

Projet Fondamentaux Scientifiques

Cardiofréquencemètre

Rapport synthétique du projet

17/11/2017

Chef de projet : Joël DIDIER

Membres du groupe : Louis MARJOLET

Philippe BURLET

Vicente VAZ

• Introduction

HeXart Care est une startup très prometteuse spécialisée dans l'électronique et l'informatique.

Son dernier projet innovant est un lecteur portatif grand public de la fréquence cardiaque.

L'entreprise a développé un savoir-faire depuis une dizaine d'années et elle s'impose petit à petit comme un acteur important dans la recherche et l'implémentation de solutions innovantes dans le monde de la santé. . .

Malheureusement cette bonne perspective a été coupée net le 31/10/2017 après avoir été victime d'un sabotage industriel. L'ingénieur principal de l'entreprise a disparu mystérieusement, emportant avec lui tous les prototypes ! Tous les salariés ont découvert leur poste formaté et toutes les sauvegardes ont été détruites. L'entreprise se trouve en grande difficulté car un prototype du projet devait être présenté le 20 novembre auprès des investisseurs.

HeXart Care a fait appel par le passé à des stagiaires Exia en A4 lors de sa mission à l'étranger. Les étudiants ingénieurs de l'EXIA sont très appréciés. Le dirigeant a fait appel à nous pour l'aider. Tous nos étudiants de A4 étant en stage actuellement, nous lui avons proposé de présenter le projet à nos étudiants de première année qui travaillent sur les fondamentaux scientifiques et informatiques en ce début d'année.

En accord avec l'entreprise, les meilleures équipes seront sélectionnées pour la présentation aux investisseurs lors de la soutenance du 20/11. . .

Extrait du sujet de présentation du projet

• Objectifs

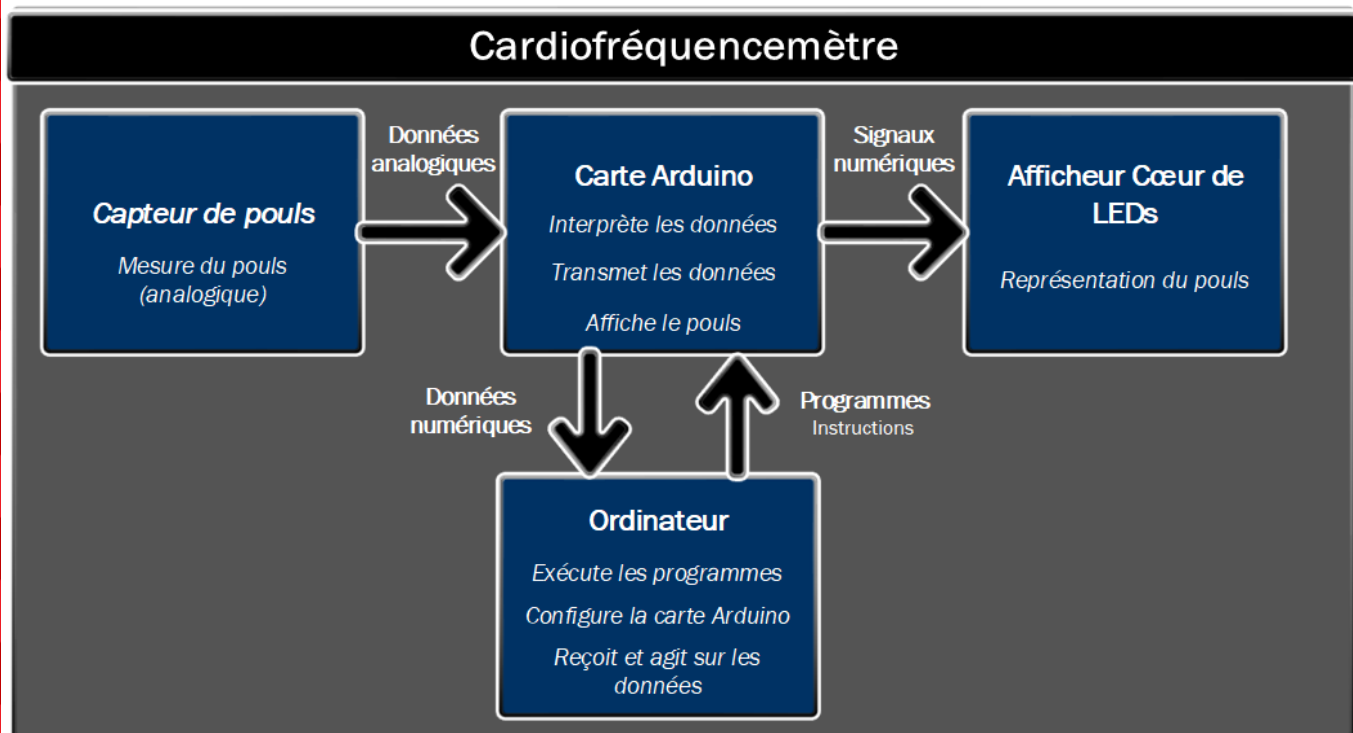
Il s'agit ici de réaliser un capteur cardiaque mesurant le pouls d'une personne insérant son doigt dans une pince créée dans ce but.

Un afficheur composé de LEDs (disposées en forme de cœur) doit clignoter dès lors que le pouls est détecté (un clignotement par battement détecté) et suivant un motif choisi au préalable par l'utilisateur.

Enfin, les données issues du capteur (temps, pouls) doivent être enregistrées et pouvoir être exploitées par un programme exécuté sur un ordinateur.

- Vue d'ensemble

Voici un diagramme permettant d'avoir une vue d'ensemble du dispositif demandé.



• Module 1 - Cardiofréquencemètre

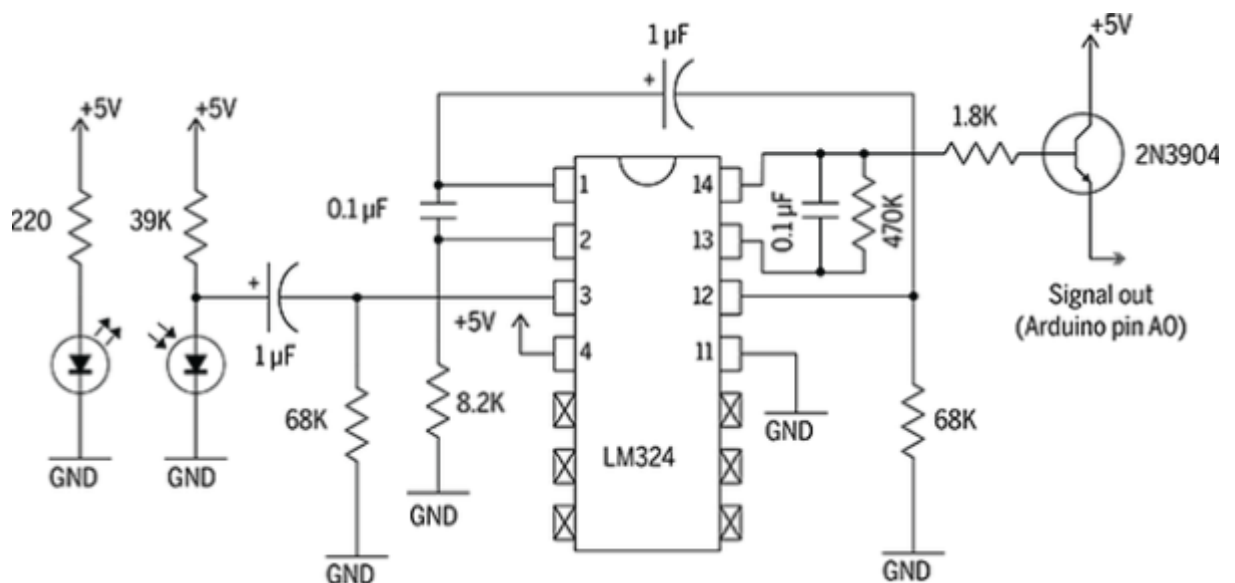
Partie I – Réalisation du montage électronique

Il s'agit ici de réaliser le montage électronique du capteur.

