Université François-Rabelais de Tours
Institut Universitaire de Technologie de Tours
Département Génie Électrique et Informatique Industrielle

UNIVERSITE FRANCOIS-RABELAIS
TOURS
Institut Universitaire de Technologie
Département
GENIE ELECTRIQUE ET
INFORMATIQUE INDUSTRIELLE

Projets d'étude et réalisation
Détection du kart
(Bornes du 50m départ arrêté)

Jérôme BEGNEU
Matthieu JOUFFREY
2ème Année – P2
Promotion 2007/2009

Enseignants
Thierry LEQUEU
Jacky BRUN

Université François-Rabelais de Tours
Institut Universitaire de Technologie de Tours
Département Génie Électrique et Informatique Industrielle

UNIVERSITE FRANCOIS-RABELAIS
TOURS
Institut Universitaire de Technologie
Département
GENIE ELECTRIQUE ET
INFORMATIQUE INDUSTRIELLE

Projets d'étude et réalisation
Détection du kart
(Bornes du 50m départ arrêté)

Jérôme BEGNEU
Matthieu JOUFFREY
2ème Année – P2
Promotion 2007/2009

Enseignants
Thierry LEQUEU
Jacky BRUN

Sommaire

Introduction :

Le projet que nous avons réalisé, sert à détecter la présence d'un kart, spécialement conçu pour une épreuve appelée le 50m départ-arrêté. Le but de cet épreuve est de réaliser le temps le plus faible entre la ligne de départ et la ligne d'arrivée. Le départ se fait comme le nom de l'épreuve l'indique, arrêté ; sans aucuns obstacles, comme des cônes ou des balises seront placés sur la piste, qui sera une ligne droite d'une distance de 50m.

Le projet sera réalisé avec deux bornes (une au départ et une autre placée à l'arrivée) qui auront des équipements divers. Pour nous aider à réaliser notre projet, nous utiliserons : les bornes, les feux de départ, l'afficheur à led et la carte de contrôle du microcontrôleur (µC) effectués durant les années précédentes, lors d'ancien projets.

Illustration 1: Borne affichage et feux de départ

1. Cahier des charges :

Comme d'écrit dans l'introduction, il y aura une borne de départ puis une borne d'arrivé. Chaque borne sera dotée de deux détecteurs.

La borne de départ, doit pouvoir détecter la présence du kart, pour permettre le lancement du feu de « START ». La détection devra se faire soit dès l'arrivée du kart à la ligne de départ soit l'or de l'attente du « START ». Mais aussi de détecter un éventuelle faux départ et dans ce cas le signaler par l'action d'un feu orange clignotant. Le feu devra reprendre au rouge si le faux départ est corrigé.

A l'éclairage du feu vert (signal de « START ») un chronomètre devra démarrer et être affiché sur un écran.

La borne d'arrivée, doit pouvoir arrêter le chronomètre et calculer la vitesse. Le premier détecteur va mémoriser le temps du passage à son niveau. Le deuxième détecteur arrête et enregistre le temps définitif du chronomètre.

La vitesse et le temps seront affichés successivement sur l'écran à led réalisé lors des années précédentes.

Le projet doit fonctionner dans sa totalité mais aussi répondre à quelques contraintes :

DETECTEURS

◆ Les détecteurs ne devront pas avoir de connexion via un fil.

◆ Les détecteurs utilisés pourront être de n'importe quelle espèce (tout en respectant la première condition) mais doivent pouvoir détecter et commuter dans les plus brefs délais pour une précision de calcul et de mesure optimale.

◆ Les détecteurs devront avoir une portée d'au moins un mètre pour permettre le passage facile d'un kart.

◆ Le signal de sortie du capteur (présence ou non d'un kart), devra être un signal TTL (0/5V), pour permettre le traitement des données sur le µC.

◆ Les deux détecteurs seront espacés de 30 cm.

ALIMENTATION ET CONTRAINTE EXTÉRIEURE

◆ L'alimentation sera effectuée par une batterie OPTIMA 12 V 48HA JAUNE.

