RAPPORT D'APP ELECTRONIQUE

Equipe G10B

PAUL CHANSOU
JULIEN COLOMBAIN
CYPRIAN CUNIN
GABRIEL ENRIQUEZ
CELIA HOULETTE
CAROLINE YAN

2019 - 2020

Sommaire

Introduction

Nous objectif est de produire un système qui effectue des mesures psychotechniques en vue d'une utilisation par des auto-écoles ou des centres médicaux, afin de mesurer l'aptitude d'individus à la conduite.
Le système produit se devra d'être ergonomique(simple d'utilisation) et transportable (dimensionnement du système).

Notre système nécessitera des compétences tant électroniques, informatiques, que télécoms, et s'effectuera sur un an.

La première partie de notre travail, étendue sur une durée de 13 semaines, effectuée en ce moment est l'électronique. Nous allons, dans cette partie, effectuer plusieurs montages: un capteur de fréquence cardiaque, un capteur de température, un microphone pour mesurer la qualité de reconnaissance vocale de l'utilisateur, un montage à LED pour mesurer le temps de réaction à un signal lumineux inattendu, et enfin un montage qui nous permettra de mesurer le temps de réaction à un signal sonore inattendu.

Nous avons déjà réalisé 3 montages parmi les 5 à effectuer; il s'agit du :

- montage à microphone
- capteur de température
- capteur de fréquence cardiaque

1. Cahier des charges

Montage à microphone

Notre objectif est de déterminer la fréquence d'un son émis par une personne avec une précision de + ou - 10 Hz. Cette fréquence doit être comprise entre 130 et 4000 Hz.

Capteur de fréquence cardiaque

Notre objectif est de mesurer le niveau de stress de l'utilisateur en déterminant sa fréquence cardiaque à l'aide d'un phototransistor. Une fréquence cardiaque élevée ou irrégulière sera synonyme de stress. Nous allons chercher à mesurer des fréquences comprises entre 40 et 140 battements par minute, avec une précision de + ou - un battement par minute.

Capteur de température

Notre objectif est de mesurer le niveau de stress de l'utilisateur en mesurant la température superficielle de sa peau. Un haut niveau de stress se traduira par une augmentation de cette température. Le capteur devra pouvoir mesurer une température comprise entre 20 et 40 °C, avec une précision de + ou - 0,2 °C.

2. Mesure de la qualité de reconnaissance de tonalité

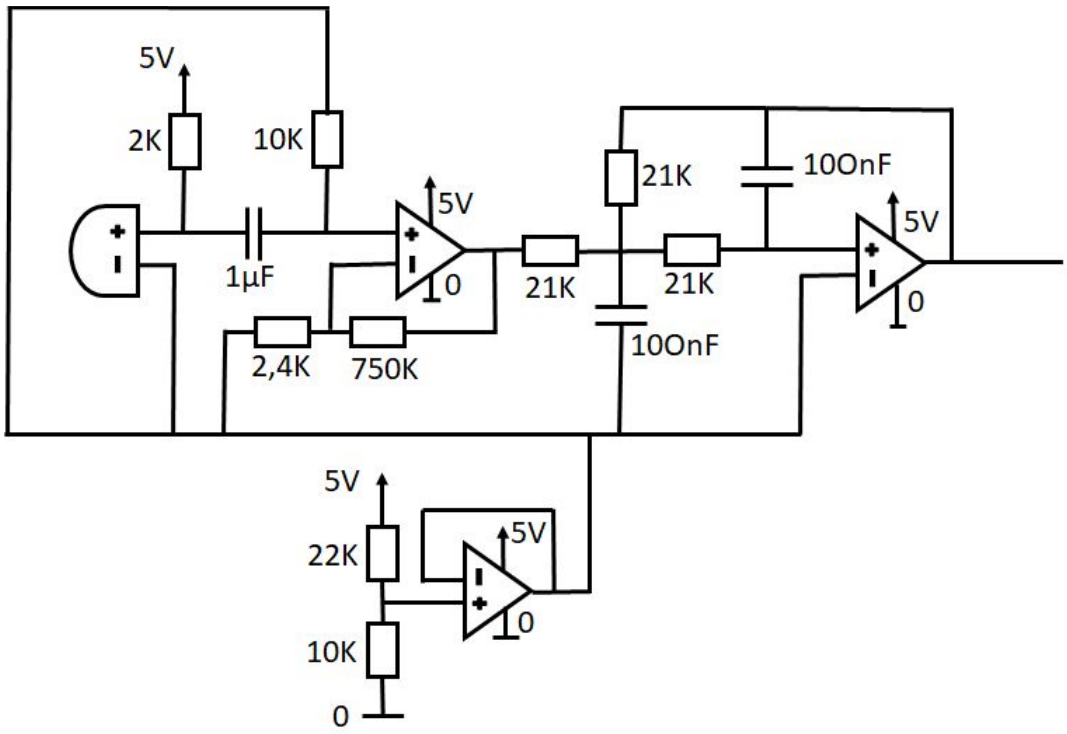
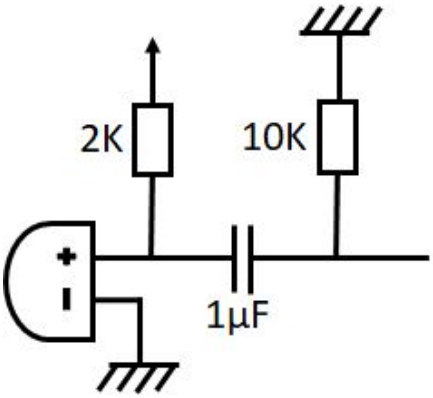