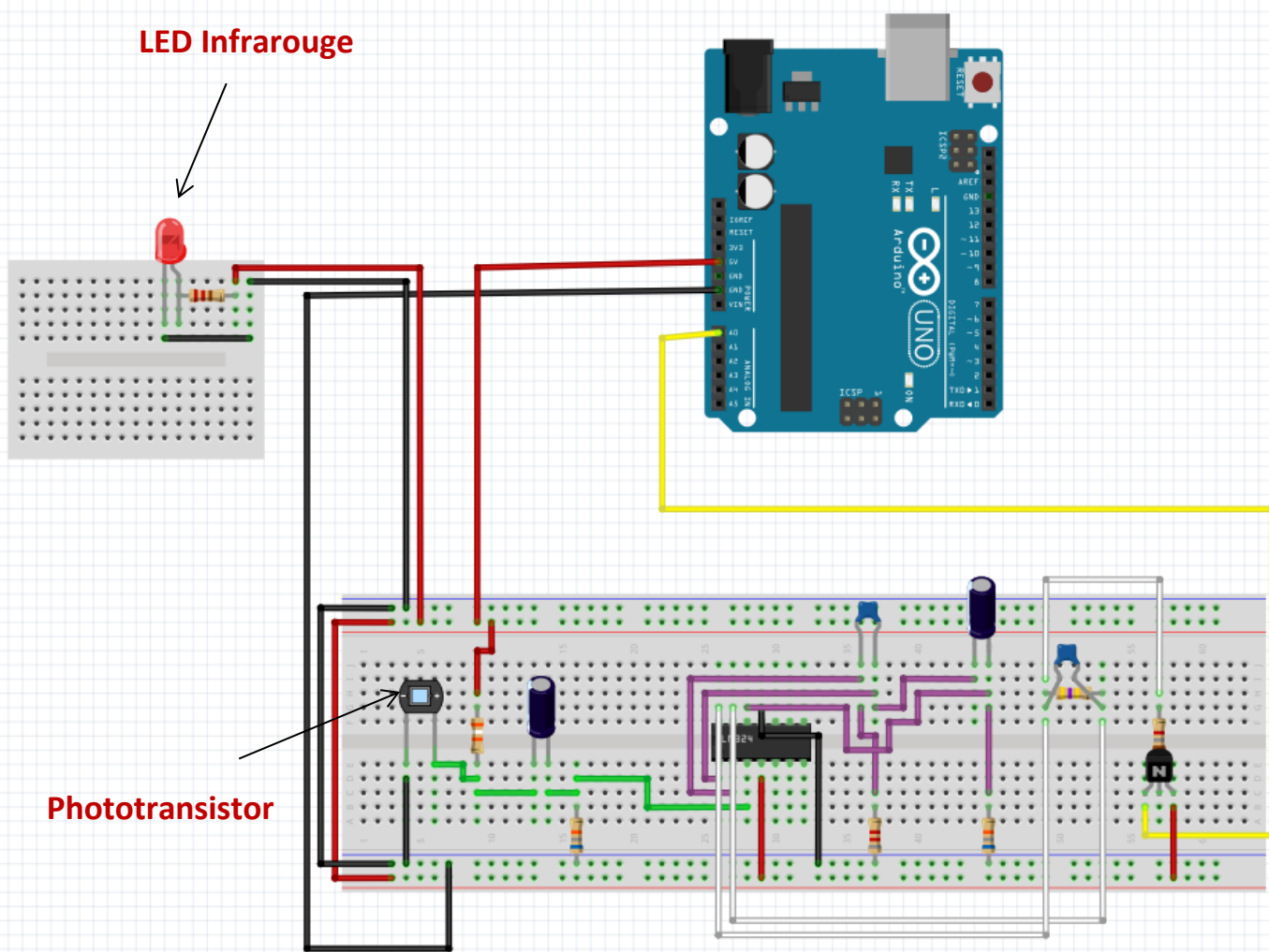
On rappelle la liste des composants fournis :

- LED Infrarouge (TSAL7400) - Tige la plus courte à GND
- Phototransistor sensible à l'infrarouge (TOPS-050) - Tige la plus longue à GND
- Amplificateur opérationnel (LM324)
- Transistor (2N3904)
- Arduino Uno
- Quelques condensateurs
- Jeux de résistances

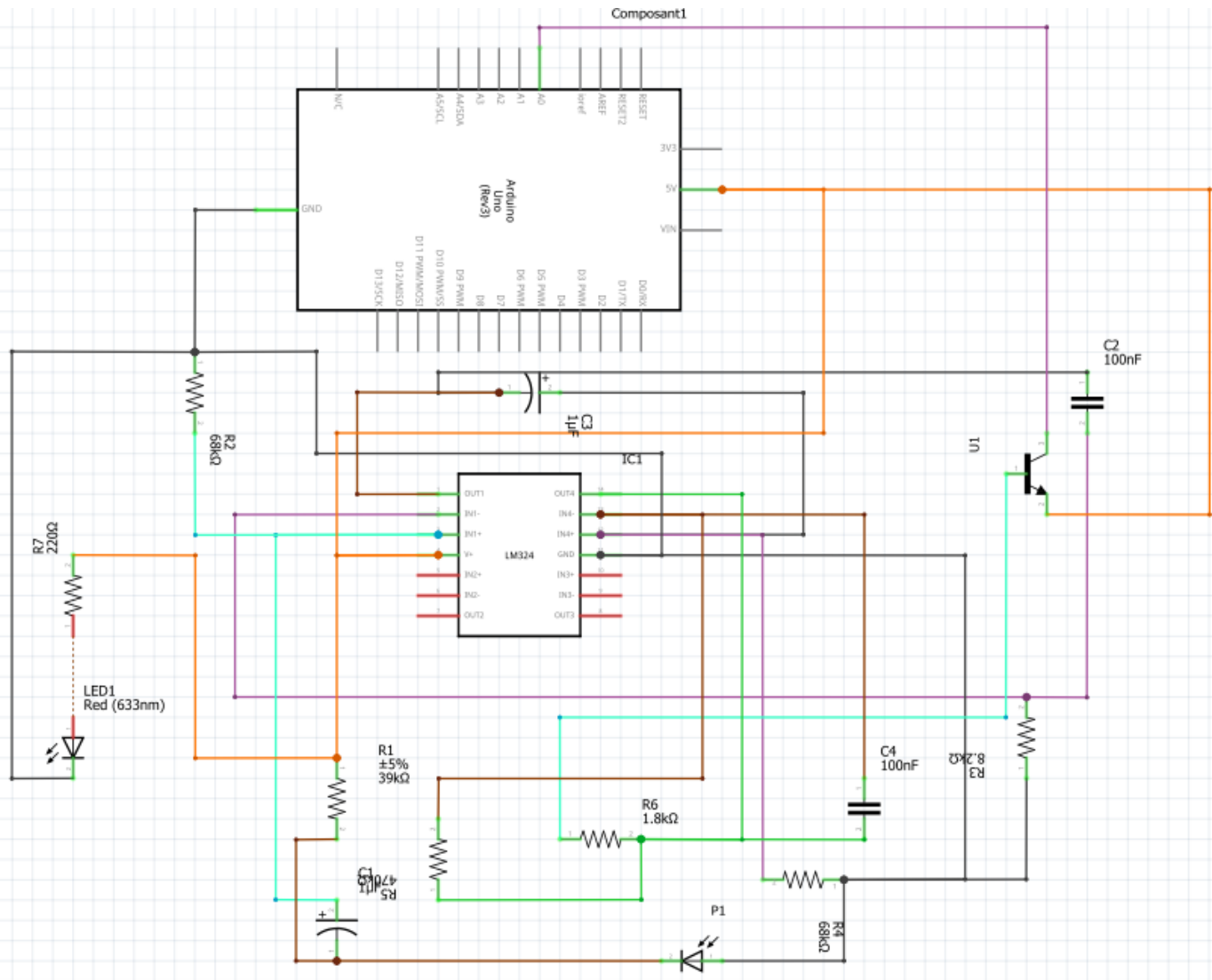
On rappelle le schéma fourni qui servira de modèle :



Vue Platine :



Vue schématique :