◆ Tous les composants utilisés devront supporter des contraintes environnementales, donc pouvoir fonctionner à des températures de 0°C à 50°C, supporter la poussière, l'humidité et les variations de lumière.

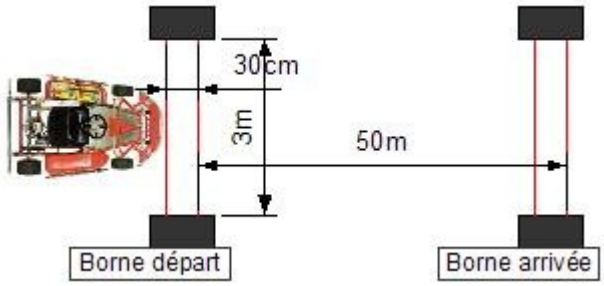


Illustration 2: Schéma du projet

2. Choix du capteur :

Dans un premier temps nous avons étudié les projets effectués lors des années précédentes. Leurs choix de capteurs étaient laser, un pointeur laser comme émetteur et un phototransistor comme récepteur. Bien qu'une lentille fût utilisée pour focaliser le rayon laser sur le récepteur, le réglage était assez minutieux donc compliqué.

Nous avons donc essayé des capteurs infrarouges, qui servent à la détection de mouvements. Après étude : les capteurs avaient un bon temps de réaction, mais ils avaient deux problèmes majeurs : le premier était qu'ils détectaient les mouvements et que le kart était à l'arrêt durant l'attente du « START », mais surtout que l'angle d'émission était trop large, ce qui aurait perturbé les récepteurs, ce qui signifie que le récepteur 1 sera parasité par l'émetteur 2 et inversement.

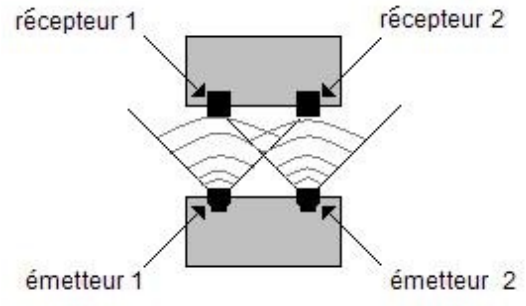


Illustration 3: Émission et réception par Infra Rouge

Nous avons donc essayé de focaliser l'émission mais les ondes se sont propagées dans un angle encore trop élevé. Nous avons donc décidé de prendre un autre capteur qui sera la solution prise par les groupes des années précédentes et que nous retiendrons : un capteur laser.

L'émission se fait directement par un faisceau (pointeurs lasers modifiés lors des années précédentes), donc pas de perturbation sur l'autre détecteur, par l'emmetteur. Le temps de commutation dépend du récepteur, nous avons opté pour une photodiode (BPW34) qui à un bon temps de réaction. Nous avons conçu une carte qui est alimentée en 5V pour permettre le traitement des signaux par le µC (un signal TTL).