Schéma final du montage avec le microphone

Nous avons réalisé :

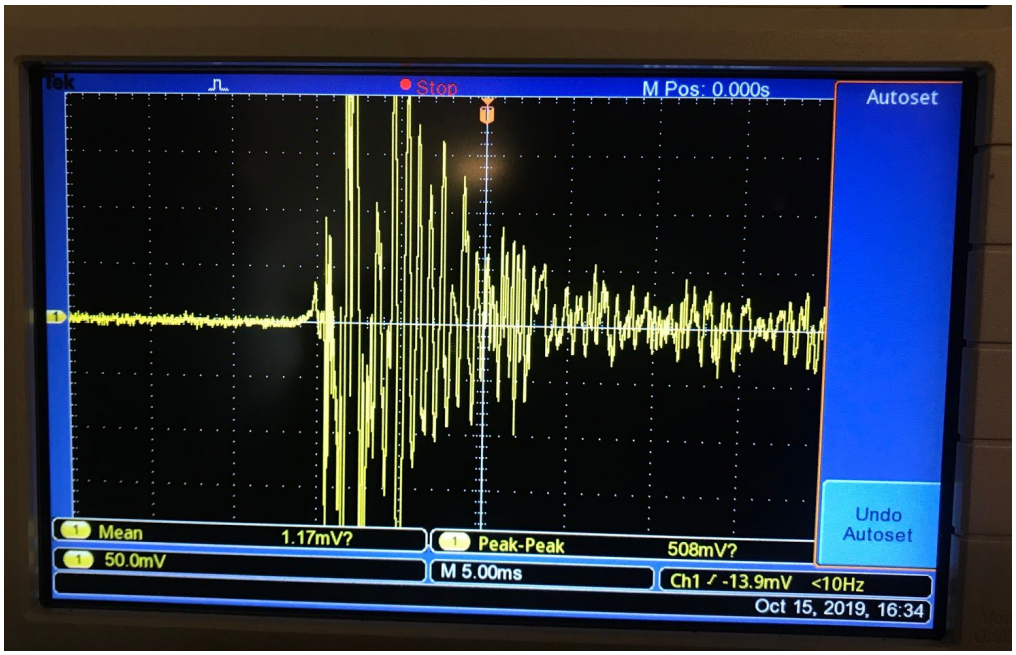
1) Un filtre passe-haut
2) Un amplificateur non-inverseur
3) Un filtre passe-bas du second ordre
4) Une masse virtuelle
5) Les soudures

2.1. Le filtre passe-haut



La fréquence de coupure de ce filtre est de 16 Hz. Il a pour but de supprimer la composante continue de notre signal de sortie.

On obtient le signal suivant :



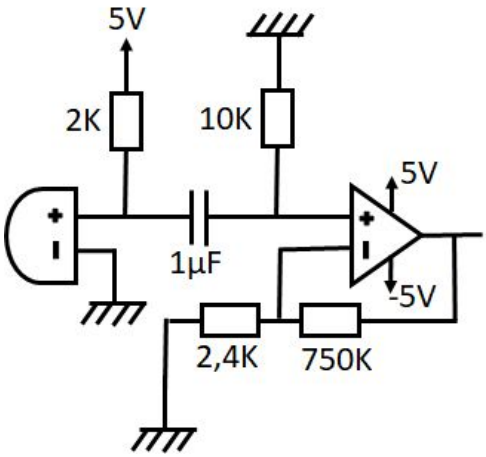
Signal obtenu avec le microphone et le filtre passe-haut

Les variations correspondent au bruit que l'on a réalisé pour faire des tests.

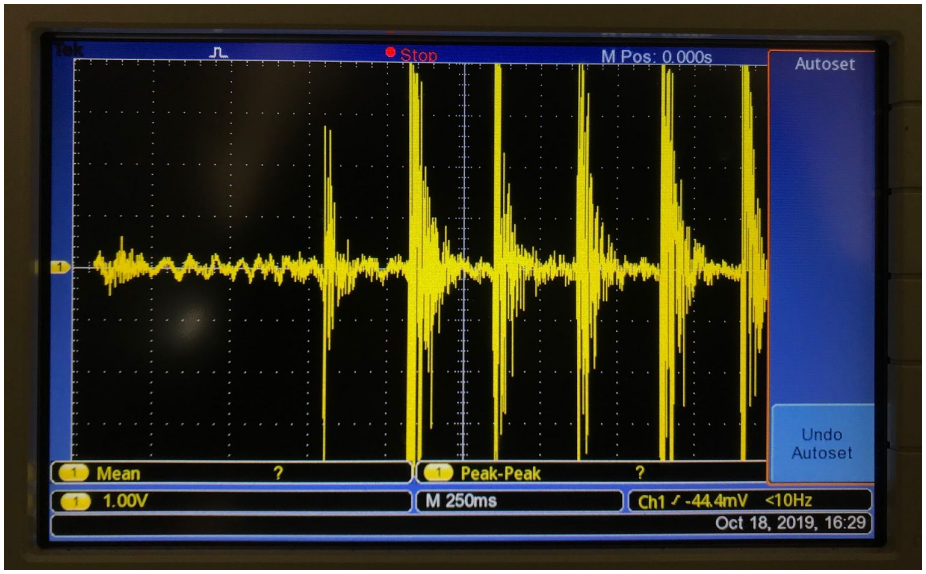
Cependant, notre signal est trop faible (de l'ordre de 50 mV) pour pouvoir être interprété par la carte TIVA. Nous devons donc l'amplifier.

