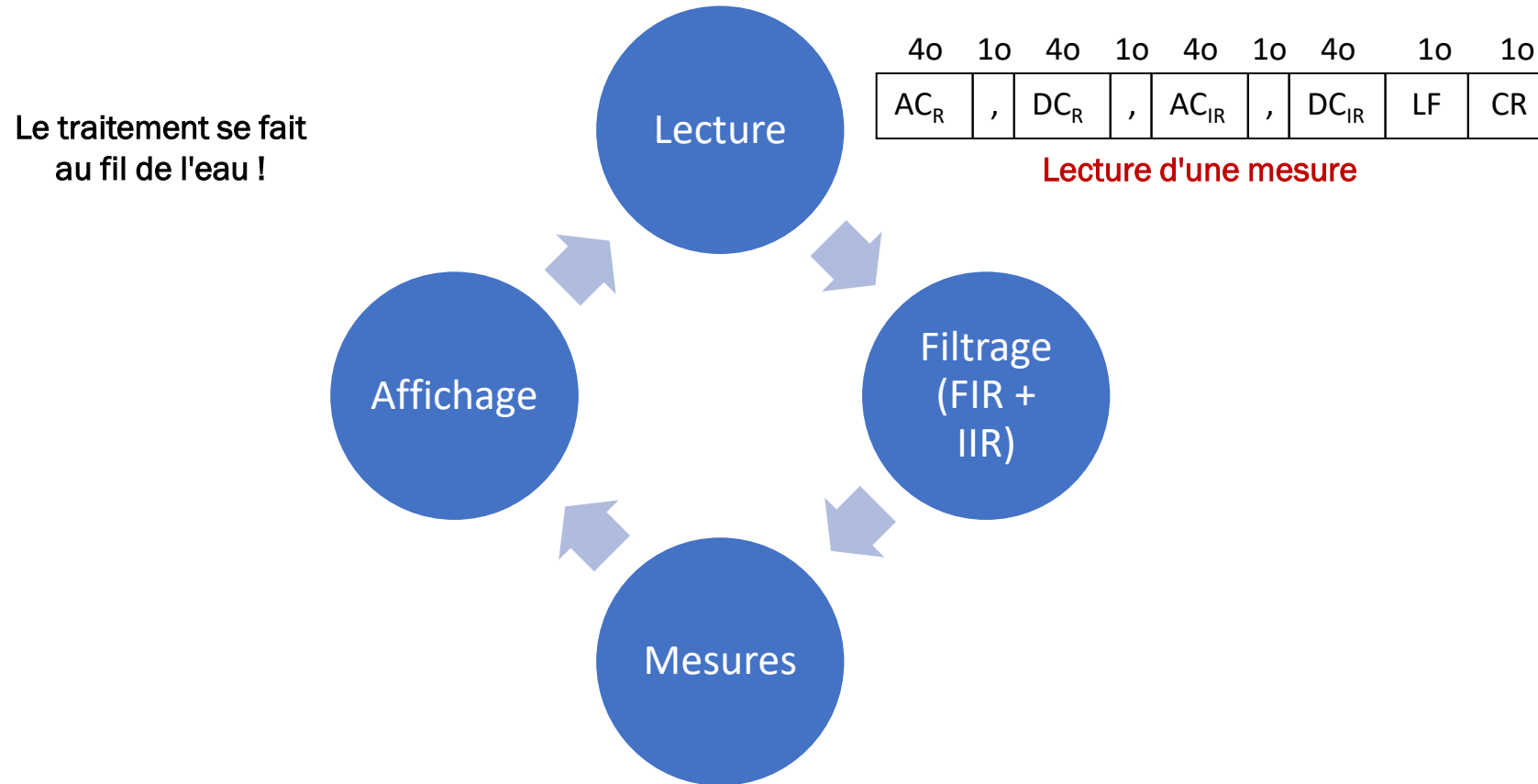
■ Détail de la trame

21 octets (rafraichissement toutes 2 ms)