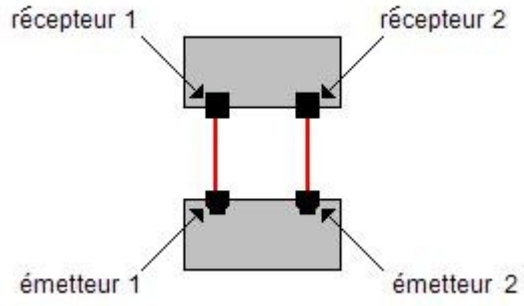


Illustration 4: Émission et réception par laser

3. Capteur laser :

3.1. Schéma du récepteur :

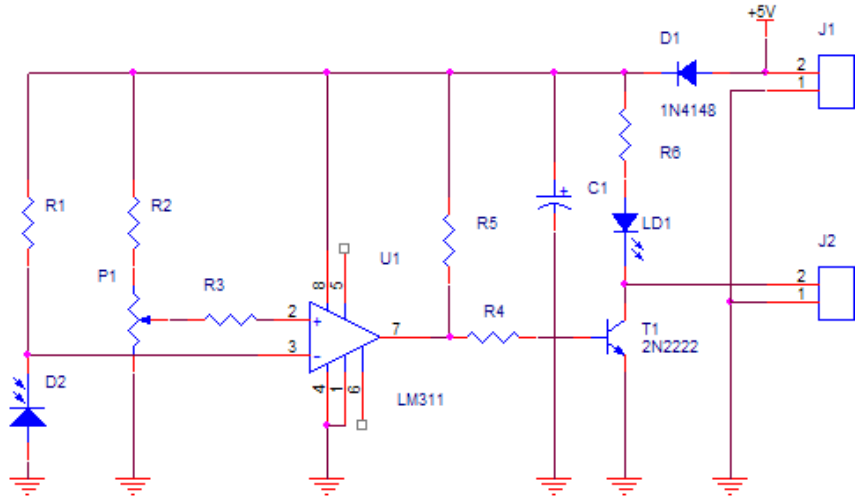


Illustration 5: Schéma du montage

3.2. Description des composants :

J1 et **J2** sont des borniers qui vont permettre, pour **J1** de fournir au montage la tension de 5 Volts. Et pour **J2** de récupérer le signal TTL à envoyer au µC.[1]

La diode **D1** est une diode qui sert à protéger le circuit en cas de mauvais branchements de la tension d'entrée.

Le condensateur **C4** est un condensateur électrochimique qui permet de lisser la tension du circuit.

Le **U1** est un Aop (LM311) utilisé en comparateur, il sert à donner 0 ou 5 volts suivant la tension présente sur ces bornes d'entrées. Si V+>V- alors la sortie de l'Aop (Vs) sera égale à 5 Volts. Si V+<V- alors Vs=0 Volts. En réalité la tension sera inférieure à 5 volts dûe à la chute de tension créée par la diode **D1**.

Une tension de seuil réglable est créée sur l'entrée positive de l'Aop, le réglage du seuil est effectué par le potentiomètre **P1** et sert à régler le temps de réaction de détection pour permettre une grande précision.

Lorsqu'il n'y a pas de faisceaux laser sur la photodiode (BPW34) **D2**, la résistance **R1** crée une tension fixe supérieur a la tension de seuil, Vs et donc égale a 0. Si un faisceau laser est sur **D2** alors une chute de tension va se créer, la tension sur l'entrée négative sera alors inférieure à celle de la tension de l'entrée positive. Vs sera donc égale à 5V.

La résistance **R5** est une résistance de pull-up qui sert au bon fonctionnement du LM311.

Le transistor **T1,** sert d'interrupteur qui allumera la diode électroluminescence **LD1**, si un faisceau est détecté sur la photodiode **D2**, le transistor se sature. Dans le cas contraire **T1** se bloc et **LD1** s'éteint.

3.3. Tests de temps de réaction :

Pour réaliser les tests de temps de réaction de notre détecteur, nous avons utilisé un système avec moteur électrique possédant une hélice ; réalisé par nos collègues du groupe P1, qui l'ont baptisé « la machine à découper les faisceau »

Illustration 6: Machine à découper les faisceau

Nous avons donc prit le système qui sert à couper le faisceau laser à une vitesse constante qui sert à juger de l'efficacité du temps de réaction. Nous rappelons qu'une mesure de vitesse est faite en fin de course sur 30 cm et le capteur doit être capable de détecter la présence du kart à une vitesse élevée. À 100 km/h le kart fait 2 778 cm/s sur une distance de 30 cm cela représente 10,79ms.

Les tests ont été effectués avec un récepteur alimenté en 0-12V, nous n'avons donc pas de signal TTL en sortie du récepteur. Le moteur était à sa vitesse maximale.