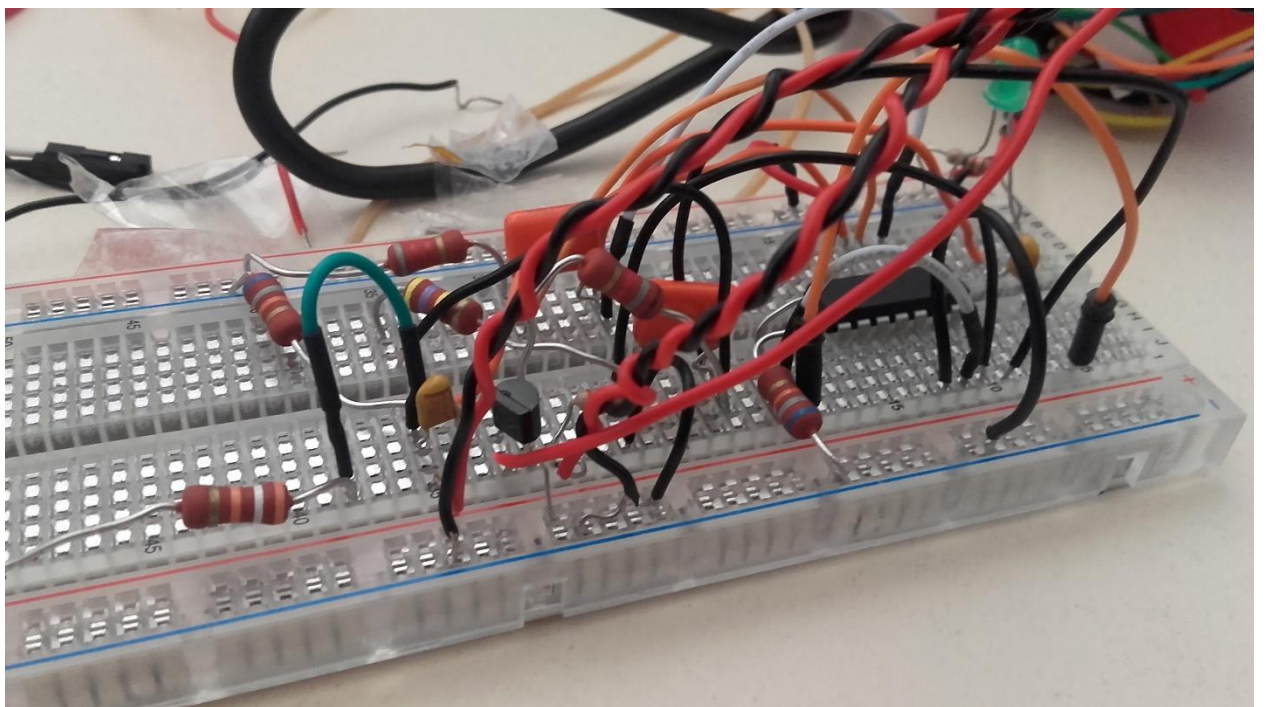
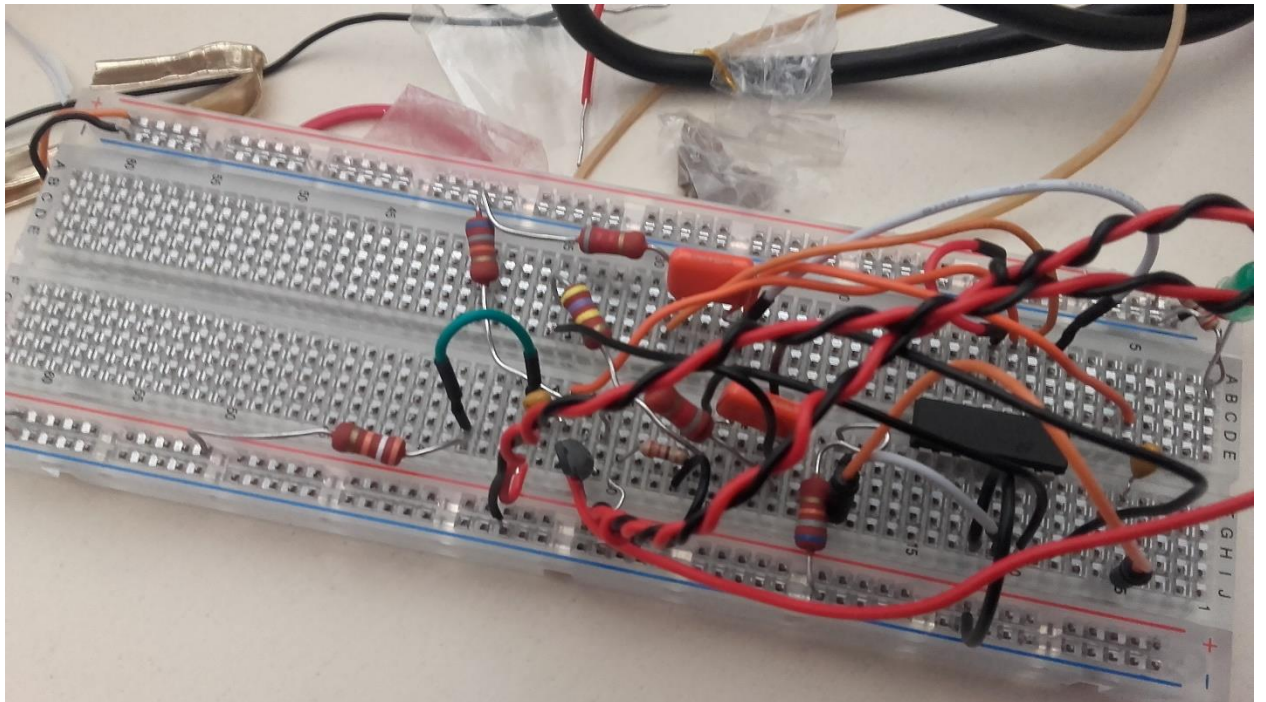


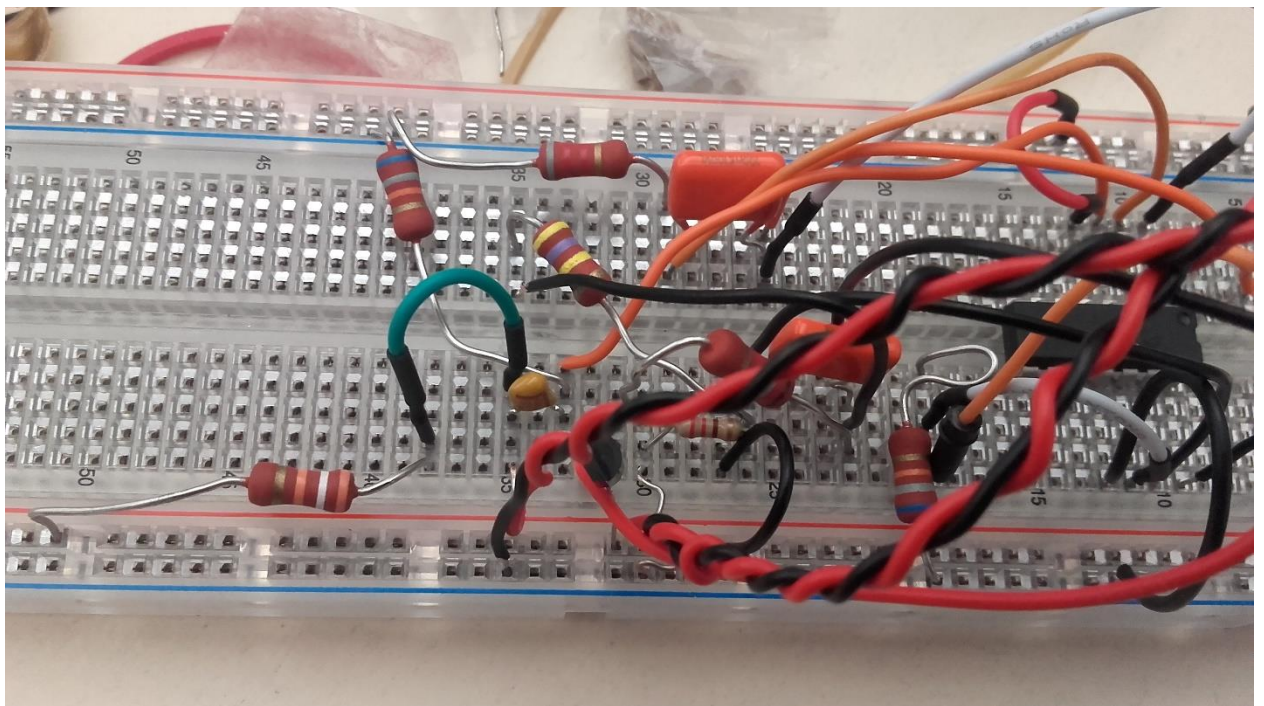
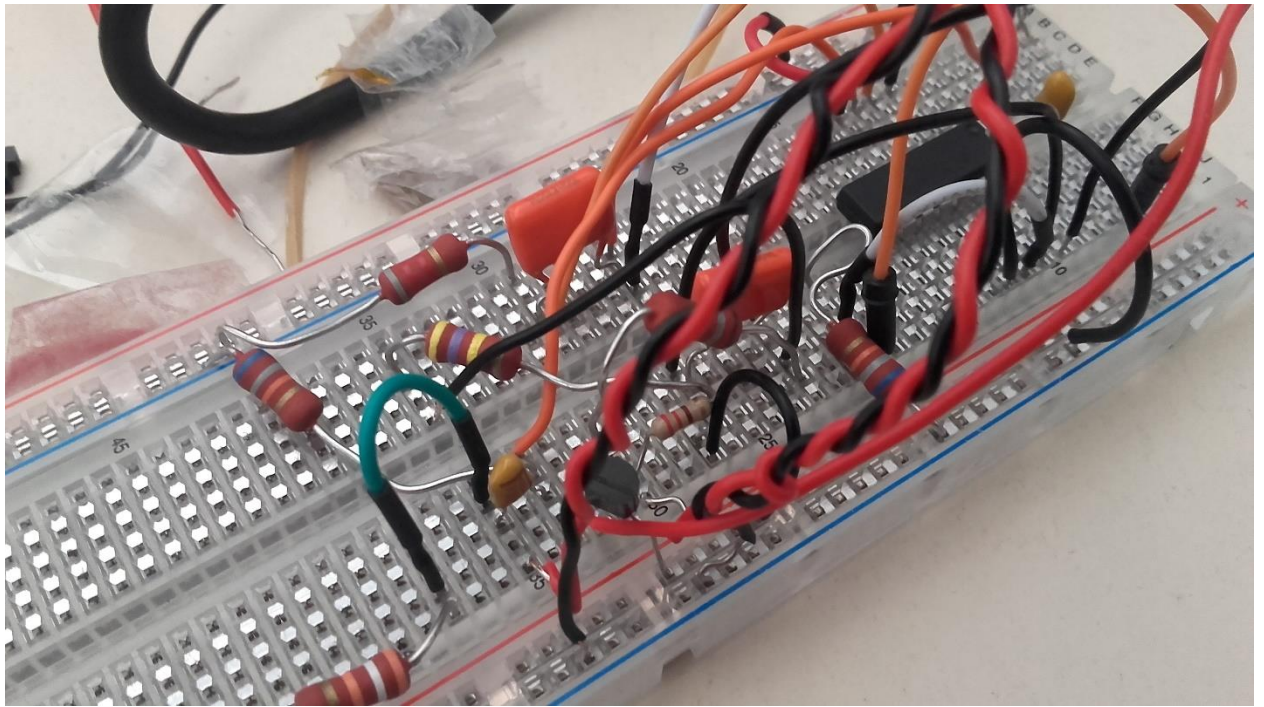
Projet