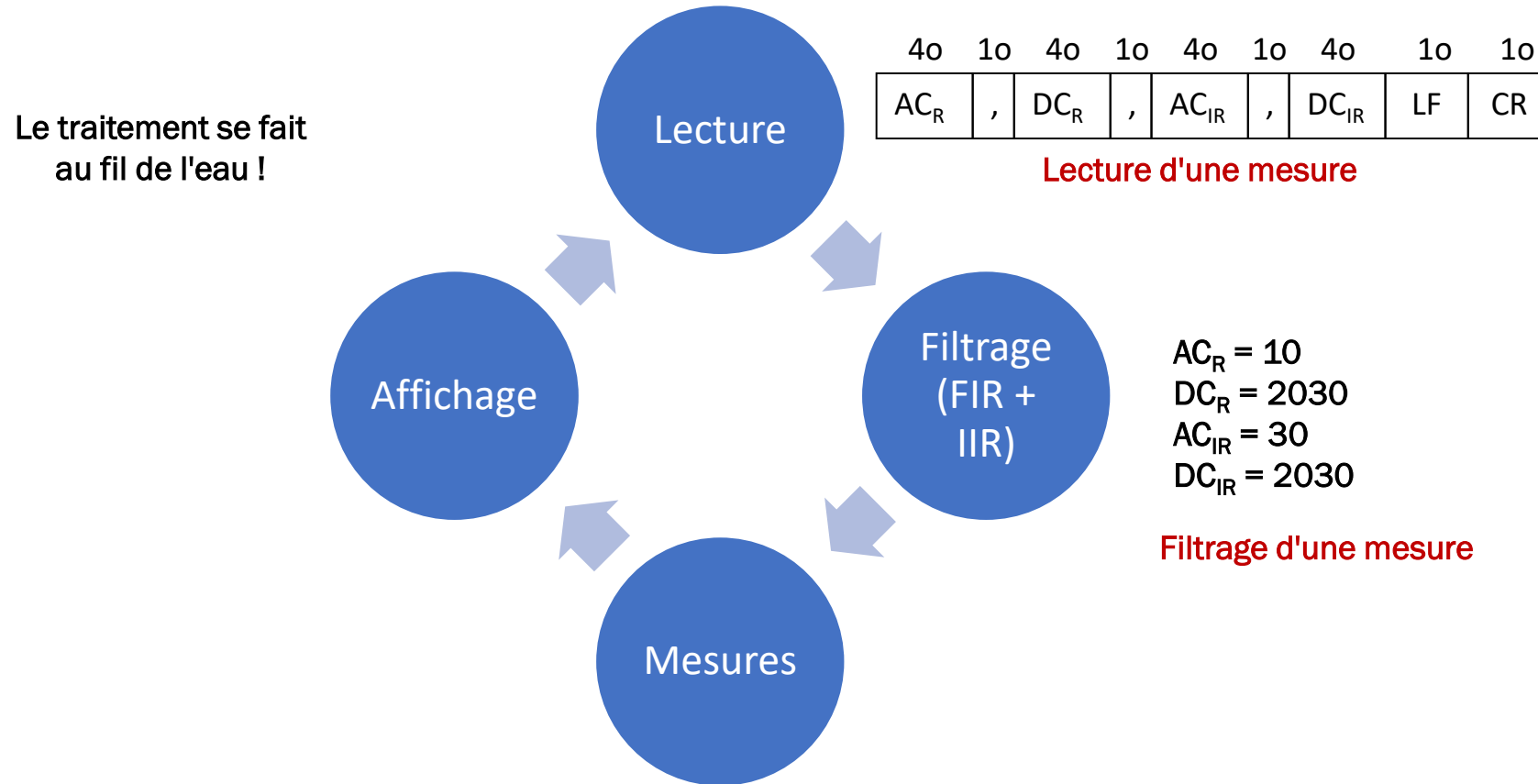
Taille (octets)	4	1	4	1	4	1	4	1	1
Champ	AC _R	,	DC _R	,	AC _{IR}	,	DC _{IR}	LF	CR