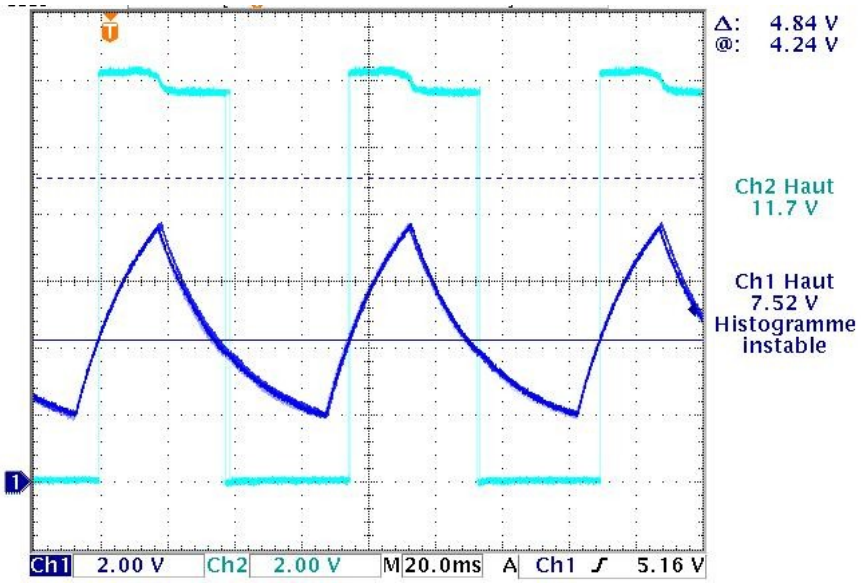


Illustration 7: Oscillogramme du temps de réaction du récepteur

Sur cet oscillogramme nous pouvons voir apparaître en bleu foncé, sur CH1 la tension sur l'entrée inverseuse de l'Aop. Nous avons la tension de la photodiode qui agit sur cette entrée. La phase descendante correspond à une détection du faisceau laser. La phase ascendante correspond à une absence de ce faisceau laser.

La voie CH2 correspond au signal récupéré en sortie du comparateur. C'est elle qui va nous donner le temps de réaction du récepteur. Ici, nous pouvons voir que à la vitesse maximal du moteur, la photodiode détecte bien le laser et que le signal a récupéré est un signal créneau qui correspond à la tension comparée entre l'entrée positive et négative.

3.4. Problèmes rencontrés :

Lors de l'étude du capteur laser nous avons été confronté à quelques problèmes mineurs, notamment avec l'utilisation du comparateur et la mise en forme du typon final. De nombreuses séances auront servi au dépannage de la carte.

Nous aurions voulu étudier plus en détails, pour la facilité de mise en place du système et ainsi gagner du temps lors de la mise en place de celui-ci, par le biais d'une lentille convergente vers la photodiode.

Nous avons aussi manqué de temps pour réaliser plusieurs détecteurs, au nombre de quatre, et pour les placer dans des boîtiers propres à l'abri de la poussières, de l'humidité et de tout autre élément extérieur.

4. Composants :

Voici la liste des composants utilisés pour la réalisation d'un récepteur laser, tous les composants auront été fournis par le magasin de l'IUT. Tous les prix des composant auront été pris sur radiospares et sont calculés comme acheté à l'unité.

Nombres	Nom du composant	Symbole	Valeur	Nombres	Source	Prix unitaire	Total
Résistance		R1,R3,R5	10kΩ	3	Magasin IUT	1,10 €	3,30 €
		R2,R4	22kΩ	2	Magasin IUT	1,10 €	2,20 €
		R6	220Ω	1	Magasin IUT	1,10 €	1,10 €
Potentiomètre		P1	100kΩ	1	Magasin IUT	0,82 €	0,82 €
Condensateur électrochimique		C1	100µF/25V	1	Magasin IUT	0,12 €	0,12 €
Connecteur 2 broches		J1		1	Magasin IUT	1,42 €	1,42 €
Bornier		J2		1	Magasin IUT	0,58 €	0,58 €
LED Rouge		LD1		1	Magasin IUT	0,43 €	0,43 €
Support 8 broches				1	Magasin IUT	0,45 €	0,45 €
Diode de redressement	1N4148	D1		1	Magasin IUT	0,02 €	0,02 €
Photodiode laser	BPW34	D2		1	Magasin IUT	1,51 €	1,51 €
Transistor bipolaire NPN	2N2222	T1		1	Magasin IUT	0,51 €	0,51 €
Amplificateur Opérationnel	LM311	U1		1	Magasin IUT	0,41 €	0,41 €
TOTAL				16			12,87 €