Fondamentaux Scientifiques

Cardiofréquence-mètre

Photos du montage :

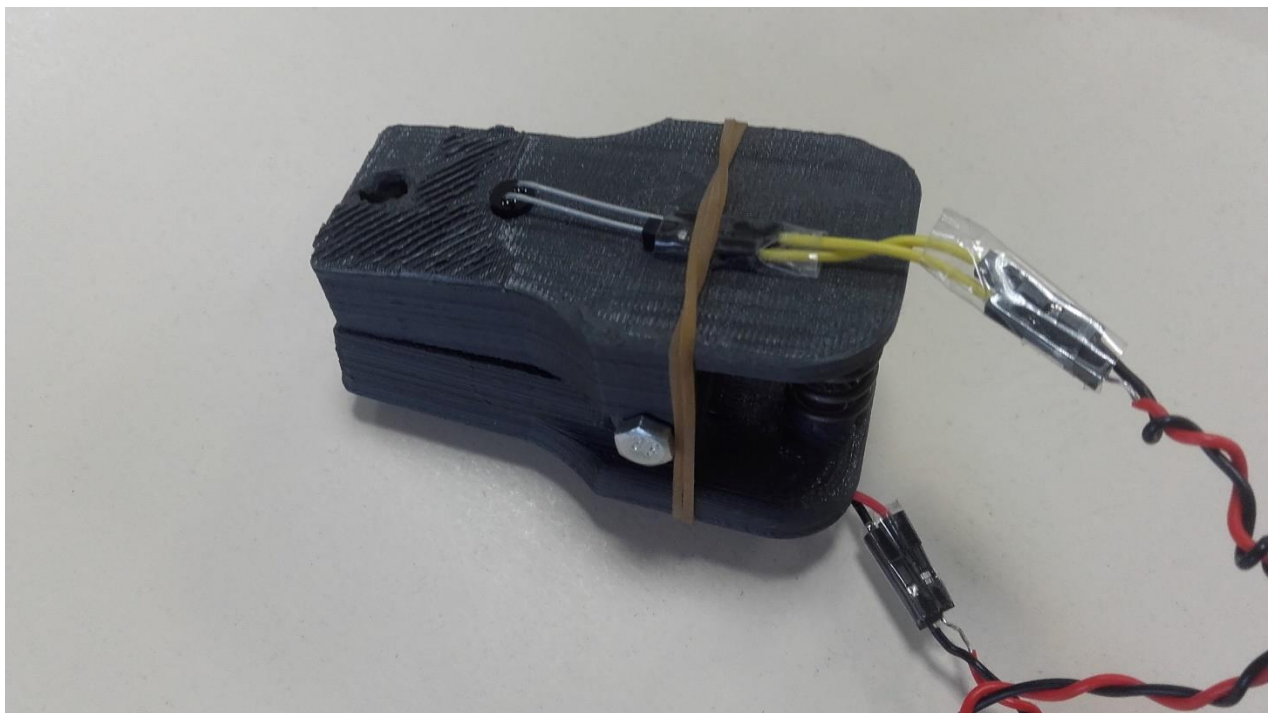
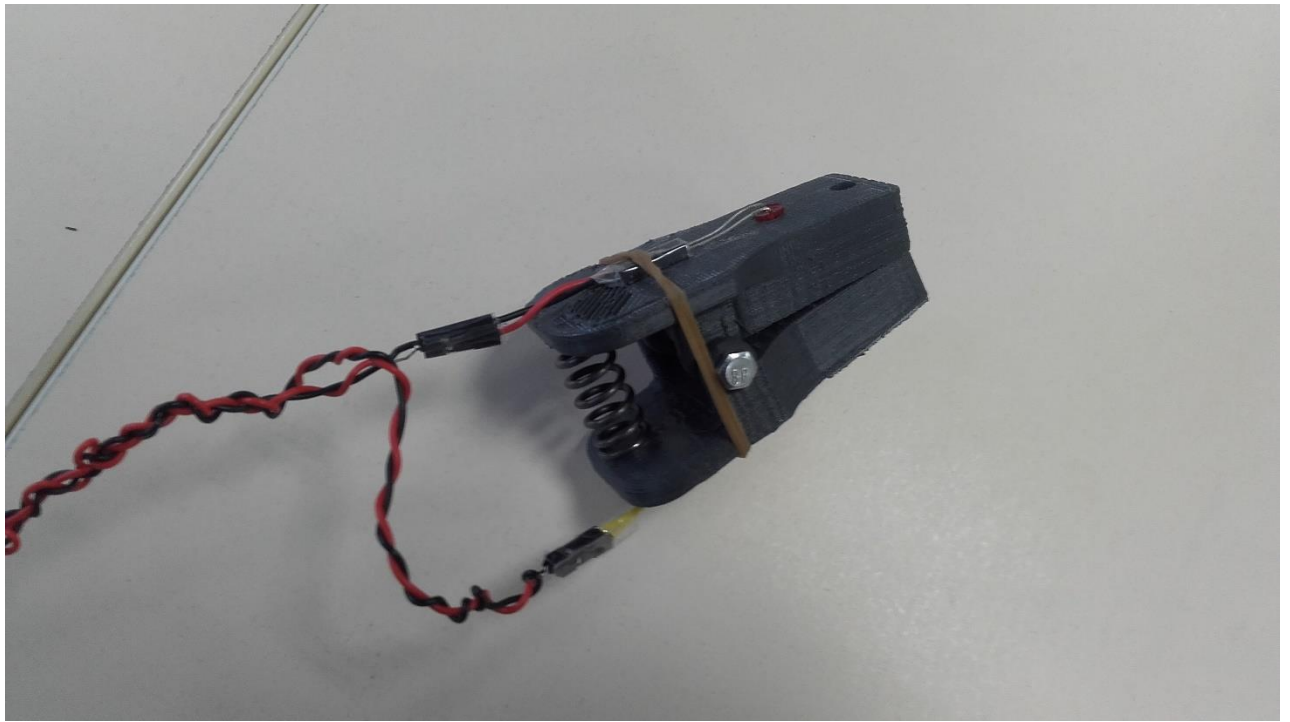




Projet

Fondamentaux Scientifiques

Cardiofréquencemètre



Partie II – Réalisation du programme Arduino

L'objectif de ce programme est de détecter, récupérer et calculer le pouls issu du capteur conçu précédemment.

Les données obtenues sont « censées » être lues par le module Processing (module 3) afin d'être exploitables.

Vous trouverez ci-après une fiche synthétique portant sur le fonctionnement du programme.

Fiche synthétique

Calcul du pouls

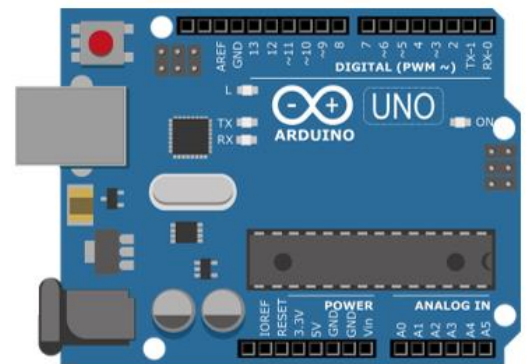
(Module 1)

Objectif : déterminer un pouls précis à partir de valeurs analogiques issues du capteur.

Simule des valeurs analogiques
(remplace le circuit physique)



Valeurs Analogiques



Valeurs Analogiques

Récupération des valeurs
(0 à 1023)
depuis le port A0
(où le capteur de pouls est branché)

Envoi des valeurs analogiques
brutes



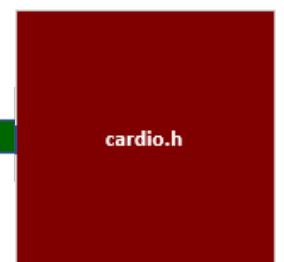
Retourne le pouls calculé à
partir des valeurs analogiques
brutes