Architecture

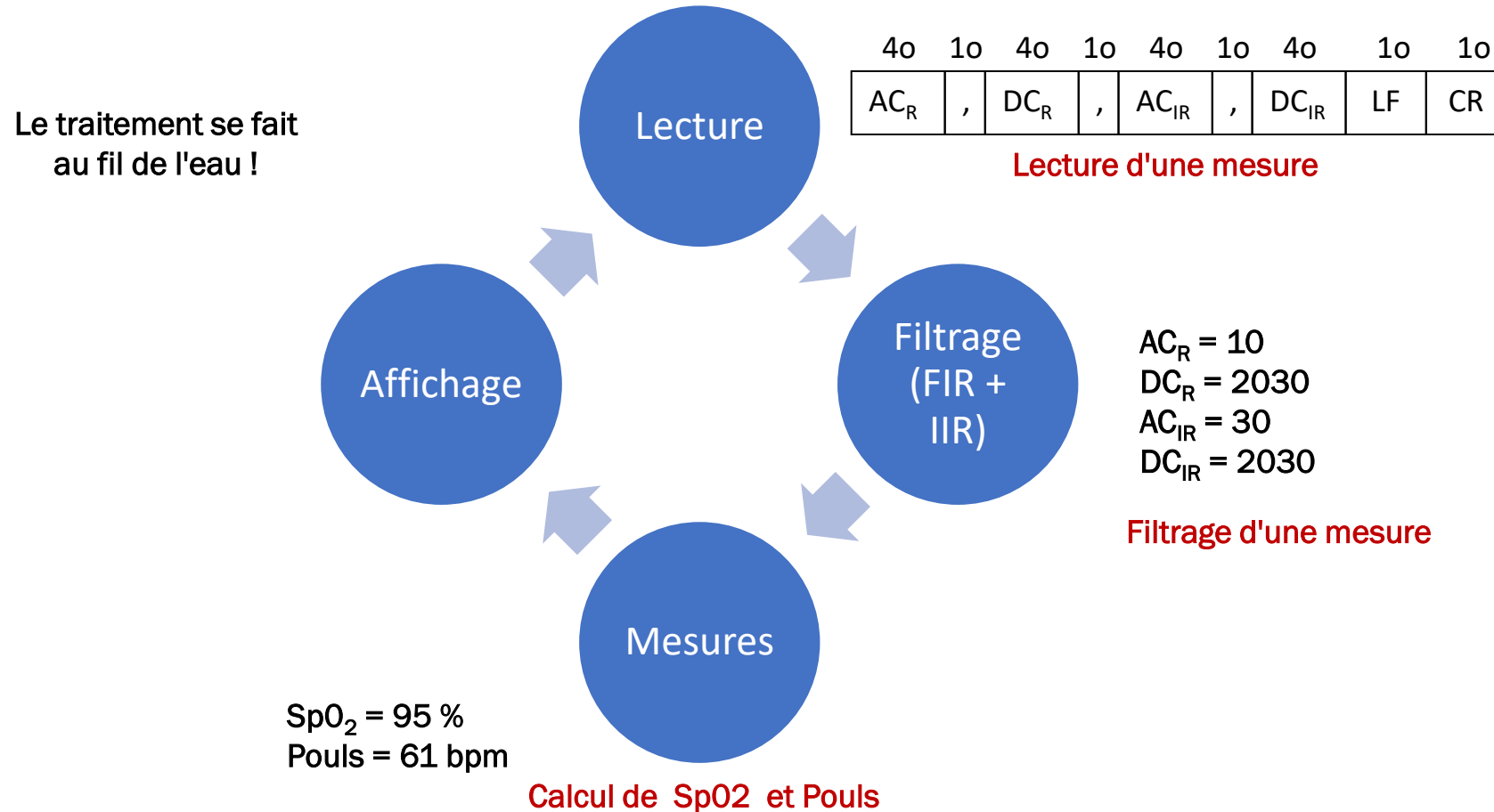
Organisation du programme



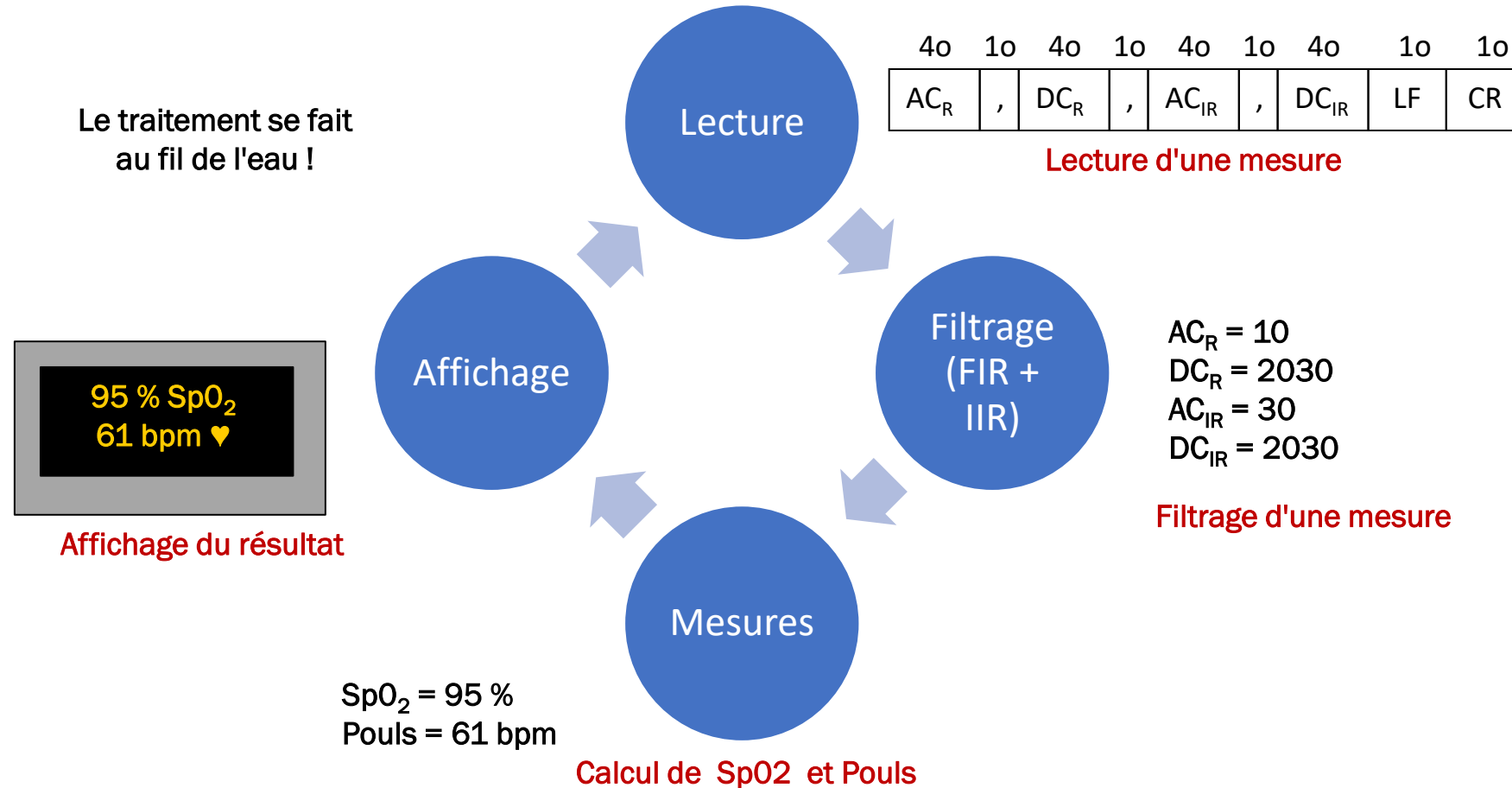
Organisation du programme



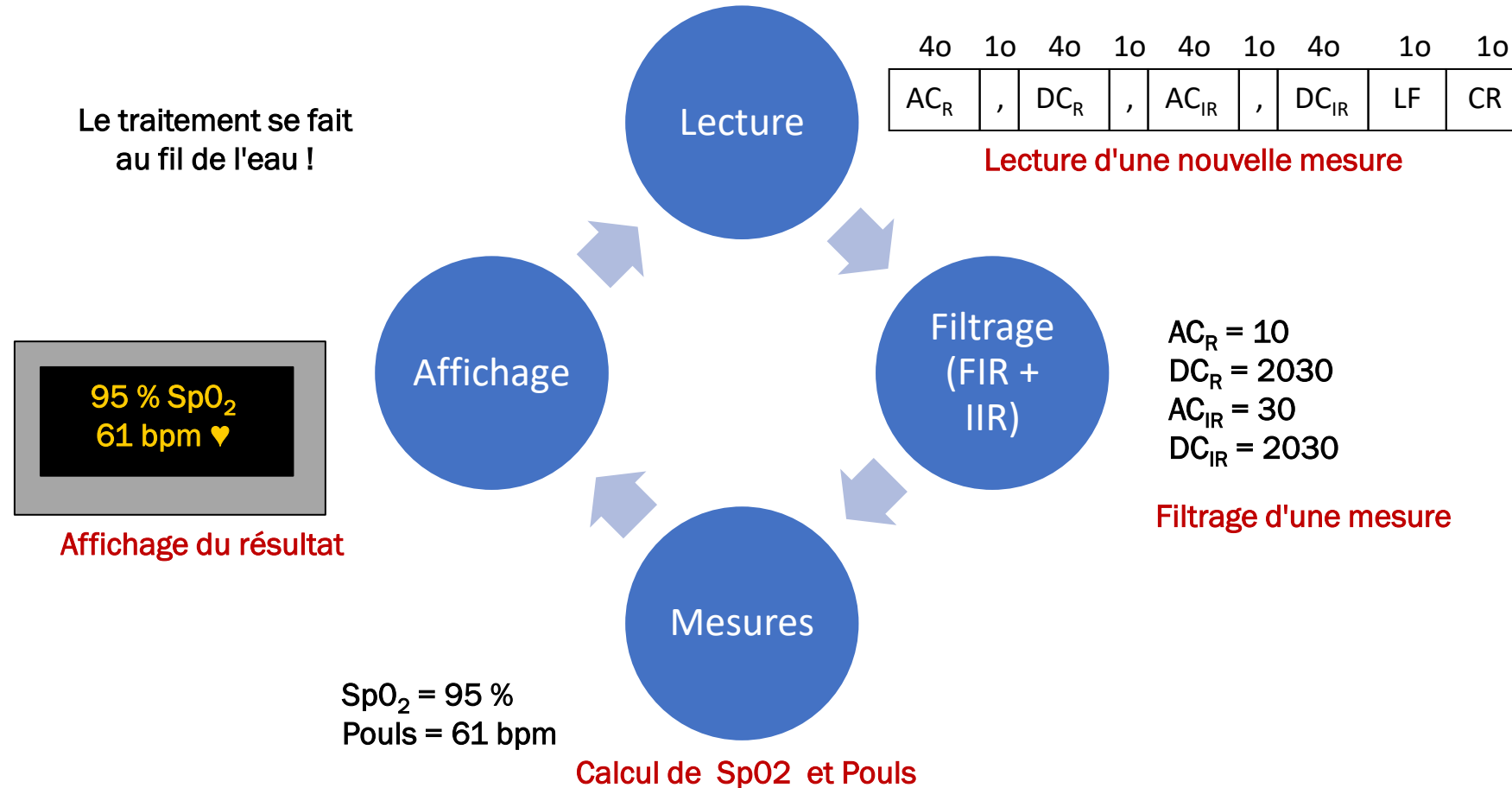
Organisation du programme



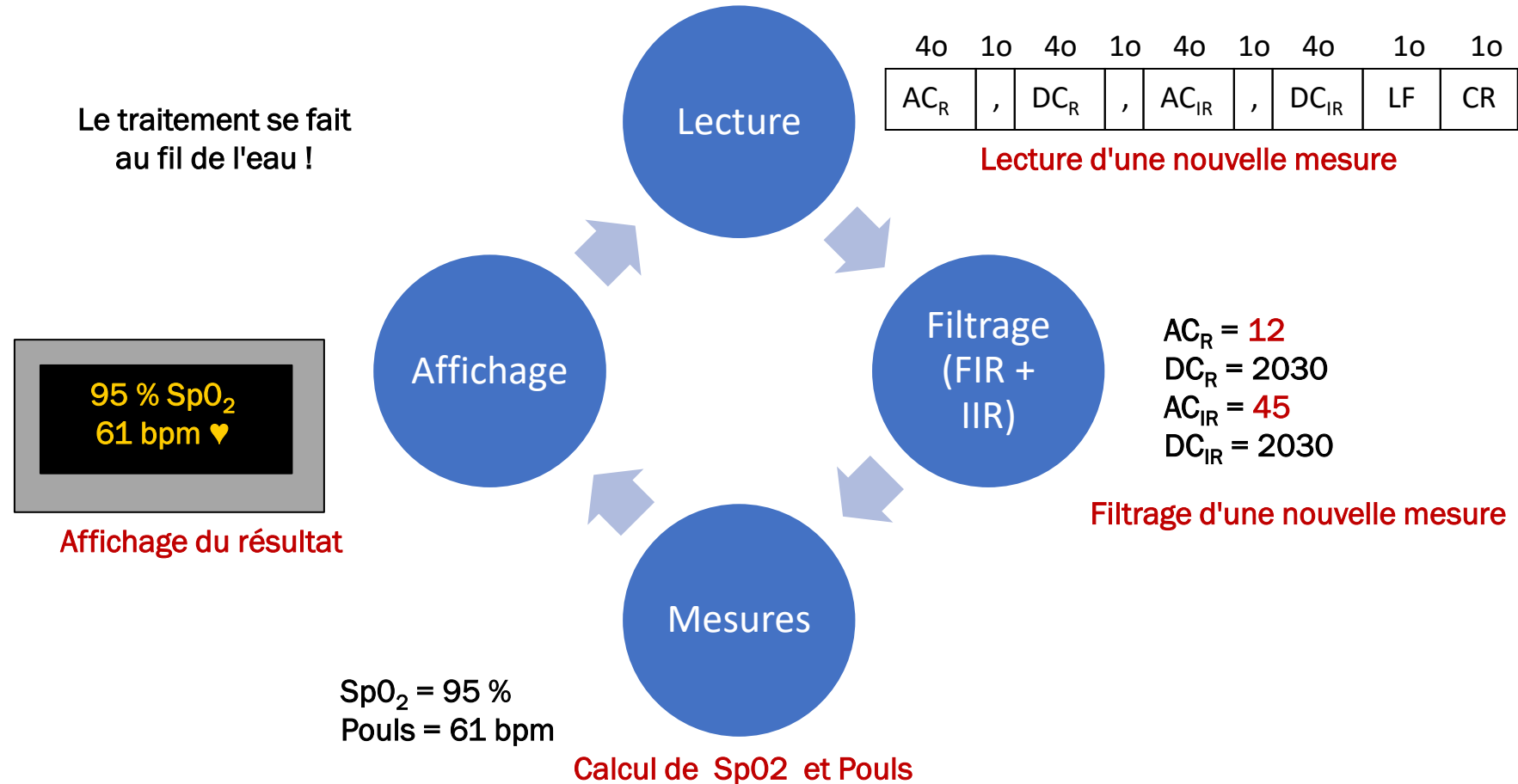