Le coût total de la réalisation du projet est d'environ 12,87€, sans compter que plus on commande de composants, moins le coût du composant à l'unité est élevé. Mais nous n'avons pas évalué le coût de la plaque sérigraphiée.

5. La programmation :

5.1. Généralité :

Le programme qui a été réalisé, nous permettra de contrôler l'allumage du feux tricolore, l'affichage du chronomètre ainsi que le calcul de la vitesse du kart. Ces différentes fonctions seront activées selon la détection du kart par les quatre capteurs. Deux capteurs seront sur la borne de départ et deux autres sur la borne d'arrivée.

5.1.1. La borne de départ :

Quand le kart ne sera pas détecté par les deux capteurs, le feux sera orange clignotant, signalant ainsi que le kart peut se préparer. Une fois que le kart sera détecté par le premier capteur, le feux passera au rouge, puis à l'orange et pour finir au vert ; le chronomètre s'enclenchera et le kart pourra alors commencer sa course. Le deuxième capteur aura pour fonction de détecter un faux départ, c'est-à-dire que si le kart est détecté par ce capteur, le feux passera immédiatement au orange clignotant ; le kart devra alors revenir se placer de façon à ce que seulement le premier capteur soit détecté. Le feux tricolore s'activera jusqu'à repasser au vert et réenclencher le chronomètre.

Si le kart part au feux vert sans toucher le deuxième capteur avant, il pourra faire sa course jusqu'à la borne d'arrivée 50 mètres plus loin.

5.1.2. La borne d'arrivée :

Le premier capteur de cette borne aura pour fonction de retenir la valeur du chronomètre(T1). Le deuxième capteur, arrêtera le chronomètre indiquant la fin de la course, puis retiendra la nouvelle valeur (T2). Une fois réalisé, le programme va calculer la vitesse du kart quand celui ci aura passé le dernier capteur par la formule suivante :

Vitesse = distance/temps = 0,0003/((T2 – T1)/3600)

La distance entre les deux capteurs sera toujours égale à 30cm. Nous mettons cette valeur en kilomètres ce qui fait 0,0003km.

Le temps sera la soustraction : du temps de passage du kart retenu par le deuxième capteur moins celui retenue par le premier capteur. La valeur obtenue est en seconde, c'est pour cela que nous la diviserons par 3600 pour avoir des heures.

La vitesse est égale à la division de la distance par le temps ; nous obtiendrons donc un résultat en km/h qui pourra être affiché successivement sur l'afficheur à led avec le temps effectué lors de la course.

5.2. Le programme :

Nous programmons le projet avec le logiciel CodeVisionAVR par l'intermédiaire d'une carte Atmega8535 et d'un afficheur LCD. Cette carte sera reliée par un cordon de programmation avec la carte du connecteur ISP norme TL, qui elle sera branchée à l'ordinateur.