Calcule le pouls à
partir des
variations des
valeurs
analogiques



Calcule le pouls

Définition des fonctions



Contient le prototype des fonctions
contenues dans cardio.c

Projet

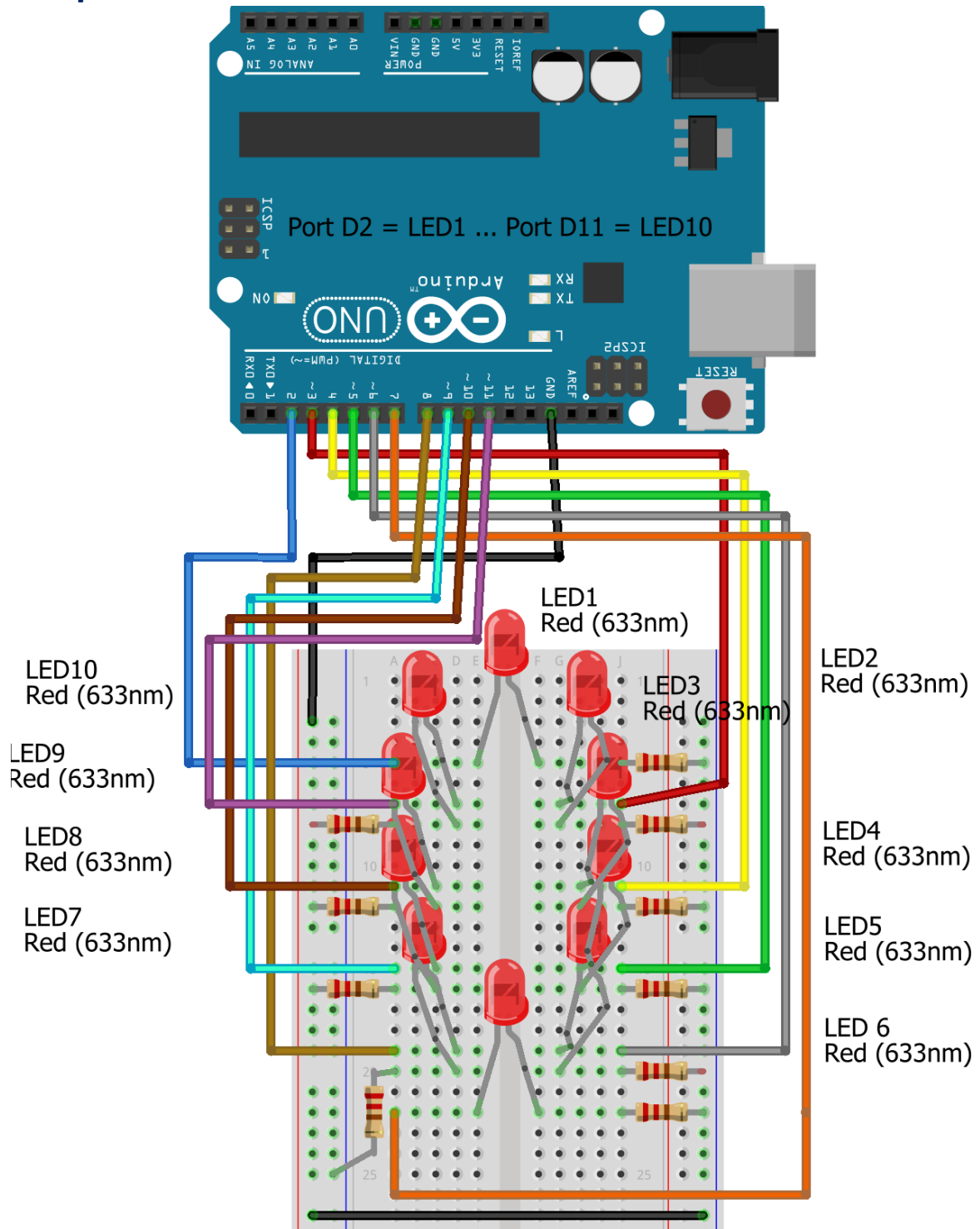
Fondamentaux Scientifiques
Cardiofréquencemètre

• Module 2 – Cœur de LEDs

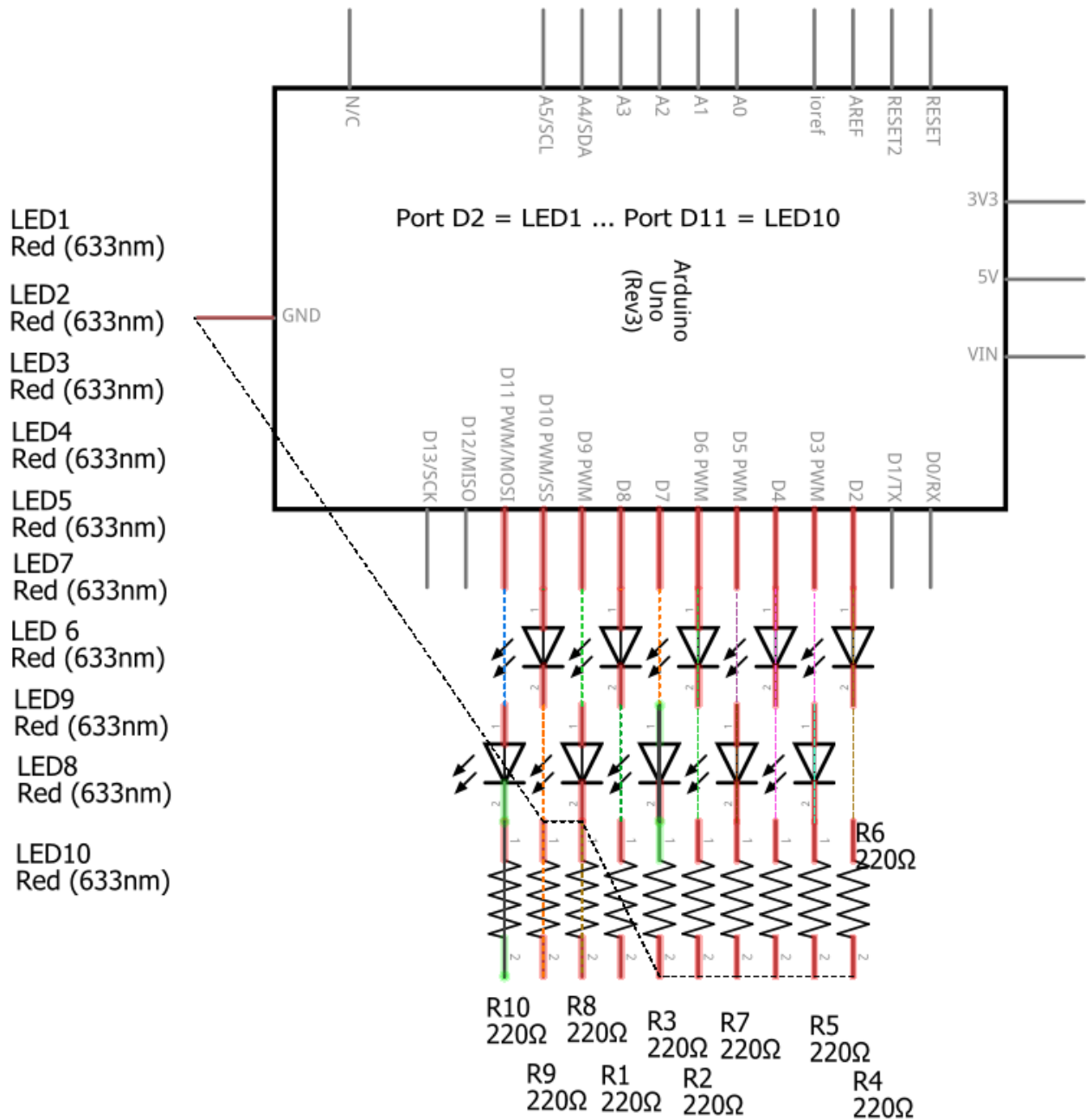
Ce module correspond à l'affichage du pouls sur le cœur de LEDs.

Voici le montage réalisé :

Vue platine :



Vue schématique :

