INSTITUT d'OPTIQUE GRADUATE SCHOOL ParisTech | LEnsE Laboratoire d'Enseignement Expérimental

LASER GAME

Rapport technique

Arthur Pavlovsky
Yohan Peñaranda
Wafaa Bousaid

SOMMAIRE

1 Présentation du système

1.1 Conception générale du système

Le laser Game est «une activité physique où les participants, revêtus habituellement d'une veste à capteurs, se tirent dessus avec des pistolets laser », d'après Wikipédia.
Nous constatons facilement dans un premier temps que notre système est composé en fait de 2 sous systèmes : un système d'émission laser et un système de détection, qui correspondent respectivement au pistolet laser et à la veste à capteur de la définition de Wikipédia. Un troisième sous système s'avère nécessaire pour assurer la récupération et le traitement du signal détecté. Le contrôle de ces systèmes est assuré par une carte Nucléo STM32.

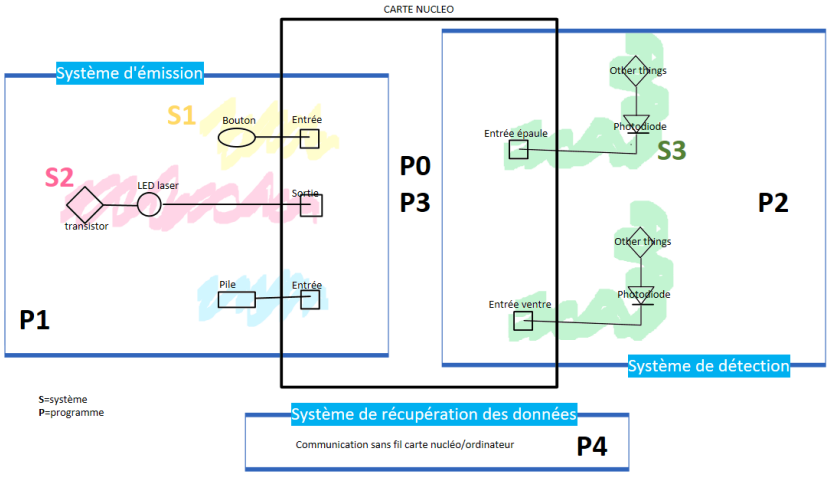


Figure 1: Schéma illustrant les 3 sous systèmes

Le système d'émission est composé d'un laser contrôlé par un bouton. Sa durée du pulse est modulée à l'aide de la carte Nucleo.
Le système de détection est un circuit contenant une photodiode assurant la détection du laser ainsi que la communication du signal détecté à la carte Nucleo. Le système final contient 3 photodiodes : une placée sur le ventre et une placée sur chaque épaule.
Le système de récupération de données est un programme embarqué dans la carte Nucleo assurant le traitement des informations détectées ainsi que le calcul et la communication du nombre de points de chaque équipe avec l'ordinateur sur lequel ce nombre est affiché.

Le calcul des points se fait en respectant les règles suivantes :
• Tire sur le ventre : +50 pts pour l'équipe du tireur , -10 pts pour l'équipe de la victime et et pause de 1 min pour la victime.
• Tire sur l'épaule : +100 pts pour l'équipe du tireur, -10 pts pour l'équipe de la victime et pause de 1 min pour la victime.
• Tire sur un coéquipier : -20 pts pour l'équipe du tireur et 30 sec de pause pour les 2 (tireur et son coéquipier).

Les différents fonctionnalités du système sont décrites par le schéma suivant :