2.2. L’amplificateur non-inverseur

L’amplificateur que nous avons réalisé va, comme son nom l’indique, amplifier le signal de sortie avec un gain de 300.



On obtient la sortie suivante (les pics correspondent à des claquements de main) :

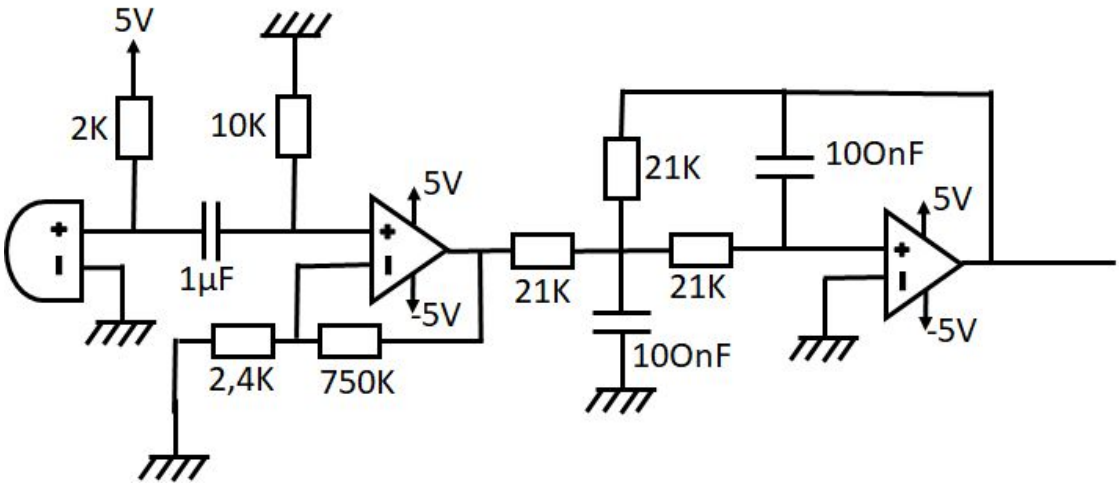


Signal obtenu avec l’amplificateur non-inverseur

On remarque bien que le signal est maintenant de l’ordre du Volt alors qu’il était en sortie de l’amplificateur de l‘ordre de 50 mV.

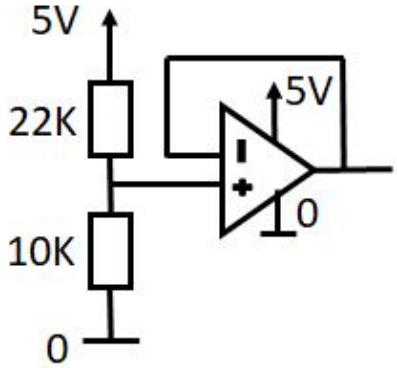
2.3. Le filtre passe-bas du second ordre

Nous avons choisi de réaliser une filtre passe-bas actif du second ordre pour plus de précision. Ce filtre va permettre de supprimer le bruit qui parasite notre signal. Dans le cahier des charges, il est inscrit que l'on souhaite mesurer des fréquences allant jusqu'à 4000 Hz, nous avons donc fixé la fréquence de coupure du filtre à 5 KHz.



Pour s'assurer que notre filtre fonctionne bien, nous avons généré un son de 6 KHz en face du micro et nous avons ensuite visualisé la sortie à l'oscilloscope. On a bien que pour cette fréquence supérieure à notre fréquence de coupure, le signal reste "plat".

2.4. La masse virtuelle

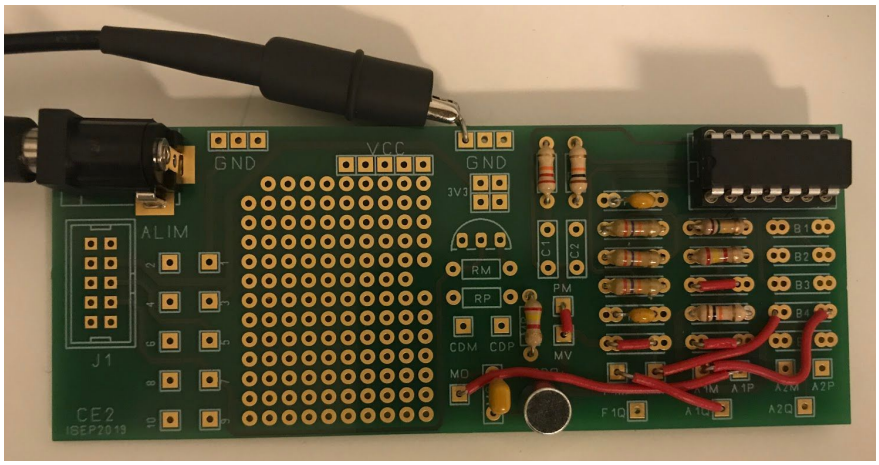


La masse virtuelle s'effectue par l'intermédiaire d'un pont diviseur de tension suivi d'un montage suiveur :