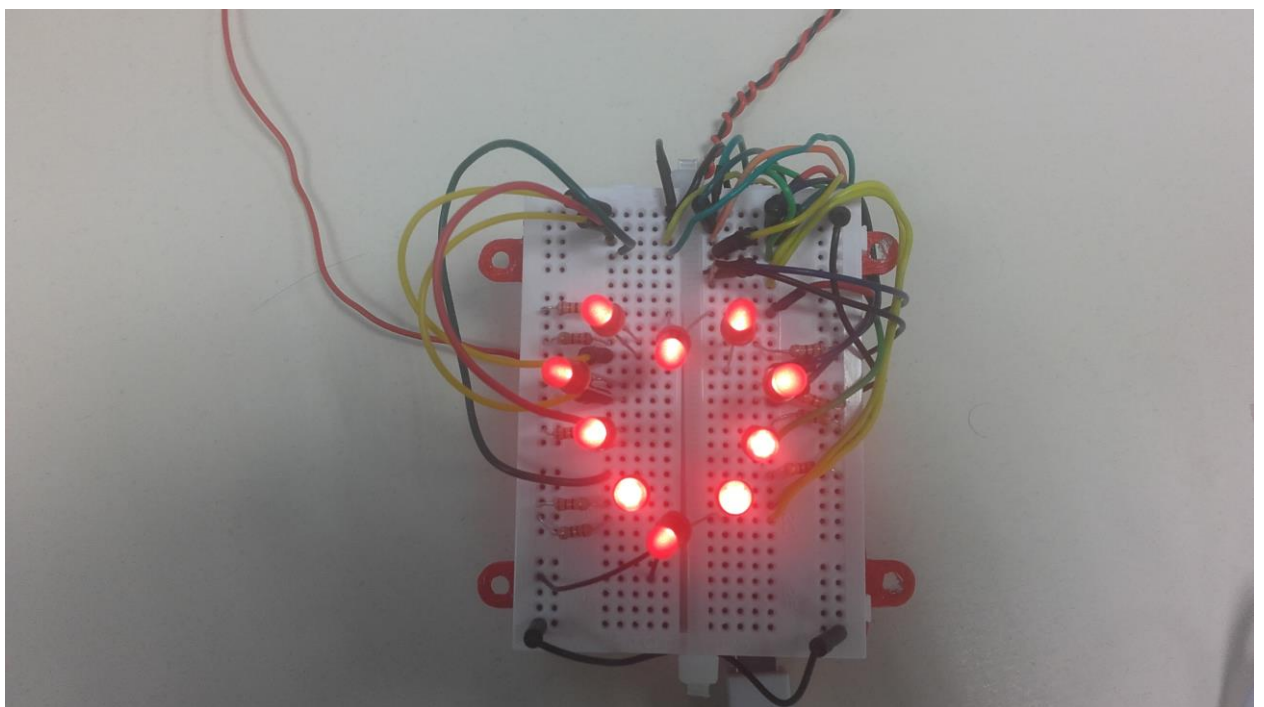
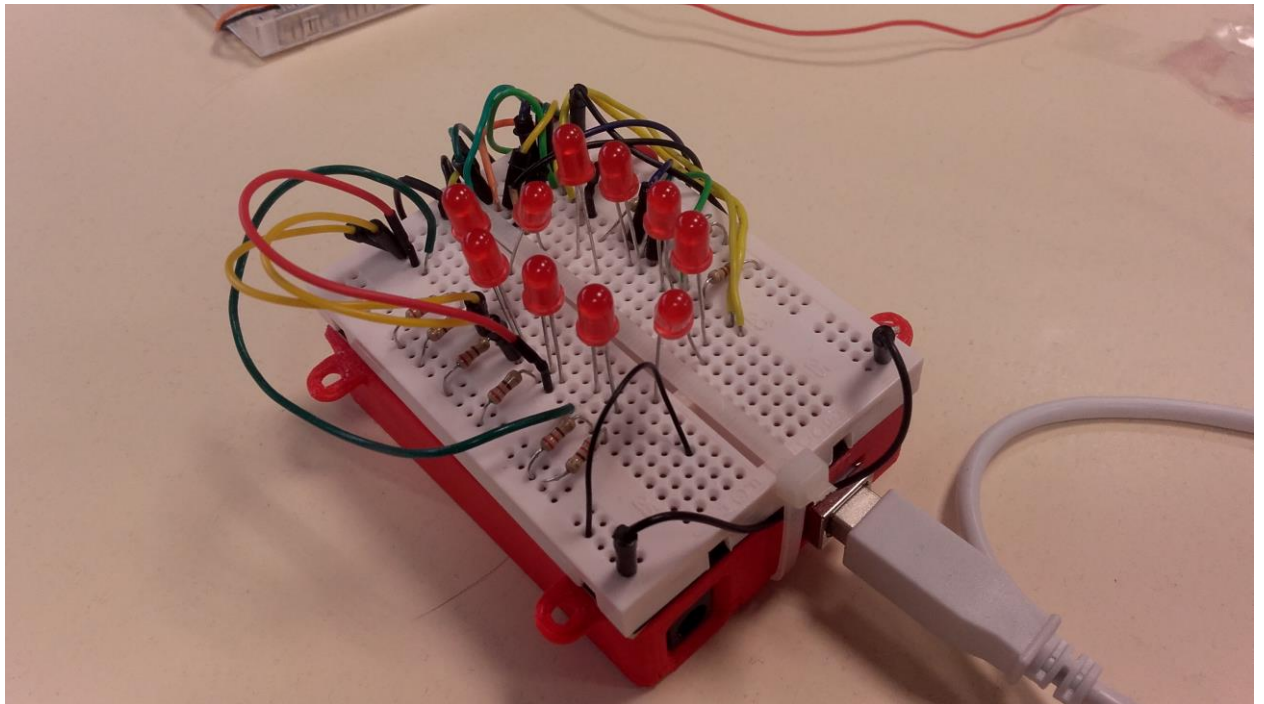


Projet

Fondamentaux Scientifiques

Cardiofréquencemètre

Photos :



- 2.1 – Générateur de code

Dans le cahier des charges, il était mentionné que l'utilisateur *pouvait* choisir le mode d'affichage des LEDs à chaque battement (exemples : toutes les LEDs s'allument, une LED sur deux s'allume, ...).

Nous étions d'abord partis sur la création d'un param.h.ino qui serait automatiquement transféré à la carte Arduino, ce qui fut un succès, mais ne correspondait pas à ce qui était demandé (le contenu de la carte était effacé, et on ne pouvait plus récupérer le pouls).

Nous avons donc changé ce fonctionnement en ne laissant qu'une seule ligne dans le fichier param.h.

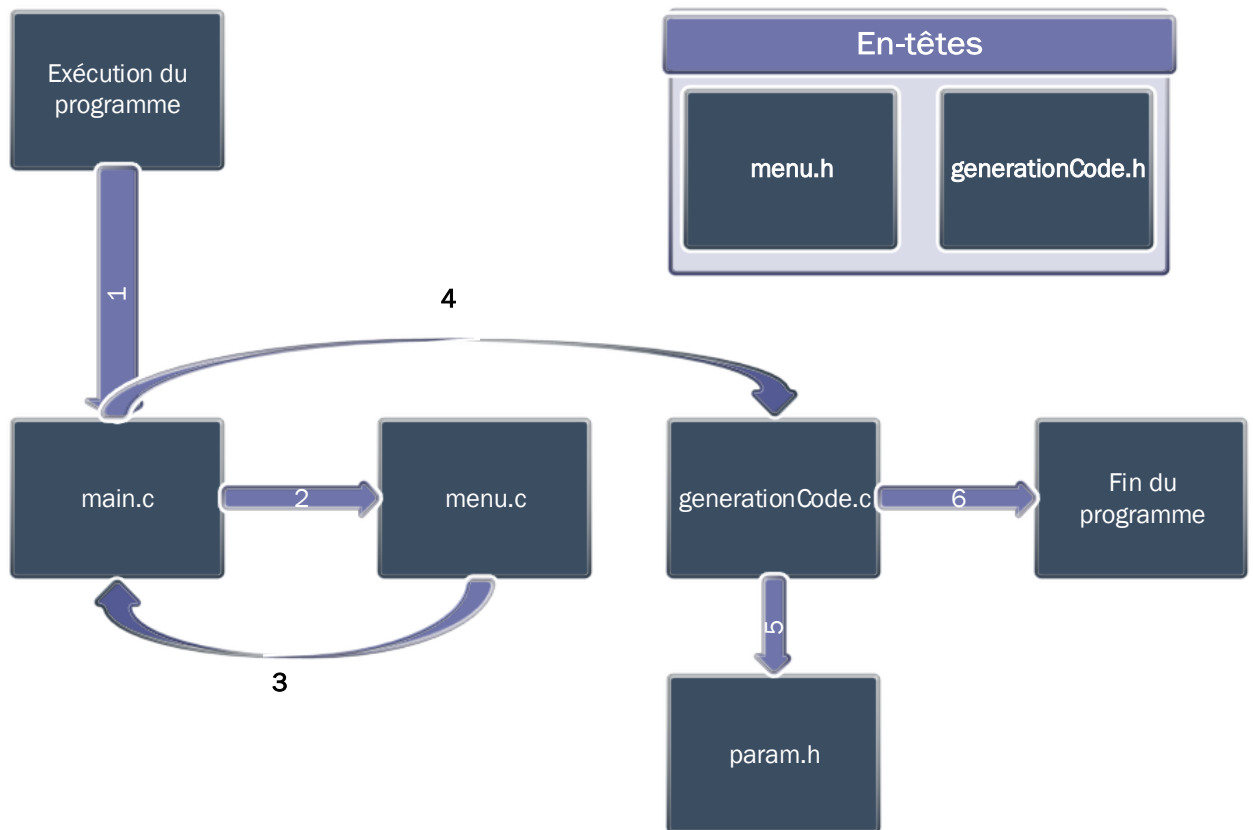
Elle consisterait en la déclaration d'une variable dont la valeur varierait selon le choix de l'utilisateur, et nous nous servirions alors de cette variable.

Un diagramme synthétique est inséré ci-après.

Fiche synthétique

Génération du fichier param.h

Objectif : générer un fichier permettant de configurer le mode d'affichage des LEDs en fonction du choix de l'utilisateur.