Organisation du programme



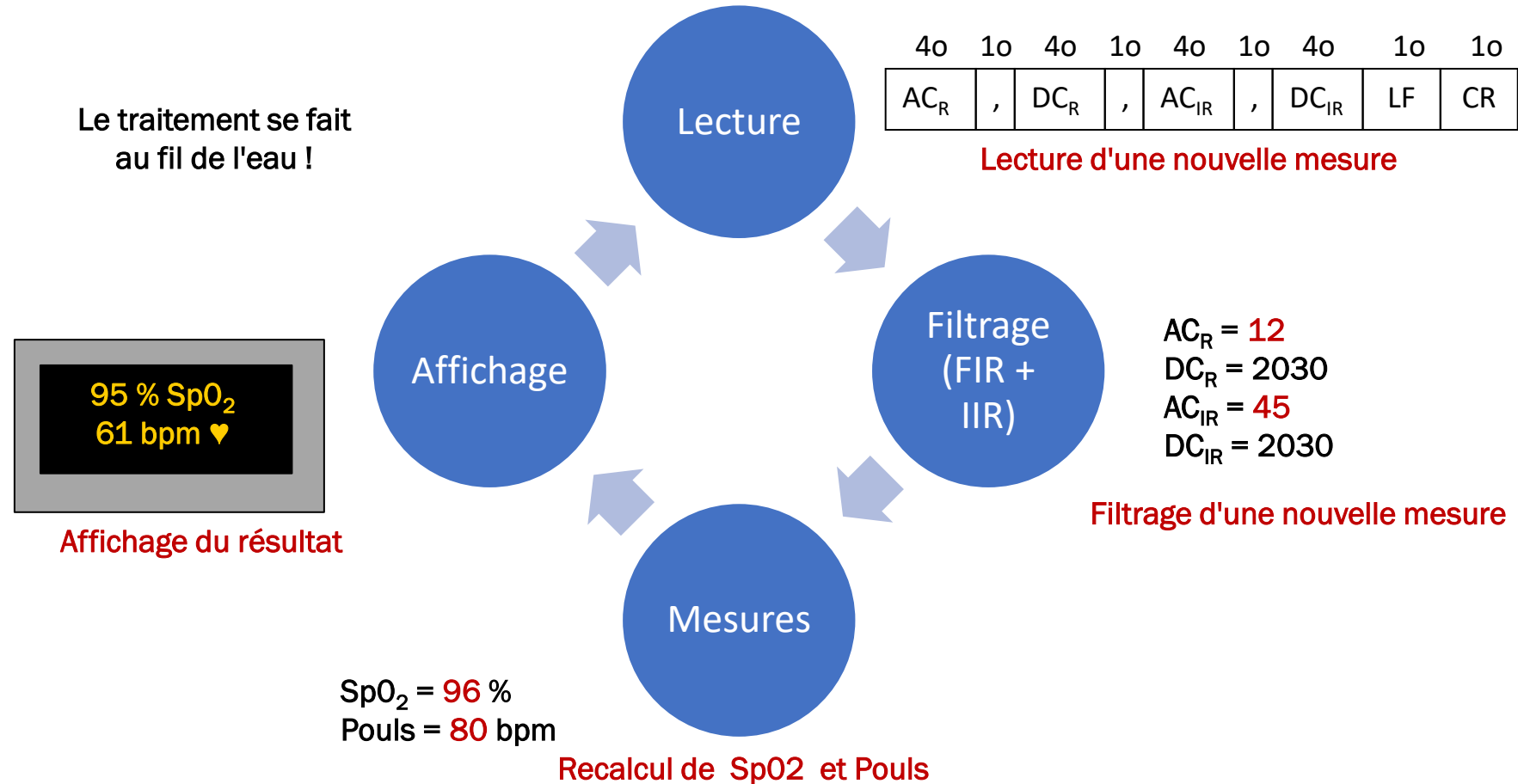
Organisation du programme



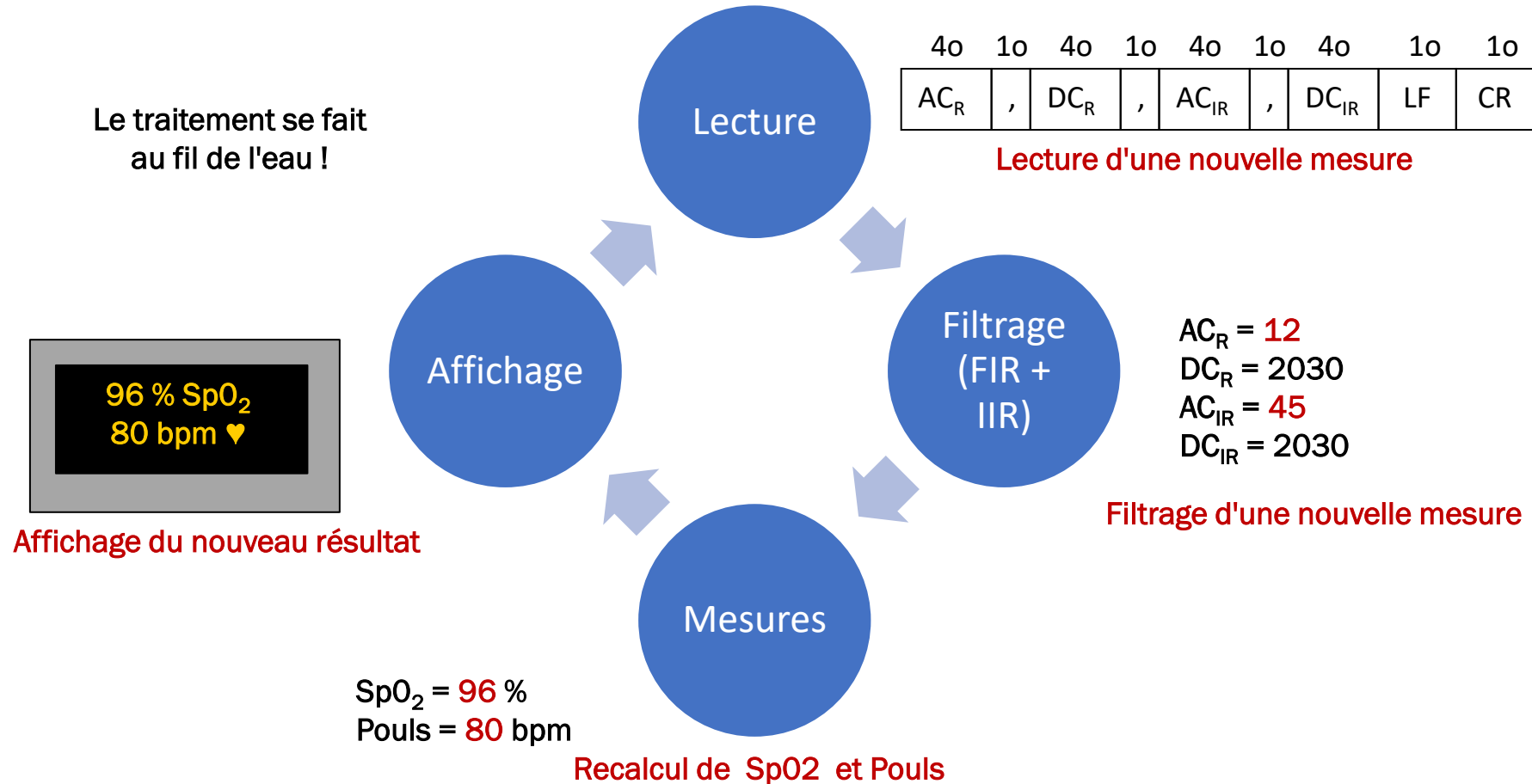
Organisation du programme



Organisation du programme



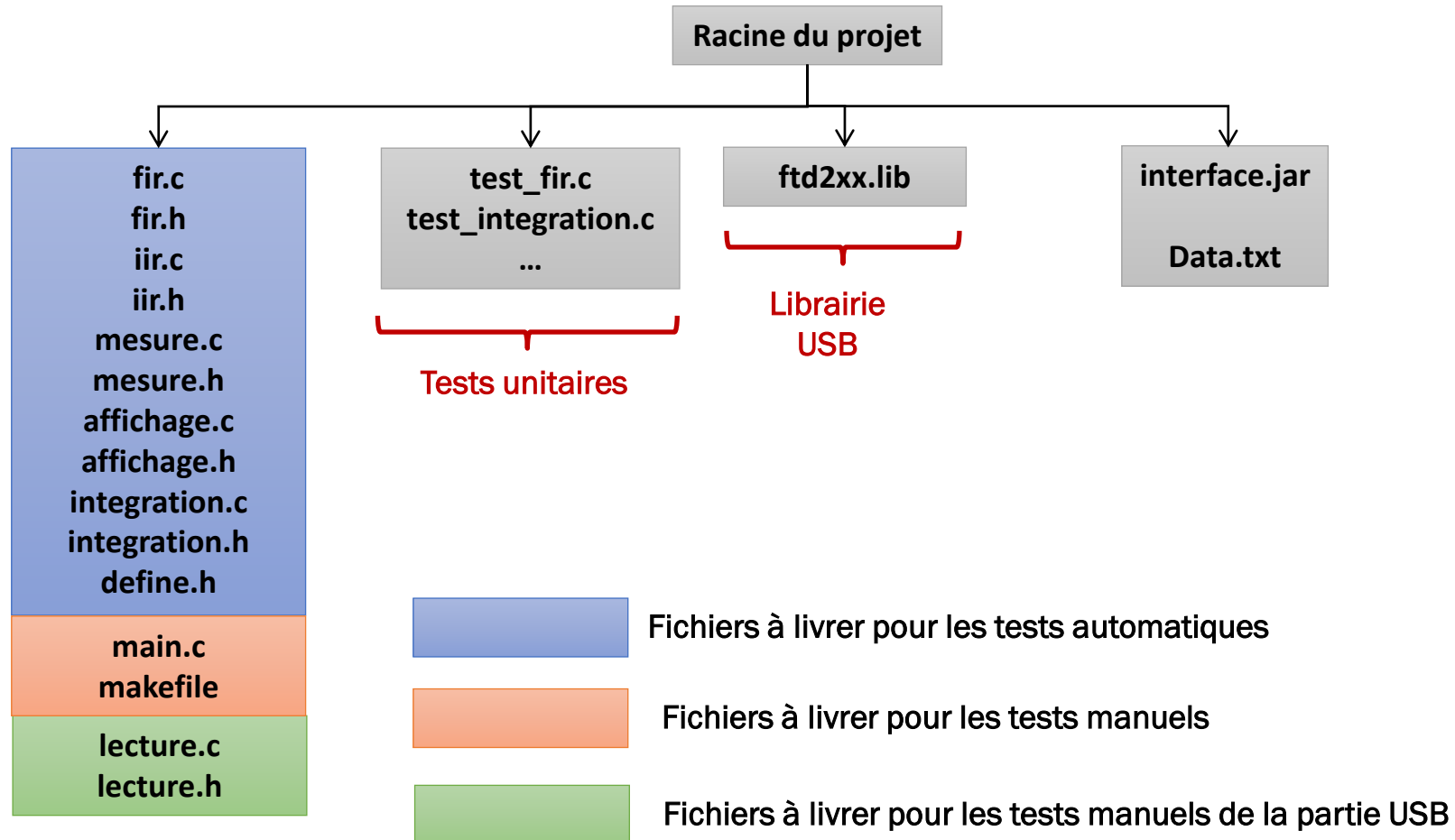
Organisation du programme



Convention de nommage

Bloc	Nom fonction	Nom fichier source	Nom fichier header
Filtrage FIR	<code>fir</code>	<code>fir.c</code>	<code>fir.h</code>
Filtrage IIR	<code>iir</code>	<code>iir.c</code>	<code>iir.h</code>
Mesure	<code>mesure</code>	<code>mesure.c</code>	<code>mesure.h</code>
Affichage	<code>affichage</code>	<code>affichage.c</code>	<code>affichage.h</code>
Intégration	<code>integration</code>	<code>integration.c</code>	<code>integration.h</code>
Lecture	<code>lecture</code>	<code>lecture.c</code>	<code>lecture.h</code>
Définition globales			<code>define.h</code>