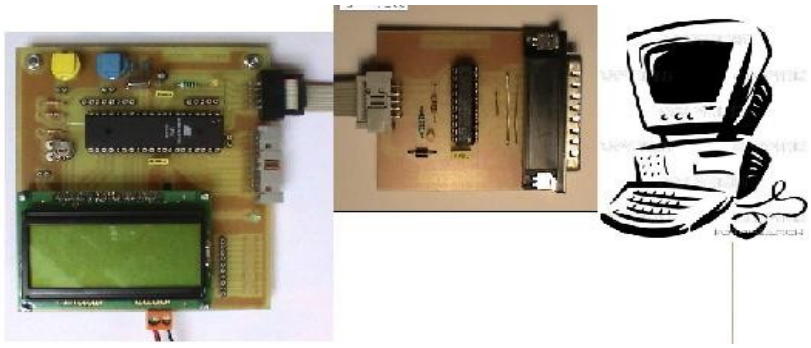


Illustration 8: Schéma du câblage pour la programmation

5.2.1. Les fonctions « feux rouge », « feux orange » :

Ces deux fonctions vont tout simplement simuler le feux rouge et le feux orange. La simulation se fera par l'affichage de « feux rouge » et « feux orange » sur l'afficheur LCD. Le « feux vert » étant légèrement différent il apparaîtra ultérieurement.

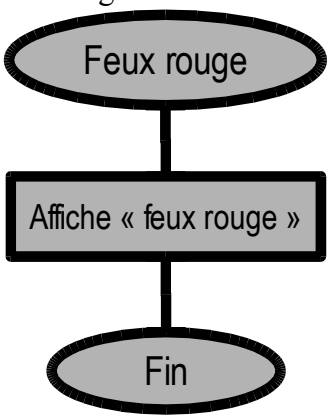


Illustration 9: Ordinogramme de "feux rouge"

```
void Feux_R(void)     //Fonction feux rouge
{
 lcd_gotoxy(0,1);              //Place le texte sur l'afficheur
 lcd_putsf("feux rouge");    //Permet d'écrire «feux rouge»

 lcd_gotoxy(0,2);          //Place le texte sur l'afficheur
 lcd_putsf("          ");   //Permet d'effacer l'écriture «feux orange»
 lcd_gotoxy(0,3);          //Place le texte sur l'afficheur
 lcd_putsf("          "); //Permet d'effacer l'écriture «feux vert»
}
```

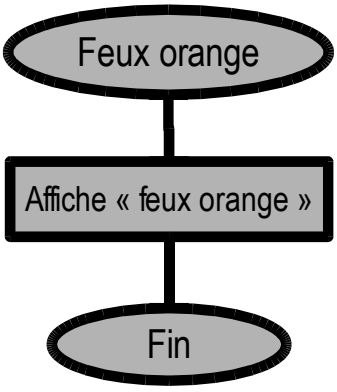


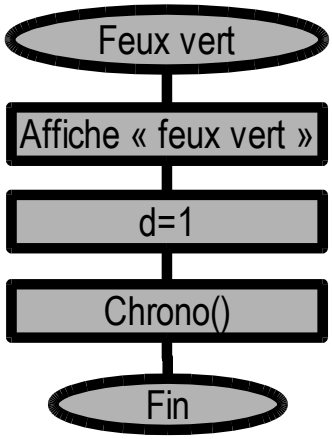
Illustration 10: Ordinogramme de "feux orange"

```
void Feux_O(void)          //Fonction feux orange
{
 lcd_gotoxy(0,2);               //Place le texte sur l'afficheur
 lcd_putsf("feux orange");    //Permet d'écrire «feux orange»

 lcd_gotoxy(0,1);            //Place le texte sur l'afficheur
 lcd_putsf("          ");      // Permet d'effacer l'écriture «feux rouge»
}
```

La fonction lcd_gotoxy(); nous permettra de placer où bon nous semble le texte voulu. Le premier chiffre dans les parenthéses sera celui des abscisses représenté par les x, et le deuxième chiffre des parenthéses sera celui des ordonnées représenté par les y. La fonction lcd_putsf(); nous permettra de saisir un texte. Pour n'avoir que l'écriture souhaité, nous effaçons les parties de texte qui reste après l'affichage. Cela sera réalisable par l'écriture d'un espace blanc à la place du texte.

5.2.2. La fonction « feux vert » :

Cette fonction permettra de simuler le feux vert, tout simplement par l'affichage de « feux vert » sur l'écran LCD. La fin de la fonction, une fois le feux vert affiché, enclenchera le chronomètre pour permettre au kart de commencer sa course.