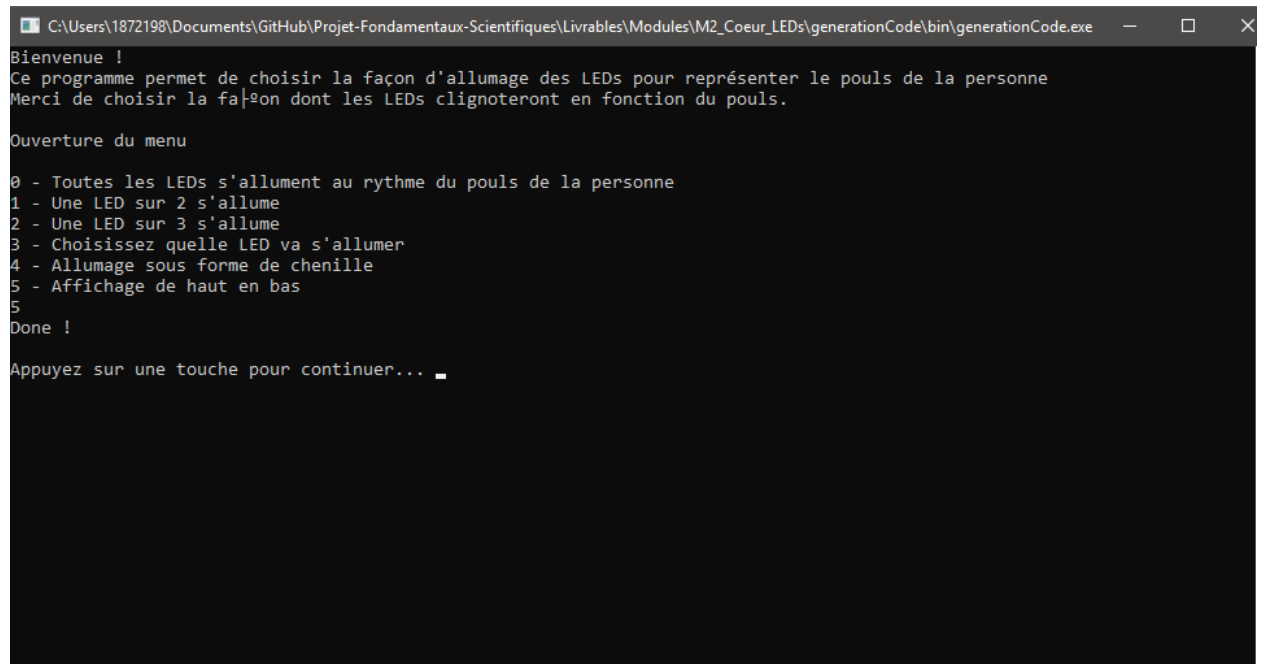


Explications :

- 1 – On rentre dans la fonction principale. Elle appelle le menu dans un premier temps.
- 2 – On récupère le choix de l'utilisateur concernant le mode d'affichage des LEDs voulu
- 3 – On retourne dans la fonction principale
- 4 – La fonction principale appelle la fonction de génération du fichier param.h
- 5 – La fonction génère un fichier param.h contenant la ligne « #define LEDPARAM » suivie de la valeur entrée par l'utilisateur

(Note : On lira la valeur de LEDPARAM pour le mode d'affichage, en incluant bien param.h en en-tête du fichier cœur.c

Captures d'écran :



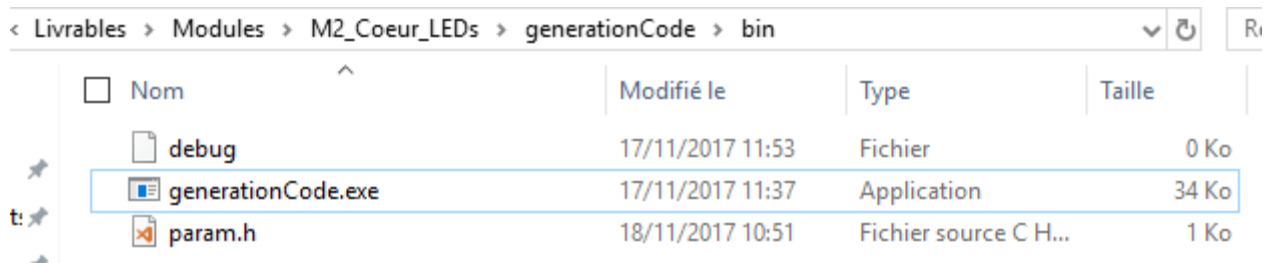
```
C:\Users\1872198\Documents\GitHub\Projet-Fondamentaux-Scientifiques\Livrables\Modules\M2_Coeur_LEDs\generationCode\bin\generationCode.exe
Bienvenue !
Ce programme permet de choisir la façon d'allumage des LEDs pour représenter le pouls de la personne
Merci de choisir la façon dont les LEDs clignoteront en fonction du pouls.

Ouverture du menu

0 - Toutes les LEDs s'allument au rythme du pouls de la personne
1 - Une LED sur 2 s'allume
2 - Une LED sur 3 s'allume
3 - Choisissez quelle LED va s'allumer
4 - Allumage sous forme de chenille
5 - Affichage de haut en bas
5
Done !

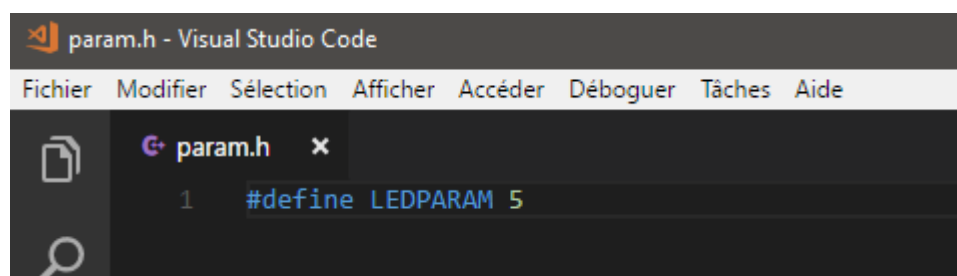
Appuyez sur une touche pour continuer... _
```

Dans l'interface, plusieurs choix sont proposés. L'utilisateur entre le numéro associé au choix voulu puis valide. Ici, on a choisi la 5^{ème} proposition.



Nom	Modifié le	Type	Taille
debug	17/11/2017 11:53	Fichier	0 Ko
generationCode.exe	17/11/2017 11:37	Application	34 Ko
param.h	18/11/2017 10:51	Fichier source C H...	1 Ko

Le fichier param.h a été créé après choix de l'utilisateur . . .



```
param.h - Visual Studio Code
Fichier Modifier Sélection Afficher Accéder Débugger Tâches Aide

1 #define LEDPARAM 5
```

En ouvrant le fichier param.h, on constate que la variable LEDPARAM a été déclarée à 5, comme demandé.

- 2.2 – Programme Affichage Cœur de LEDs

Le fichier param.h a été précédemment créé. Le générateur de code doit obligatoirement être exécuté avant l'exécution du programme d'affichage sur le cœur de LEDs (ou un fichier param.h artificiel doit être créé).

L'objectif est de faire clignoter la ou les LEDs dès qu'un battement est détecté, et de faire l'opération inverse lorsqu'aucun pouls n'est détecté.

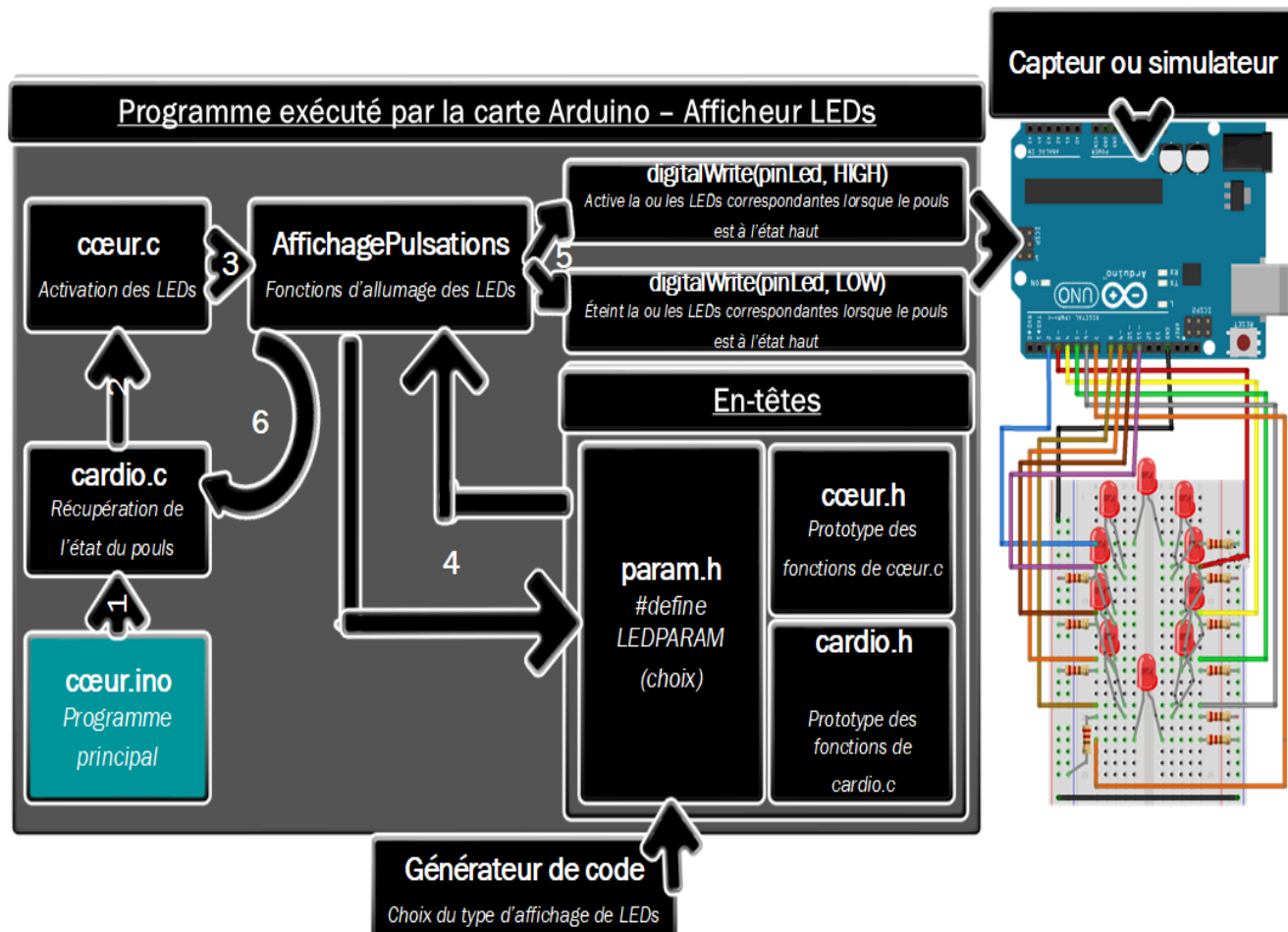
Il est nécessaire de faire appel au loop du main.ino. Il permettra de déterminer lorsque le pouls est à l'état haut (dans ce cas, val_pouls = 1) ou non (val_pouls = 0).