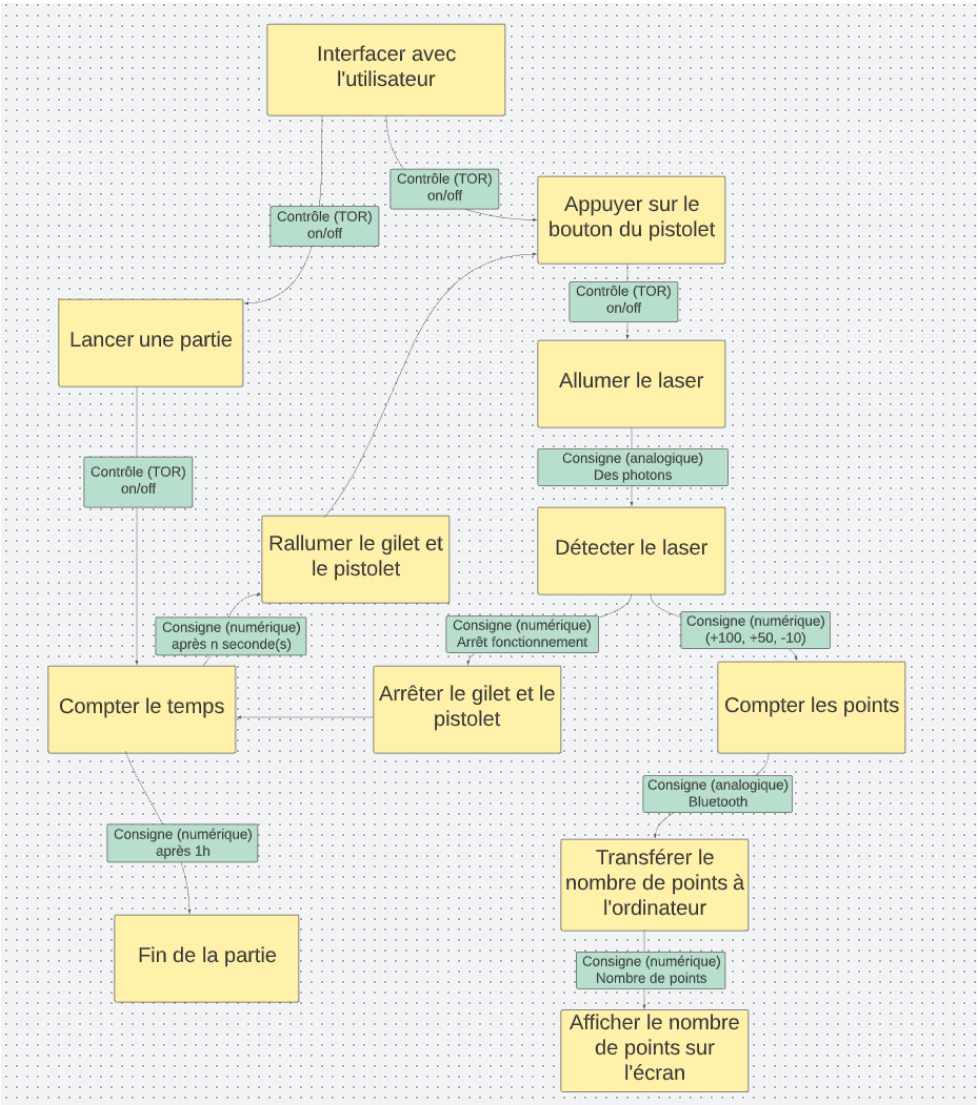


Figure 2 : Schéma fonctionnel du système

Les éléments du cahier de charges que nous avons fixé au début de la réalisation du projet sont les suivants :

- Le jeu doit pouvoir être utilisé par un enfant de 10 ans.
- Le laser doit être de classe 1.
- Le temps de réaction du système pour s'éteindre après une détection laser doit être inférieure à 3 sec (on verra si on peut faire moins).
- Le transfert du nombre de points vers l'ordinateur après une détection laser doit durer moins de 3 sec.
- Le laser doit être déclenché moins d'une seconde après appui sur le bouton et de la même façon doit s'éteindre moins d'une seconde après avoir relâché le bouton.
- Le gilet doit être facile à porter et l'utilisateur ne doit pas être gêné par l'électronique du système.
- Le système doit être autonome durant toute la durée du jeu soit 1h.
- Les gilets de chaque équipe doivent être distingués par une couleur particulière.
- Le gilet doit être résistant aux mouvements.
- Le gilet ne doit pas excéder 1kg.
- Le pistolet ne doit pas excéder 500g.
- L'esthétique est un point secondaire mais important.