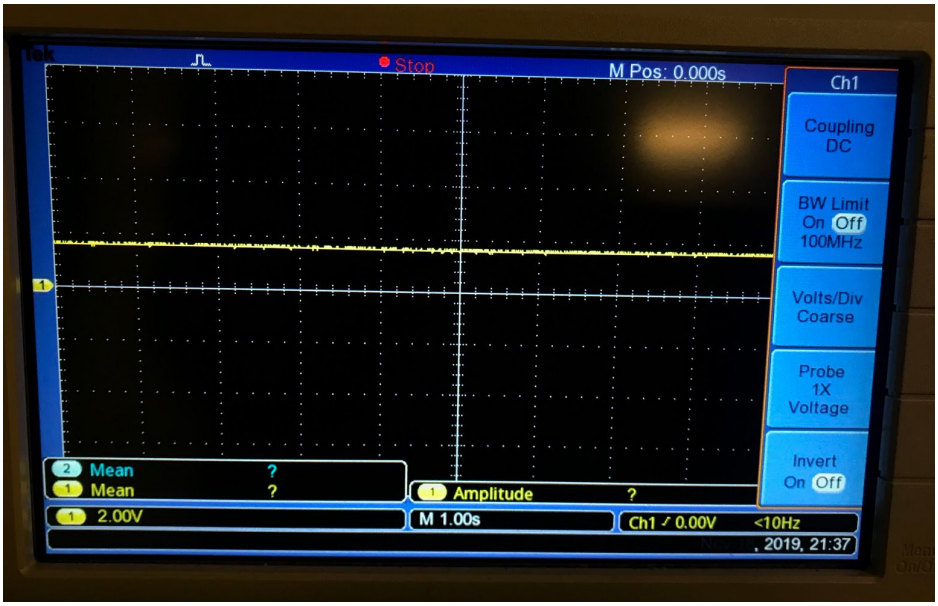
Elle va permettre de centrer notre signal de sortie autour de Vcc/2 (ici Vcc =5V) et ainsi d'avoir un signal compris entre 0 et Vcc au lieu d'entre - Vcc et + Vcc. Cela va permettre à notre signal de sortie d'être exploitable par la carte TIVA.

2.5. Les soudures

Notre montage fonctionnant bien, nous avons soudé nos composants à la carte.

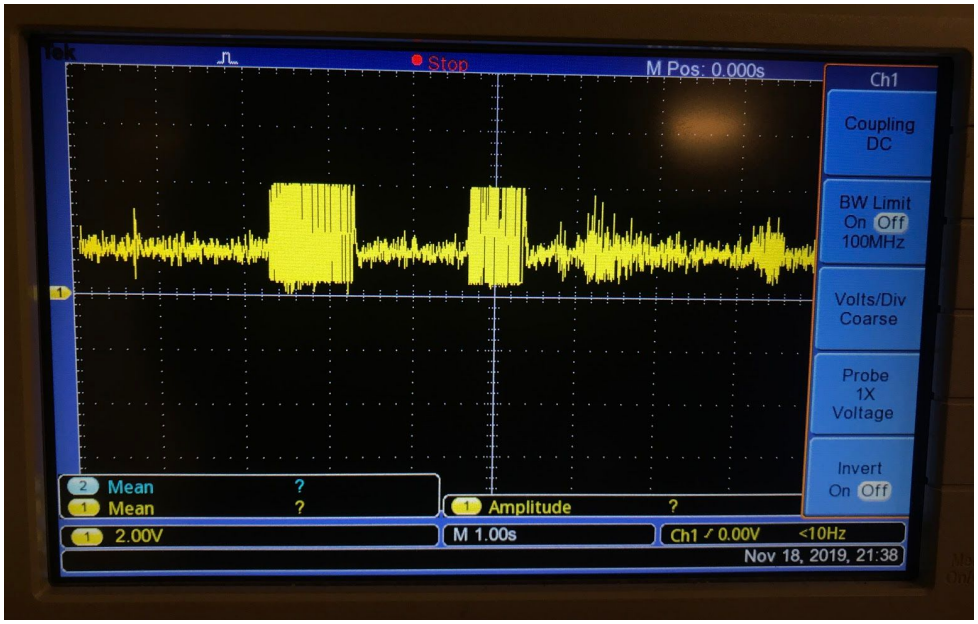


Cependant, il semblerait que suite à cela, le filtre passe-bas du micro ne fonctionne plus.
En effet, lorsqu'on visualise le signal en sortie du micro, on obtient la sortie suivante :



Le signal est très faible, trop faible pour être exploité par la carte TIVA, mais il réagit tout de même au bruit.