Il faut également prendre en compte le mode d'affichage des LEDs, d'où le fichier param.h inclu en en-tête qui contient un chiffre correspondant au choix de l'utilisateur lors de l'utilisation du générateur de code.

Une synthèse est incluse ci-après.

Fiche synthétique

Module 2 – Afficheur LEDs



Explications :

- 1 – On rentre dans cardio.c où on va récupérer l'état du pouls (0 ou 1)
- 2 – On rentre dans cœur.c
- 3 – On appelle la fonction AffichagePulsations, qui contient les fonctions de chaque mode d'affichage
- 4 - On récupère la valeur de LEDPARAM contenue dans param.h (préalablement généré avec le générateur) et on rentre dans la fonction correspondante à la valeur de LEDPARAM
- 5 – On allume ou non les LEDs en fonction de l'état du pouls et de la fonction d'allumage.

• Module 3 – Génération du fichier CSV (Processing)

Le programme Processing était déjà fourni.

Il récupère le temps regroupe en un fichier le temps écoulé depuis le début de l'exécution du programme avec le pouls calculé, en lisant sur la liaison série.

La carte Arduino doit donc renvoyer d'abord le temps (avec la fonction *millis*), envoyer un « ; » puis envoyer le pouls calculé.


Les résultats sont enregistrés dans un fichier .CSV.

Cela présente plusieurs intérêts lorsque l'on ouvre le fichier avec un tableau (Excel) , les données sont séparées dans des colonnes différentes, pouvant faciliter le tri (le « ; » sert de séparateur). On peut alors choisir de traiter ces données directement avec le tableur.

Captures d'écran :

```
0.0001;66
0.0002;90
0.0003;96
0.0004;136
0.0005;113
0.0006;134
0.0007;79
0.0008;158
0.0009;72
0.001;176
0.0011;210
0.0012;176
0.0013;140
```

Ouverture du fichier CSV avec un éditeur de texte basique



	A	B
1	0	54
2	0.0001	66
3	0.0002	90
4	0.0003	96
5	0.0004	136
6	0.0005	113
7	0.0006	134
8	0.0007	79
9	0.0008	158
10	0.0009	72
11	0.001	176
12	0.0011	210
13	0.0012	176
14	0.0013	140

Ouverture du fichier CSV avec un tableur

- Module 4 – Lecture et traitement des données

L'objectif est de lire et de traiter les données enregistrées.

On a alors décidé de créer une structure nommée « tableau » à deux entrées.

Dans l'une, on entrera la valeur du temps (en ms), et dans l'autre la valeur du pouls associé.

On ouvre donc le fichier Battements.csv et on le parcourt, pour rentrer les valeurs dans la structure.

Une fois les données récupérées dans la structure, l'utilisateur peut alors choisir l'action suivante à effectuer : trier, ou rechercher une donnée.

Il est possible d'afficher toutes les lignes dans l'ordre du fichier CSV, mais il est également possible de les trier.

Pour le tri, on utilise un tri à bulles (qui ne sera pas efficace pour un nombre plus important de données) pour les tests.

On peut alors trier selon le temps ou selon le pouls, décroissant ou non.

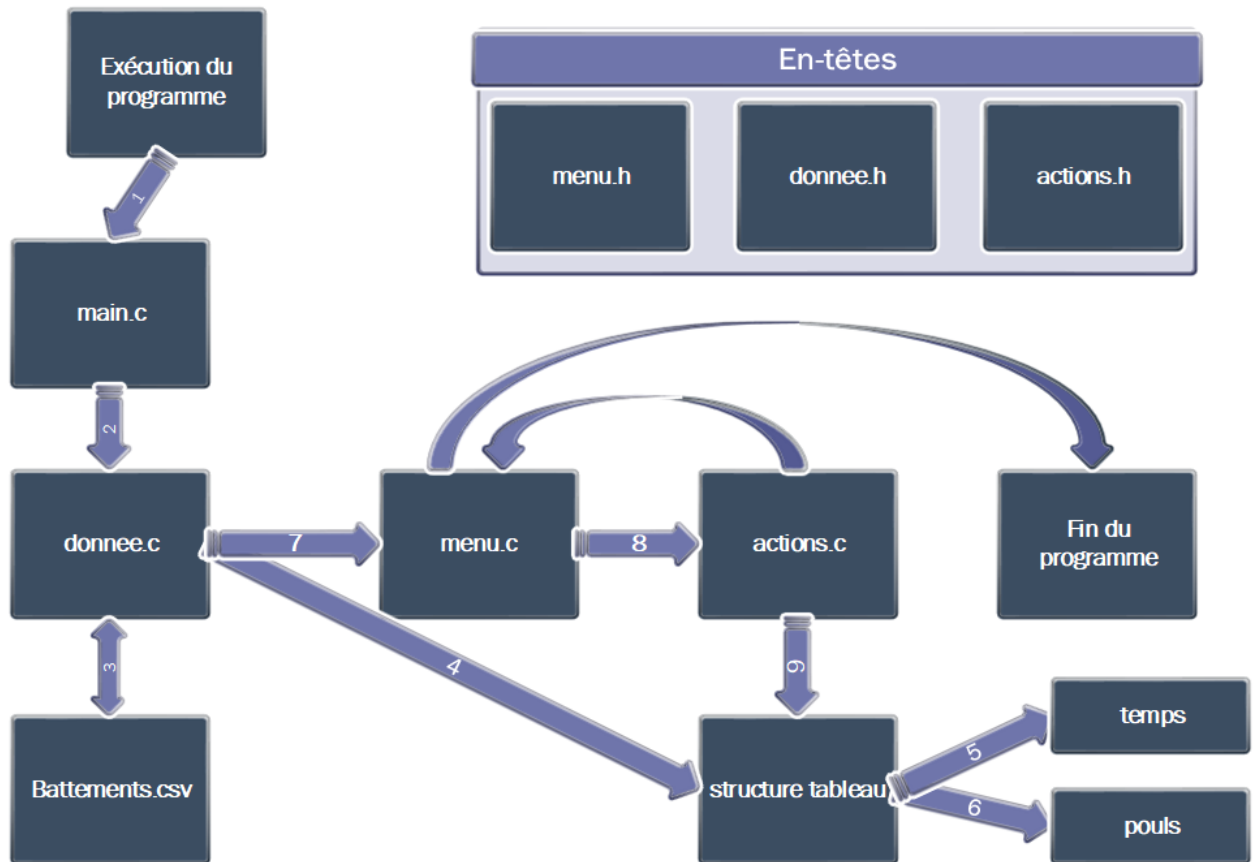
Il est possible d'effectuer une recherche (dichotomique) basée sur le temps.

On peut afficher le pouls moyen, le pouls minimum et le pouls maximum, ou le nombre de lignes dans le fichier.

Une synthèse est disponible ci-après.

Fiche synthétique

Module 4 – Lecture et traitement des données



Explications :

- 1 – On rentre dans la fonction principale. Elle appelle le menu dans un premier temps.
- 2 – La fonction `RemplirTableau` contenue dans `donnee.c` est exécutée dans un premier temps.
- 3 – La fonction accède au fichier `Battements.csv`
- 4 – Le contenu du fichier CSV est placé dans une structure avec un tableau à 2 colonnes :
- 5 - ... une pour le temps (avant le ;) ...
- 6 - et l'autre pour le pouls (après ;).
- 7 – On rentre ensuite dans le menu (`main.c`) où l'utilisateur choisit quel tri/recherche il va effectuer sur les données
- 8 – On rentre dans la fonction correspondant au choix de l'utilisateur, contenue dans `actions.c`
- 9 – La fonction choisie accède à la structure et effectue les opérations demandées.
- 10 – On retourne au menu une fois l'opération terminée. L'utilisateur peut choisir d'effectuer d'autres actions s'il le désire.
- 11 – A la demande de l'utilisateur, on quitte le programme.