1.2 Répartition des tâches

Pour réaliser notre projet, nous avons utilisé l'outil de gestion de projet Trello. Il nous a permis de centraliser les informations utiles et de noter notre avancement à la fin de chaque séance. En ce qui concerne la répartition des tâches, nous avons traité la partie concernant le système d'émission ensemble, par la suite Arthur s'est concentré sur la partie de l'électronique analogique du système de détection, Yohan sur la partie de la communication Bluetooth entre 2 cartes Nucléo et Wafaa sur le programme du traitement du signal détecté et le calcul des points de chaque équipe.

2 Réalisation du prototype

2.1 Système d'émission

En premier lieu, il a fallu connaître les caractéristiques optimales d'utilisation de nos composants. Ainsi, grâce aux méthodes acquises lors des premières séances de l'année en photo-émission/détection, nous avons pour la LED :

- Tension de fonctionnement : 9 V.
- Courant de fonctionnement : 25 mA.

La carte Nucléo permet une tension de sortie maximale de 5 V, ce qui nous a poussé à utiliser un transistor pour piloter la LED avec une pile de 9 V.

Ensuite, anticipons un peu, nous voulions supprimer la lumière de la salle à 50Hz dans la détection, ce qui nous a forcé à utiliser un filtre passe-haut du second ordre. Malheureusement, le filtrage n'était suffisant qu'à partir de quelques kHz. Nous avons donc mis en place un pilotage à partir de 5 kHz pour l'émission. La LED ne permettait pas une modulation convenable à 5 kHz, nous avons donc utilisé, pour les tests, un laser fourni par le laboratoire. Il ne nécessitait pas de transistor.

2.2 Système de détection

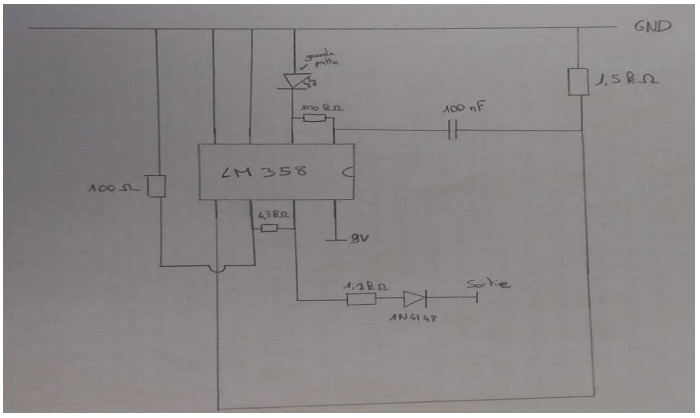
Tout d'abord, il a fallu choisir le bon montage de photodétection afin que la fréquence que l'on cherche à capter soit bien dans la bande passante du système et que le gain qui est associé à cette fréquence soit bon. C'est pour cela que nous avons choisi un montage transimpédance avec une résistance de 10Mohm.

Ensuite, nous avons voulu couper la fréquence de la lumière environnante (100Hz), nous avons donc mis un passe haut du premier ordre ayant une fréquence de coupure à 1000Hz afin de ne pas impacter la fréquence du signal et de bien couper la lumière parasite.

Une fois arrivé ici, nous nous sommes rendus compte que notre gain, que l'amplitude de notre signal était relativement faible. Nous avons donc décidé d'y ajouter un amplificateur multiplicateur (x47).

Pour cela, nous avions besoin d'un second AO, nous avons donc utiliser le LM358, un AO possédant 2 ali mais aussi pouvant se brancher à la masse côté négatif ce qui nous arrangeait bien sachant qu'on avait aucun signal négatif ni à traiter ni à amplifier.

Enfin, il ne restait plus qu'à s'assurer que le signal entrant dans la carte nucléo soit bien compris entre 0V et 3,3V. Pour cela nous avons utilisé une diode 1N4548 alimentée en 3,3V.