En sortie de l'amplificateur, on visualise le signal suivant :



Notre signal est bien amplifié (de l'ordre de 4V), et le micro réagit bien au bruit comme on peut le constater avec les pics obtenus sur l'oscilloscope en réponse à un sifflement. Cependant, le bruit parasite beaucoup trop notre signal.

En sortie du filtre passe-bas, on obtient exactement le même signal qu'en sortie de l'amplificateur. Nous en avons donc déduit que le filtre posait problème.
Afin de déterminer l'origine exacte de notre problème, nous avons vérifié les valeurs de nos résistances et condensateurs, vérifié nos soudures, vérifier notre montage point par point, sans trouver de défaut. Nous cherchons encore à déterminer l'origine de ce problème.

Ce qu'il nous reste à faire

Il nous reste donc à déterminer l'origine exacte de la défaillance de notre filtre après l'avoir soudé à la carte électronique. Il nous restera par ailleurs à - lorsque ce problème de filtre sera réglé - interpréter le signal obtenu en sortie par la carte qui devra renvoyer la fréquence du son émis par l'utilisateur du système.

3. Le phototransistor

Afin de mesurer le rythme cardiaque, nous allons chercher à détecter les variations de la circulation du sang à l'extrémité d'un doigt. En effet, l'activité du coeur provoque la circulation du sang à travers l'organisme se traduisant par une variation du volume sanguin à l'extrémité d'un doigt. Nous allons exploiter le fait que cette variation engendre une variation de la transparence du doigt à la lumière visible, en utilisant un phototransistor pour les détecter.

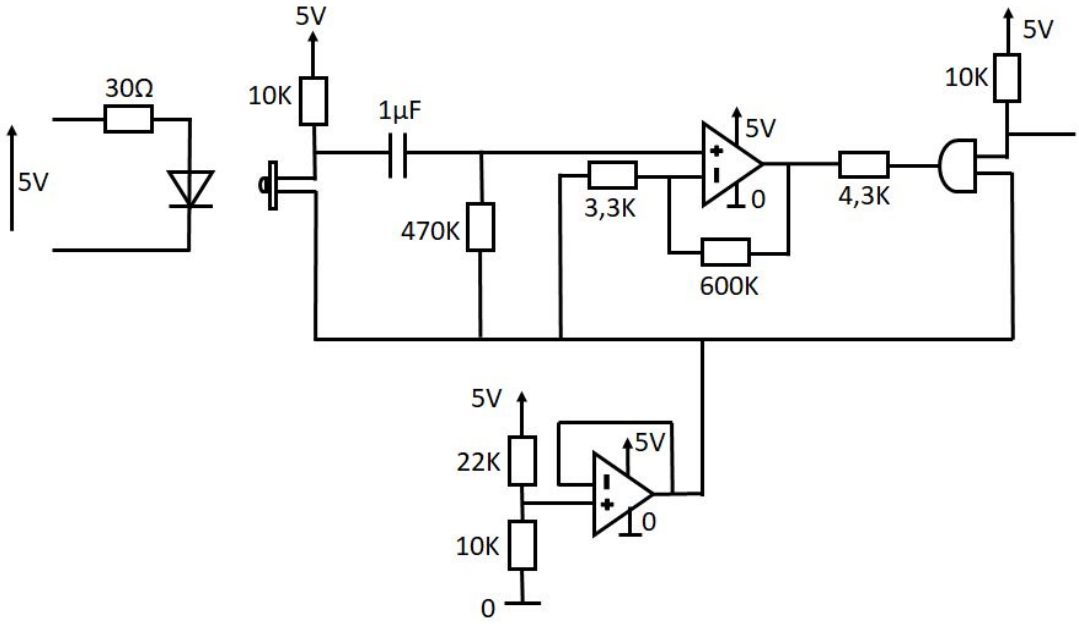


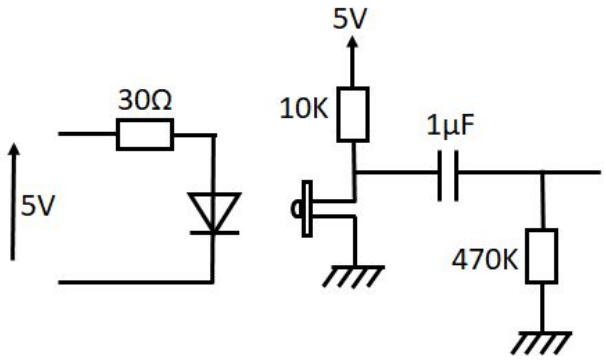
Schéma du montage

Nous avons réalisé :

1) Un filtre passe-haut
2) Un amplificateur non-inverseur
3) Un montage avec transistor
4) Une masse virtuelle
5) Les soudures

3.1. Le filtre passe-haut

Dans le cahier des charges, il est indiqué que l'on doit pouvoir mesurer des fréquences cardiaques comprises entre 40 et 140 bpm soit entre 0,7 et 2,3 Hz. On souhaite par conséquent que toutes les fréquences inférieures à 40 Hz n'apparaissent pas sur notre signal de sortie et c'est pourquoi nous avons utilisé un filtre ayant une fréquence de coupure de 0,3 Hz. (En plaçant la résistance du phototransistor à l'alimentation 5V et non à la masse, on fait en sorte d'avoir des impulsions positives.)