Arborescence



Exemple de programme

main.c

```
int main() {
    int etat=0;
    absorp myAbsorp;
    oxy myOxy;
    param_fir* myFIR = init_fir(...); // init FIR
    param_iir* myIIR = init_iir(...); // init IIR
    param_mesure* myMes = init_mesure(...) // init mesure
    FILE* myFile = initFichier("record1.dat");
    do{
        myAbsorp = lireFichier(myFile,&etat);
        myAbsorp = fir(myAbsorp,myFIR);
        myAbsorp = iir(myAbsorp,myIIR);
        myOxy = mesure(myAbsorp,myMes);
        affichage(myOxy);
    }while( etat != EOF );
    finFichier(myFile);
    fin_mesure(myMes);
    fin_iir(myIIR);
    fin_fir(myFIR);
    return EXIT_SUCCESS;
}
```

define.h

```
#ifndef DEFINE_H
#define DEFINE_H

typedef struct{
    float acr; /*!< AC R */
    float dcr; /*!< DC R */
    float acir; /*!< AC IR */
    float dcir; /*!< DC IR */
} absorp;
typedef struct{
    int spo2; /*!< SPO2 */
    int pouls; /*!< Pouls */
} oxy;

#endif
```


Déroulé du projet

4 phases



Lundi matin	Lundi après-midi	Mardi matin	Mardi après-midi	Mercredi matin	Mercredi après-midi	Vendredi matin	Vendredi après-midi
Analyse	Réalisation				Intégration		Recette

- ✓ Analyse de chaque bloc
- ✓ Architecture du programme
- ✓ Définition des Types de données

- ✓ Code source de chaque bloc
- ✓ Tests unitaires
- ✓ Makefile

- ✓ Code source programme principal
- ✓ Test intégration avec le simulateur ou avec la carte
- ✓ Validation avec l'interface java

- ✓ **Code source à rendre avant 15:00**
- ✓ QCM de 15:00 à 15:30
- ✓ Validation automatique et manuelle par l'enseignant sur PC enseignant à partir de 15:30
- ✓ Qualimétrie

Au jour le jour

■ Travail en binôme

- Les binômes sont constitués le premier jour
 - Les binômes à constituer au sein de chaque sous-groupe
 - Les redoublants doivent se mettre ensemble ou en monôme le cas échéant
- Chaque étudiant connaît l'ensemble du projet
- Attention à bien se répartir le travail

■ Ressources externes

- Tous les documents sont autorisés...
 - ... mais les informations nécessaires au projet sont fournies !!!
- Attention à utiliser avec une grande précaution tous documents venant du web (site de vulgarisation, forum, code d'autrui,...)

■ Livraison de code ou de document

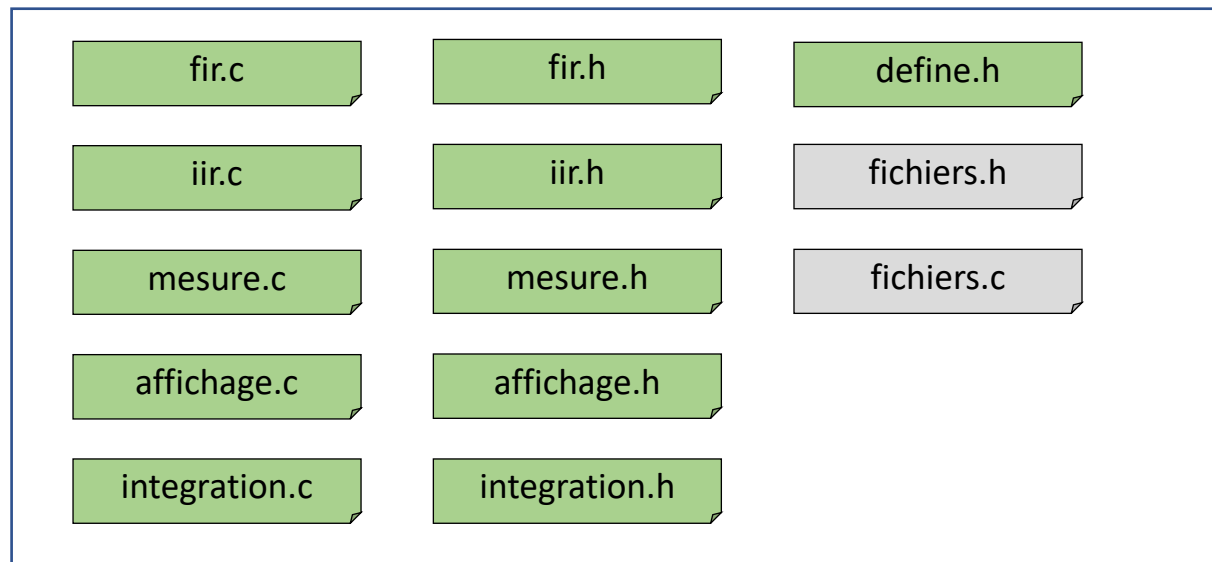
- Ne pas attendre la dernière minute pour poster le livrable
 - Préparer des livrables intermédiaires
 - Sauvegarder régulièrement vos données

Recette



Ce qu'il faut rendre (CGSI3/CENT3/CBIO3/CEST3)