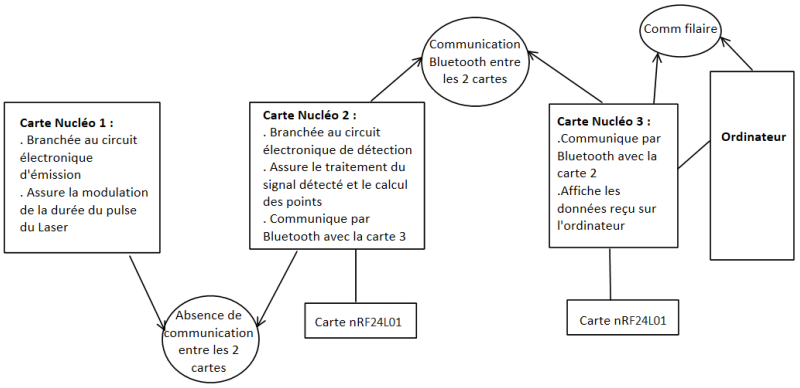


2.3 Système de traitement des données détectées

Après la détection du signal laser, la photodiode envoie un signal à la carte Nucleo à travers l'une de ses entrées analogiques. Le but maintenant est de tirer une information de ce signal qui nous permettra de différencier entre le signal laser émis par chacune des équipes.
Le signal laser émis, et donc le signal détecté, a une forme carré de période T et de rapport cyclique r. Mathématiquement ce signal est modélisé par une fonction porte de largeur T.r convolué à une peigne de dirac de période T: $s(t) = \Pi_{Tr} * \text{Ш}_T$. La transformée de fourier de ce signal nous donne accès au rapport cyclique r :

$$max(TF(s)) = max(rT\ sinc(rT\nu)\ \text{Ш}_{\frac{1}{T}}) = rT$$

Le rapport cyclique est une information qui nous intéresse, car c'est elle qui nous permettra de différencier entre les 2 équipes (chaque équipe a son propre rapport cyclique).
Le programme que nous avons conçu se base sur cette donnée. En effet, le programme embarqué dans la carte Nucleo récupère le signal détecté par la photodiode à partir de l'une de ses entrées analogiques : il fait un échantillonnage du signal en respectant la condition de Shannon et stocke les différents échantillons dans un tableau. Ensuite, il calcule la transformation de Fourier du signal et stocke ses valeurs dans un nouveau tableau. Enfin, il récupère la valeur maximale de cette transformée de Fourier ainsi que son indice dans le tableau.
C'est cette dernière donnée (l'indice du max de la TF) qui nous permettra de différencier entre les 2 équipes plus efficacement. En effet, lorsque nous avons fait des tests du programme, nous avons constaté que la valeur maximale de la transformée de fourier du signal détecté varie beaucoup après chaque acquisition, en revanche son indice dans le tableau reste presque invariable (avec un défaut de $\pm$ 1). Nous avons donc décidé de baser le calcul des points sur cette donnée.
Après la récupération de l'indice de la valeur maximale de la transformée de Fourier du signal détectée, le programme fait une comparaison entre cet indice et les 2 indices attendus (un indice est attribué à chaque équipe selon le rapport cyclique du signal laser qu'elle émis). Ensuite, il attribue à chaque équipe le nombre de points qui lui correspond en respectant les règles du jeu.
Pour communiquer sans fil ce nombre de points avec l'ordinateur, nous avons utilisé des cartes nRF24L01 qui assurent une communication par Bluetooth entre 2 cartes Nucléo. Les cartes nRF24L01 se branchent aux cartes Nucléo, il fallait donc brancher en permanence une autre carte Nucleo avec l'ordinateur. Nous avons utilisé le logiciel Teraterm pour visualiser sur l'ordinateur les données reçues par cette carte. Le schéma suivant représente le système global final :



Le programme complet est en annexe.