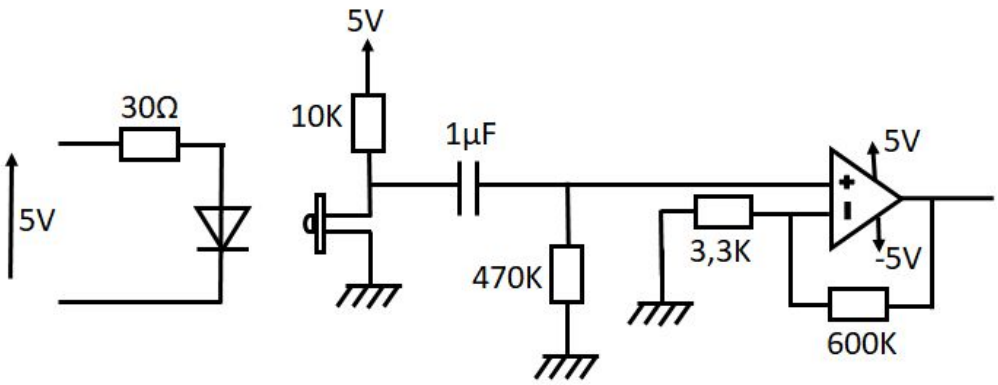


Signal de la fréquence cardiaque avec le filtre passe-haut

On peut ici nettement discerner les battements du coeur et lire la fréquence cardiaque. Cependant, le signal est trop faible puisqu'il est de l'ordre de 200 mV et que l'on souhaite obtenir un signal de l'ordre du Volt et c'est pourquoi, comme nous l'avons fait pour le micro, nous allons l'amplifier.

3.2. L’amplificateur non-inverseur

Afin d’amplifier notre signal, nous avons réalisé un amplificateur non inverseur (pour que nos impulsions restent positives) avec un gain de 200 :

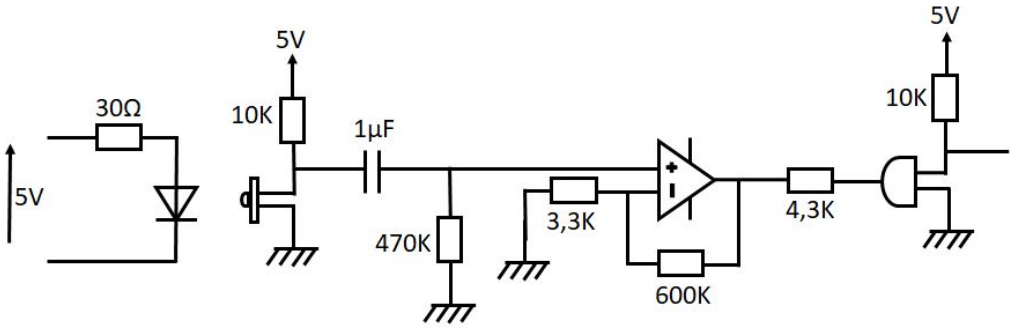


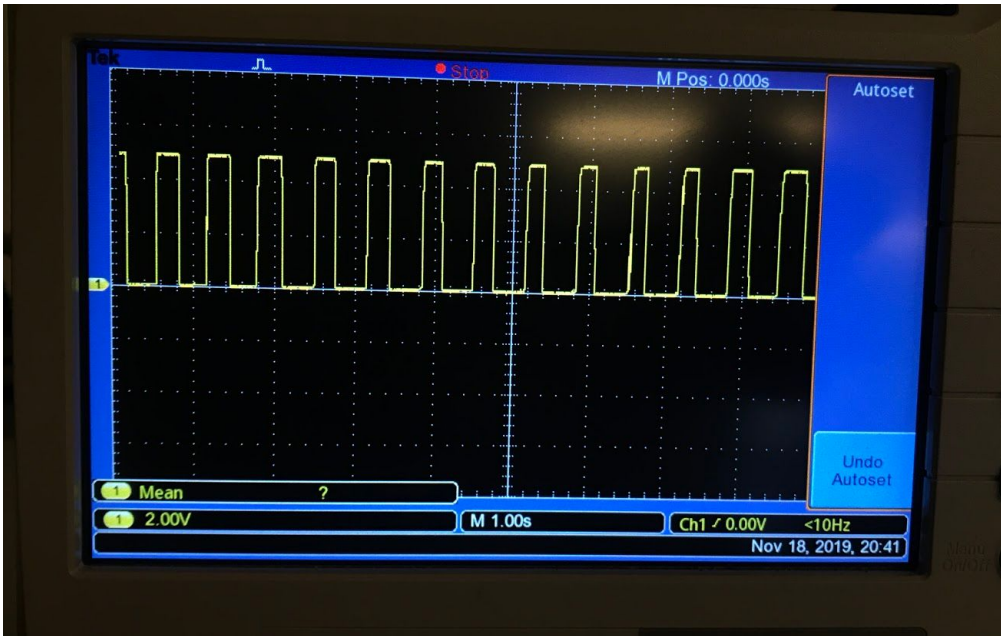
Signal de la fréquence cardiaque avec l’amplificateur non-inverseur

On obtient donc bien le même signal que précédemment, mais amplifié puisqu’on a à présent du 4V crête à crête. Notre signal n’étant pas parasité par le bruit et étant suffisamment “lisse”, nous avons décidé de ne pas mettre de filtre passe-bas en sortie de l’amplificateur.