■ Sources pour tests automatiques



tests_auto_xx_yy.zip

Pour vous aider, une coquille vide comprenant l'architecture imposée est disponible sur l'ENT



Fichiers à modifier



Fichiers à ne pas modifier

■ Sources pour tests manuels

- Fichiers précédents
- `main.c`
- Makefile
- Si USB: `lecture.c` et `lecture.h`



tests_manuels_xx_yy.zip

Ce qu'il faut rendre (CIR3)

■ Sources pour tests automatiques

<code>fir.c</code>	<code>fir.h</code>	<code>define.h</code>
<code>iir.c</code>	<code>iir.h</code>	<code>fichiers.h</code>
<code>mesure.c</code>	<code>mesure.h</code>	<code>fichiers.c</code>
<code>affichage.c</code>	<code>affichage.h</code>	<code>lecture.c</code>
<code>integration.c</code>	<code>integration.h</code>	<code>lecture.h</code>



tests_auto_xx_yy.zip

Pour vous aider, une coquille vide comprenant l'architecture imposée est disponible sur l'ENT



Fichiers à modifier



Fichiers à ne pas modifier

■ Sources pour tests manuels

- Fichiers précédents
- main.c
- Makefile



tests_manuels_xx_yy.zip

Le code source

■ Chaque bloc :

- Un fichier
- Une fonction

Les interfaces de [la fonction principale de chaque bloc](#) sont imposées afin de permettre une validation automatique lors de la recette. Les prototypes vous seront fournies en début de réalisation !



■ Programme principal

- Permet de tester chaque bloc de façon indépendante (ne doit pas remis)
- Permet de tester le programme complet (obligatoire pour tests manuels)
 - Avec le simulateur
 - Avec la carte électronique

■ Makefile (obligatoire seulement pour tests manuels)

- Permet de compiler automatiquement votre code source

■ Revue de code

- Lors de la recette, il vous sera demandé d'expliquer une ou plusieurs portions de votre code
- **Attention pénalité si les explications ne sont pas convaincantes**
- Pas de points données lors de la revue de code (seulement pénalités le cas échéant)

Barème final (indicatif)

	Coefficient
Recette programme	6
Qualimétrie code	2
QCM	2
Total	10

Barème recette (indicatif)

#	Nom du test	Type	Barème A3 hors CIR3	Barème CIR3
1	FIR	Automatique	3,5	3
2	IIR	Automatique	3	2,5
3	Mesure SPO2	Automatique	2,5	2
4	Mesure Pouls	Automatique	2,5	2
5	Affichage	Automatique	2,5	2,5
6	Intégration	Automatique	3,5	3
7	Programme global simulation	Manuel	2.5	2
8	Lecture / USB	Automatique / manuel	Bonus	3
Total			20	20



- **Aucun point donné au regard du code source (l'évaluateur ne rentre pas dans le code)**
- **Seulement sont testées les fonctionnalités d'exécution de votre code**

Tests automatiques

- **Projet à sélectionner :**
 - Oxymétrie-A3 pour les A3 hors CIR3
 - Oxymétrie-CIR3 pour les CIR3 / ou non CIR souhaitant tester la fonction lecture

General informations

Project: Oxymetrie
Student: eleven3 EleveN3
Mark: 2

[Download Src](#)

Details

Function	Test	Status	Mark
test_affichage	test_affichage_display	1	0.5
	test_affichage_lock	1	0.5
test_fir	test_fir	1	1
test_iir	test_iir	1	1
test_integration	test_integration1	0	0.5
	test_integration2	0	0.5
test_mesure	test_mesure_pouls	0	0.5
	test_mesure_spo2	0	0.5

Les ressources

Le matériel

■ Votre PC portable

- Analyse
- Réalisation
- Intégration



Casque / écouteur
audio déconseillés (consignes non
appliqués = pénalité) !

■ Carte électronique

- Validation de la partie USB



Boissons et nourriture
interdites !

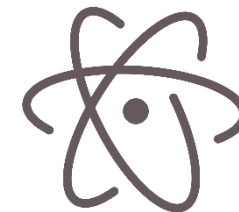
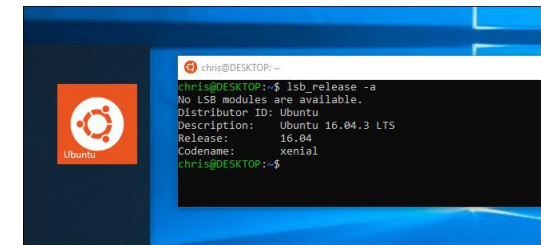
Outils de développement

■ Noyau de compilation

- Bash Windows
- Jupyterhub
- Linux natif

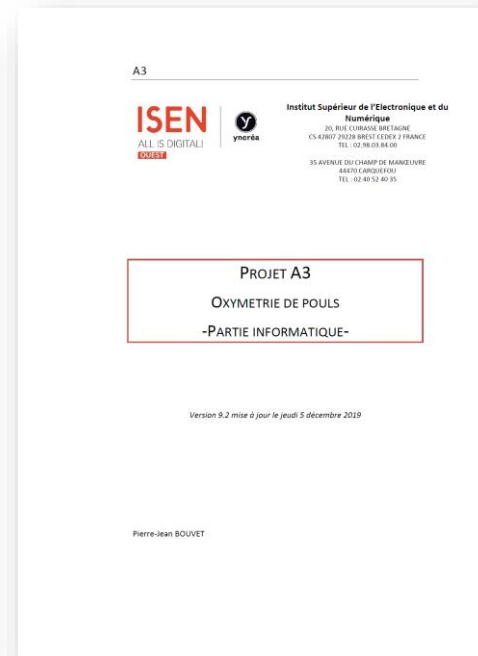
■ Editeur

- Notepad++
- Sublime Text
- Atom
- Visual studio code
- ...



Documentation

- Support de présentation
- Description détaillée
- Guide de programmation FTDI



MERCI
Des questions ?