3 Système final

3.1 Tests de validation

Nous avons dans un premier temps évaluer chacun des sous-systèmes indépendamment avec des sources de tensions bien connues avec le GBF et regarder les résultats avec l'oscilloscope.
Tout d'abord, à propos du système d'émission, le programme a immédiatement marché. Cependant la fréquence de modulation du laser n'était pas de 5kHz mais plutôt de 4,4kHz. Ce qui n'était pas gênant ni d'un point de vu bande passante ni d'un point de vu gain du système de réception. Ce sous-système a donc très rapidement été validé.
Passons à présent au système de réception. Chaque élément détaillé dans le paragraphe 2.2) a été ajouté suite à ces fameux tests de validations qui ont eu lieu entre chaque nouveau composant. On a donc eu un gain final très bon et une fréquence de réception semblable à celle d'émission. De plus, cette fréquence n'a été modifiée durant tous les traitements qui ont été appliqués et finalement, le signal n'est que très peu bruité ce qui est satisfaisant.
Le dernier sous-système fut le plus délicat à mettre en œuvre. Dans un premier temps, on a testé le simple fait d'envoyer une donnée d'une carte à l'autre par bluetooth, ce qui a été un succès. Puis, indépendamment du bluetooth, on a cherché à analyser le signal reçu. Pour cela, on a cherché quelles informations étaient révélatrices de la fréquence. Une fois ceci réalisé, on a essayé le système complet avec seulement une cible. Puis une seconde ne rapportant pas le même nombre de points et malgré un très long temps de réaction notre système complet semblait marcher. Malheureusement, nous avons manqué de temps pour améliorer ce temps de réaction et bien vérifier si le système final était bien conforme au cahier des charges.

3.2 Améliorations

La première amélioration que l'on peut apporter est d'insérer les deux systèmes (émission / détection) sur une seule carte Nucléo. Aussi, nous avons remarqué que le temps d'affichage des points était relativement long (supérieur à 5 s), il peut être diminué et être quasi-instantané. A terme, on peut espérer jouer avec plus de deux équipes, voir en individuel seul contre tous.