3.3. Un montage avec transistor

À notre montage précédent, nous avons ajouté un autre montage simple avec un transistor qui va nous permettre d'obtenir en sortie un signal carré exploitable par la carte TIVA :





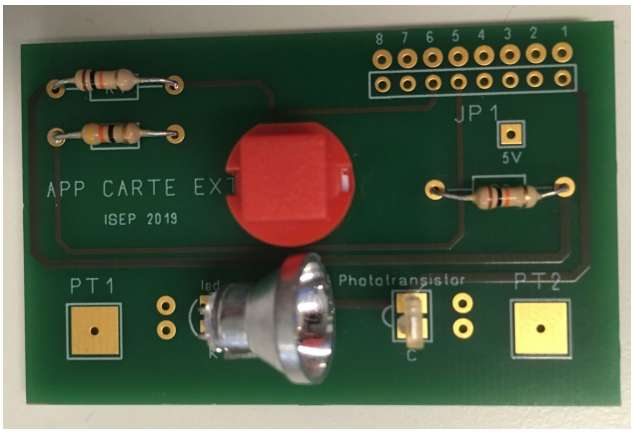
Signal de la fréquence cardiaque avec le transistor

3.4. Une masse virtuelle

De la même manière que pour le micro et pour les mêmes raisons, nous avons créer une masse virtuelle.

3.5. Les soudures

Notre montage fonctionnant, nous avons commencé à souder et avons notamment soudé le phototransistor et la LED à la carte.

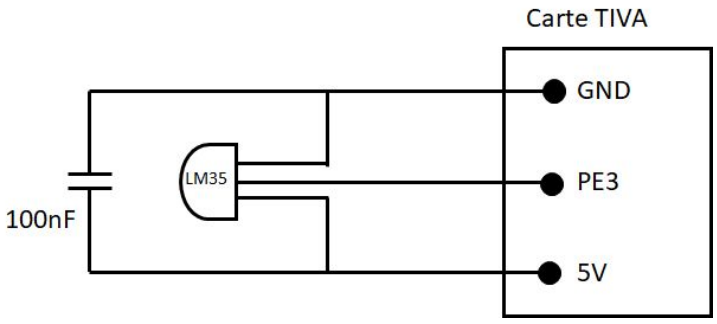


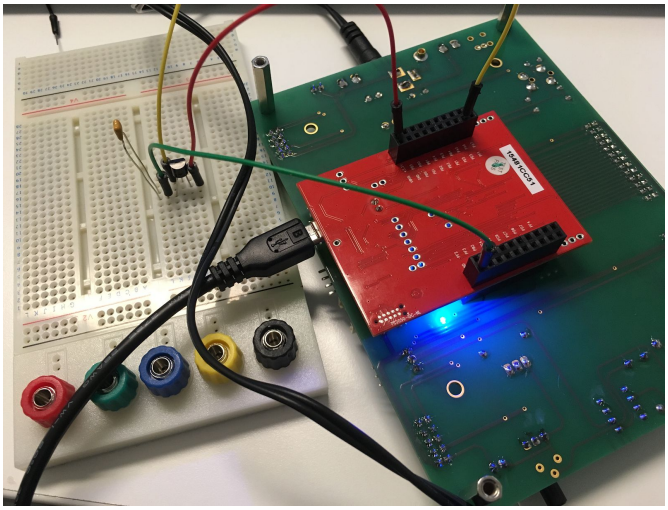
<u>Ce qu'il nous reste à faire</u>

Nous n'avons pas encore terminé de souder la totalité de notre montage à la carte. Il nous restera par ailleurs à tester ce dernier une fois terminé afin de vérifier qu'il fonctionne avec la carte. Enfin, il nous restera à interpréter le signal obtenu par la carte TIVA qui devra notamment renvoyer la fréquence cardiaque en temps réel.

4. Le capteur de température

Afin de mesurer le niveau de stress, nous utilisons le capteur de température. On réalise le montage suivant :





Sur la datasheet du capteur de température LM35 que l'on utilise, il est indiqué que la tension en sortie du capteur est proportionnelle à la température et qu'un degré Celsius correspond à un tension de 10 mV.

Cependant, avec la carte TIVA, on utilise la fonction *analogRead*() pour lire la valeur de la sortie analogique PE3, et cette fonction renvoie non pas une tension comprise entre 0 et 5V comme on le souhaiterait, mais une valeur comprise entre 0 et 1023 pour une température comprise entre 0° et 100° C. Il nous a donc fallu convertir cette valeur de sortie analogique en température.

Pour cela, nous avons fait afficher la valeur renvoyée par la fonction *analogRead()* pour la température de la pièce dans laquelle nous nous trouvions et nous avons ensuite mesuré la tension en sortie du capteur. Nous avons trouvé que la valeur 345 correspondait à 0,22 V. Or 0,22 V correspond aussi à une température de 22°C (car 10mV = 1°C).