Dessin 1: Ordinogramme de "feux vert"

La fonction affichera donc « feux vert » afin de simuler celui-ci. Puis va mettre la variable d à 1, qui nous servira dans la fonction « chronomètre ». Pour finir la fonction lancera un chronomètre.

```
void Feux_V(void)          //Fonction feux vert
 {
   lcd_gotoxy(0,1);          //Place le texte sur l'afficheur
   lcd_putsf("          ");  // Permet d'effacer l'écriture «feux rouge»
   lcd_gotoxy(0,2);          //Place le texte sur l'afficheur
   lcd_putsf("          ");  //Permet d'effacer l'écriture «feux orange»
   lcd_gotoxy(0,3);          //Place le texte sur l'afficheur
   lcd_putsf("feux vert");   //Permet d'écrire «feux vert»
   d=1;                      //On met la variable d à 1
   Chrono();                 //On appelle la fonction Chrono
 }
```

Nous ne noterons aucune nouveauté au niveau de la programmation par rapport aux autres fonctions. La seule différence de cette fonction est la présence de la variable d, qui sera mise à 1.

5.2.3. La fonction « feux orange clignotant »:

Cette fonction simulera le clignotement du feux orange. La simulation se fera par la répétition de l'affichage de « feux orange » et par l'effacement de l'écriture, ce qui nous donnera le clignotement. On aura mis y=0 avant le début de la fonction. [2]

La fonction « void feux orange » clignotant (void) » est principalement composée par un « if ». Au départ "*y*" est a 0, donc l'écriture « feux orange » apparaîtra, puis *"y"* passera à 1. De retour sur le « if », *"y"* ne sera plus à 0 donc effacera l'écriture, puis mettra *"y"* de nouveau a 0. Cette fonction réalisée en boucle permettra de simuler un clignotement de l'écriture « feux orange ».

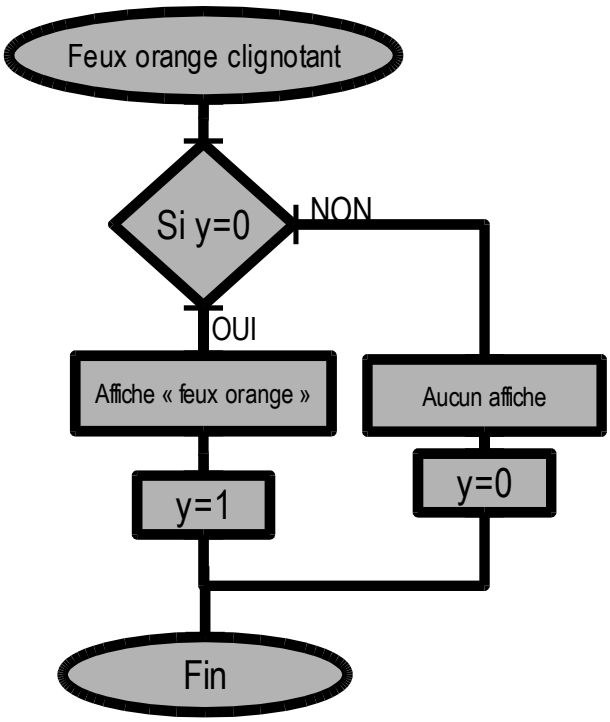


Illustration 11: Ordinogramme de "feux orange clignotant"