4 Annexe

File "/point_bluetooth_prof/main.cpp" printed from os.mbed.com on 24/05/2022

```
//programme a mettre sur les cartes de l'equipe 1, pour l'equipe 2 le calcul sur le tire sur un coequipier doit changer

/* DECLARATION DES BIBLIOTHEQUES */
#include "mbed.h"
#include "arm_math.h"
#include "dsp.h"                            // Bibliotheques contenant les fonctions necessaires au calcul de la TF du signal
#include "arm_common_tables.h"
#include "arm_const_structs.h"
#include "nRF24L01P.h"                      // Bibliotheque contenant les fonctions du modudle nRF24 (communication Bluetooth)

/* DECLARATION DES MACROS*/
#define TRANSFER_SIZE         8
#define SAMPLES             512             // 256 real party and 256 imaginary parts
#define FFT_SIZE            SAMPLES / 2     // FFT size is always the same size as we have samples, so 256 in our case
#define index_equipe1       73              // L'indice(dans le tableau output)du max de la fft du signal laser de l'équipe 1
#define index_equipe2       64
#define epsilon             2

/* DECLARATION DES VARIABLES GLOBALES */
bool        trig=0;
int         indice = 0;
int         nbr_pts_equipe1=100;
int         nbr_pts_equipe2=100;
char        k;
char        dataToSend[TRANSFER_SIZE] = {0xAA, 0x01, 0x10, 0xF0,0xAA, 0x01, 0x10, 0xF0};
char        dataReceived[TRANSFER_SIZE] = {0};
char        rxDataCnt;

float32_t   Input_ventre[SAMPLES];          // Tableau qui va contenir les échantillons du signal
float32_t   Input_epaule_droit[SAMPLES];
float32_t   Input_epaule_gauche[SAMPLES];

float32_t   Output_ventre[FFT_SIZE];        // Tableau qui va contenir les valeurs de la Tf du signal
float32_t   Output_epaule_droit[FFT_SIZE];
float32_t   Output_epaule_gauche[FFT_SIZE];

/* DECLARATION DES ENTREES/SORTIES */
DigitalOut  myled(D13);
AnalogIn    myADC(A0);
Serial      debug_pc(USBTX, USBRX);
Ticker      timer;
DigitalIn   bouton(D7);
nRF24L01P   nRF24_mod(D11, D12, D13, D10, D9, PB_8); // MOSI, MISO, SCK, CSN, CE, IRQ (les entrées de la carte nRF24)
AnalogIn    cap_ventre(A0);
AnalogIn    cap_epaule_droit(A1);
AnalogIn    cap_epaule_gauche(A2);


/* FONCTION D'INITIALISATION DU MODULE BT nRF24L01 */
void initNRF24()
{
    nRF24_mod.powerUp();
    wait_us(100000);
    nRF24_mod.setAirDataRate(NRF24L01P_DATARATE_250_KBPS);
    nRF24_mod.setRfFrequency(2400);
    wait_us(100000);
    debug_pc.printf( "nRF24L01+ Frequency    : %d MHz\r\n",  nRF24_mod.getRfFrequency() );
    debug_pc.printf( "nRF24L01+ Output power : %d dBm\r\n",  nRF24_mod.getRfOutputPower() );
    debug_pc.printf( "nRF24L01+ Data Rate    : %d kbps\r\n", nRF24_mod.getAirDataRate() );
    debug_pc.printf( "Transfers are grouped into %d characters\r\n", TRANSFER_SIZE );
    nRF24_mod.setTransferSize( TRANSFER_SIZE );
    nRF24_mod.setReceiveMode();
    nRF24_mod.enable();
}

/* FONCTION DE TEST DU MODULE BT nRF24L01 */
void testNRF24(void)
{

    // Lecture donnée depuis nRF24
    if ( nRF24_mod.readable() ) {
        debug_pc.printf(" 1");
        rxDataCnt = nRF24_mod.read( NRF24L01P_PIPE_P0, dataReceived, TRANSFER_SIZE);
        debug_pc.printf(" 2");
        debug_pc.printf("\tD = ");
        for ( int i = 0; i < rxDataCnt; i++ ) {
            debug_pc.printf(" %d \t", dataReceived[i]);
            debug_pc.printf(" Recu");
        }
        debug_pc.printf("\r\n");
    }

    // Transmission donnée depuis nRF24
    if(dataReceived[0] == 0) {
        nRF24_mod.setRfFrequency(2400);
        nRF24_mod.write( NRF24L01P_PIPE_P0, dataToSend, TRANSFER_SIZE );
        debug_pc.printf( "Nombre de point equipe 1: %d \r \n Nombre de point equipe 2: %d \r \n", 10 * (dataToSend[1] - 100), 10 * (dataToSend[2] - 100));
        debug_pc.printf( "SENDED\r\n");
    }
}

/* FONCTION D'ECHANTILLONAGE */
void sample()
{
    myled = 1;

    if(indice < SAMPLES) {
        //Real part NB removing DC offset
        Input_ventre[indice] = cap_ventre.read() - 0.5f;
        Input_epaule_droit[indice] = cap_epaule_droit.read() - 0.5f;
```

```
        Input_epaule_gauche[indice] = cap_epaule_gauche.read() - 0.5f;

        //Imaginary Part set to zero
        Input_ventre[indice + 1] = 0;
        Input_epaule_droit[indice + 1] = 0;
        Input_epaule_gauche[indice + 1] = 0;

        indice += 2;
    } else {
        trig = 0;
    }

    myled = 0;
}

/* FONCTION PRINCIPALE */
int main()
{