Ainsi, on en a déduit la conversion suivante en effectuant un simple produit en croix:

température = valeur renvoyée par la carte*22/345

De cette manière, nous avons été en mesure d'afficher la température de la pièce en temps réel. Par ailleurs, en pinçant le capteur avec nos doigts, on a bien une augmentation de la température, et lorsqu'on le relâche, on a bien une diminution jusqu'à stabilisation à la valeur que l'on obtenait initialement. Notre capteur et notre code fonctionnent donc correctement.

Ce qu'il nous reste à faire :

Il nous reste donc à souder le capteur de température à la carte avec des fils pour que ce soit plus simple d'utilisation.

Conclusion :

Nous avons réussi à bien avancer sur 3 de nos montages qui étaient en cours :

Le montage à microphone :

Nous avons réussi à terminer notre montage et à le souder. Il est apparu un problème de fonctionnement après le soudage du montage.
Il nous reste à déterminer la raison du problème et à y remédier.

Le montage à phototransistor :

Nous avons réussi à terminer notre montage, il faut encore qu'on le relie à la carte TIVA, et qu'on le soude.

Le capteur de température :

Nous avons réussi à faire fonctionner notre montage, il ne nous reste plus qu'à le souder.

Dynamique générale du groupe:

Le projet avance bien, et, mis à part un petit problème technique dans le montage à microphone, les montages que nous avions commencés sont proches d'être finis.

Ce bon avancement est dû à une bonne ambiance au sein du groupe et une bonne dynamique de travail, qui nous permet d'échanger autour du projet et d'avancer plus efficacement.

Sur les autres montages envisagés :

Pour la LED, nous nous sommes renseignés sur le code à entrer dans la carte TIVA afin de relier les LEDs à l'ordinateur. Il nous reste à effectuer ce montage.

Annexe :

Codes energia :

Programmation de l'afficheur

```
#define
NB_COL  5
              #define NB_LIG  7


              int Col[] = {10,9,8,6,5};  // Numéro des broches connectées aux
              colonnes de l'afficheur
              int Lig[] = {38,37,36,35,34,33,13};  // Numéro des broches connectées
              aux lignes de l'affficheur


              int Motif1[] =   { 1,0,0,0,1,   // Définition d'une matrice binaire
              représentant le caractère
                               0,1,0,1,0,   // à afficher
                               0,0,1,0,0,
                               0,1,0,1,0,
                               1,0,0,0,1,
                               0,0,0,0,0,
                               0,1,1,1,0};


              void setup()
              {
              int i;


               // ************ Initialisation des ports de commande de l'afficheur
              en sortie ************
              for (i=0;i<NB_COL;i++)
```

```
   pinMode(Col[i], OUTPUT);

 for (i=0;i<NB_LIG;i++)
   pinMode(Lig[i], OUTPUT);
}



// Cet exemple permet d'illuminer la matrice de LED avec un motif prédéfini
// Le principe est d'allumer les LEDs les unes après les autres. La persistence rétinienne fait le reste
// Le symbole TEMP permet de visualiser le procédé (le mettre à 0 pour l'affichage final

#define TEMP  10

void loop()
{
 int i,j;

 for (i=0;i<NB_LIG;i++)
   {
   digitalWrite(Lig[i],LOW);  // Activation de la ligne i (les autres sont éteintes
   for(j=0;j<NB_COL;j++)
     {
     digitalWrite(Col[j],Motif1[i*NB_COL+j]);  // Selon le code défini par le motif allumage de la led
     delay(TEMP);
     digitalWrite(Col[j],LOW);                // de la colonne j puis extinction
     }
```

```
            digitalWrite(Lig[i],HIGH); // désactivation de la ligne i avant
        passage à la ligne suivante
            }
        }
```

Programmation des LEDs :

```
/*
  DigitalReadSerial with on-board Pushbutton
  Reads a digital input on pin 5, prints the result to the serial monitor

  Harware Required:
  * MSP-EXP430G2 LaunchPad

  This example code is in the public domain.
*/


// digital pin 5 has a pushbutton attached to it. Give it a name:
int pushButton = 17;
int ledSens=39;
int randNum=random(1000,10000);
#define ledSens RED_LED


// the setup routine runs once when you press reset:
void setup() {
  // initialize serial communication at 9600 bits per second:
  Serial.begin(9600); // msp430g2231 must use 4800
```

```
 // make the on-board pushbutton's pin an input pullup:
 pinMode(pushButton, INPUT_PULLUP);
 pinMode(ledSens, OUTPUT);
 //time_t t1 = now();
}


// the loop routine runs over and over again forever:
void loop(){
 // read the input pin:
 int buttonState = digitalRead(pushButton);
 Serial.println(buttonState);
   delay(randNum);
   digitalWrite(ledSens,HIGH);
   int time1 = millis();

 if(buttonState==0)
   digitalWrite(ledSens,LOW);
   int time2 = millis();

 // print out the state of the button:
 int result = time2 - time1;
 Serial.println(result + "!");



 // delay in between reads for stability
}
```

Code pour le capteur de température :

```
void setup()
{
Serial.begin(9600);
}

 void loop()
 {
 int valeur_brute = analogRead(A0);
 float temperature_celcius = valeur_brute *330.0/4096.0;
 Serial.println(temperature_celcius);
 delay(250);

 }
```