    float maxValue_ventre;            // Max FFT value is stored here
    float maxValue_epaule_droit;
    float maxValue_epaule_gauche;

    uint32_t maxIndex_ventre=0;       // Index in Output array where max value is
    uint32_t maxIndex_epaule_droit=0;
    uint32_t maxIndex_epaule_gauche=0;

    initNRF24();

    while(1) {
        if(trig == 0) {

//calcul de la TF du signal detecte + recuperation de l'indice de son max
            timer.detach();

            // Fonction predefinie
            arm_cfft_f32(&arm_cfft_sR_f32_len256, Input_ventre, 0, 1);
            arm_cfft_f32(&arm_cfft_sR_f32_len256, Input_epaule_droit, 0, 1);
            arm_cfft_f32(&arm_cfft_sR_f32_len256, Input_epaule_gauche, 0, 1);

            // Fonction prédéfinie qui permet de calculer la TF du signal et de la stocker dans le tableau Output
            arm_cmplx_mag_f32(Input_ventre, Output_ventre, FFT_SIZE);
            arm_cmplx_mag_f32(Input_epaule_droit, Output_epaule_droit, FFT_SIZE);
            arm_cmplx_mag_f32(Input_epaule_gauche, Output_epaule_gauche, FFT_SIZE);

            Output_ventre[0] = 0;
            Output_epaule_droit[0] = 0;
            Output_epaule_gauche[0] = 0;

            //Fonction prédefinie qui permet de récuperer la valeur maxumale de la TF ainsi que son indice dans Output
            arm_max_f32(Output_ventre, FFT_SIZE/2, &maxValue_ventre, &maxIndex_ventre);
            arm_max_f32(Output_epaule_droit, FFT_SIZE/2, &maxValue_epaule_droit, &maxIndex_epaule_droit);
            arm_max_f32(Output_epaule_gauche, FFT_SIZE/2, &maxValue_epaule_gauche, &maxIndex_epaule_gauche);

            debug_pc.printf(" indice %d",maxIndex_epaule_droit);


// Calcul des points

            //Tire sur un coequipier
            if ((maxIndex_ventre<(index_equipe1+epsilon)) && (maxIndex_ventre>(index_equipe1-epsilon))) {
                nbr_pts_equipe1=nbr_pts_equipe1-2; // Tire sur le ventre d'un coequipier: -2pts pour l'equipe du tireur
            } else {
                if (maxIndex_epaule_droit<(index_equipe1+epsilon) && maxIndex_epaule_droit>(index_equipe1-epsilon)) {
                    nbr_pts_equipe1=nbr_pts_equipe1-2; // Tire sur le épaule D d'un coequipier: -2pts pour l'equipe du tireur
                } else {
                    if (maxIndex_epaule_gauche<(index_equipe1+epsilon) && maxIndex_epaule_gauche>(index_equipe1-epsilon)) {
                        nbr_pts_equipe1=nbr_pts_equipe1-2; // Tire sur le épaule G d'un coequipier : -2pts pour l'equipe du tireur
                    }

                    //Tire sur le ventre
                    else {
                        if (maxIndex_ventre<(index_equipe2+epsilon) && maxIndex_ventre>(index_equipe2-epsilon)) {
                            nbr_pts_equipe1=nbr_pts_equipe1-1; //Tire sur le ventre : -1pts pour l'equipe de la victime
                            nbr_pts_equipe2=nbr_pts_equipe2+5; //Tire sur le ventre : +5pts pour l'equipe du tireur
                        }
                        //Tire sur l'epaule
                        else {
                            if (maxIndex_epaule_droit<(index_equipe2+epsilon)&& maxIndex_epaule_droit>(index_equipe2-epsilon)) {
                                nbr_pts_equipe1=nbr_pts_equipe1-1; //Tire sur l'epaule : -1pts pour l'equipe de la victime
                                nbr_pts_equipe2=nbr_pts_equipe2+10; //Tire sur l'epaule : +10pts pour l'equipe du tireur
                            } else {
                                if (maxIndex_epaule_gauche<(index_equipe2+epsilon)&& maxIndex_epaule_gauche>(index_equipe2-epsilon)) {
                                    nbr_pts_equipe1=nbr_pts_equipe1-1; //Tire sur l'epaule : -1pts pour l'equipe de la victime
                                    nbr_pts_equipe2=nbr_pts_equipe2+10; //Tire sur l'epaule : +10pts pour l'equipe du tireur
                                }
                            }
                        }
                    }
                }
            }

// Envoie du nombre de point au ordinateur par bluetooth
            dataToSend[1] = nbr_pts_equipe1;
            dataToSend[2] = nbr_pts_equipe2;
            dataToSend[3] = maxValue_ventre;
            testNRF24();
            wait(0.5);

// Relance de l'acquisition du signal
            trig = 1;
            indice = 0;
            timer.attach_us(&sample,50); //critere de shannon
        }
```