Les fonctions lcd_putsf() et lcd_gotoxy() ont les mêmes propriétés que pour les feux rouge et orange. La présence de la fonction delay_ms() va permettre de laisser pendant un temps défini(ici 1 seconde) l'écriture « feux orange » et par la suite de l'effacer toujours durant ce même un temps. Ici delay_ms(1000) représente un délai d'une seconde.

```
y=0;                                    //On met la variable y à 0
void Feux_Orange_Clignotant (void)      //Fonction Feux orange clignotant
{
      if(y==0)        //Quand la variable y est à 0
       {
       lcd_gotoxy(0,2);                 //Place le texte sur l'afficheur
       lcd_putsf("feux orange");        //Permet d'écrire «feux orange»

       lcd_gotoxy(0,1);                 //Place le texte sur l'afficheur
       lcd_putsf("          ");         //Permet d'effacer l'écriture «feux rouge»
       lcd_gotoxy(0,3);                 //Place le texte sur l'afficheur
       lcd_putsf("          ");         //Permet d'effacer l'écriture «feux vert»

       delay_ms(1000);                  //Laisse afficher pendant un court temps «feux orange»
       y=1;                             //Met la variable y à 1
       }

      else                              //Quand la variable y est à 1
       {
       lcd_gotoxy(0,1);                 //Place le texte sur l'afficheur
       lcd_putsf("          ");         //Permet d'effacer l'écriture «feux rouge»
       lcd_gotoxy(0,2);                 //Place le texte sur l'afficheur
       lcd_putsf("          ");         //Permet d'effacer l'écriture «feux orange»
       lcd_gotoxy(0,3);                 //Place le texte sur l'afficheur
       lcd_putsf("          ");         //Permet d'effacer l'écriture «feux vert»

       delay_ms(1000);                  //Efface les écritures pendant un court temps
       y=0;                             //Met la variable y à 0
       }
}
```

5.2.4. La fonction chronomètre :

La fonction « chronomètre » permettra le fonctionnement d'un chronomètre qui se déclenchera au moment où le feux passera au vert. Le chronomètre s'arrêtera lorsque le kart passera devant le deuxième détecteur de la borne d'arrivée, ce qui signifie la fin de la course. Le chronomètre sera aussi arrêté sur l'écran LCD.

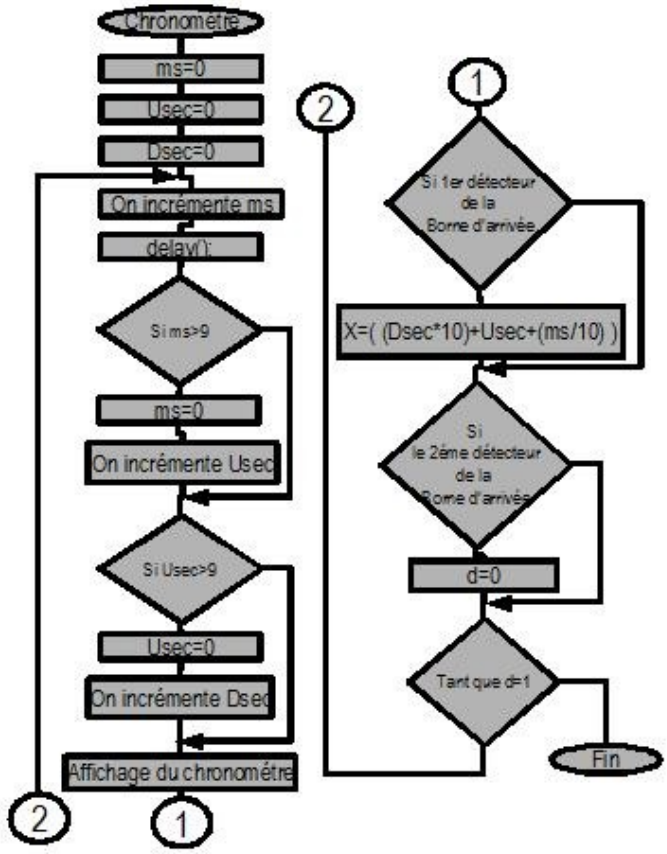


Illustration 12: Ordinogramme de "chronomètre"

Les variables « ms », « Usec » et « Dsec » sont misent à 0, afin que notre compteur commence à 0. « ms » représente les milli-secondes, « Usec » représente les unités de secondes et « Dsec » représente les dizaines de secondes. Ensuite nous rentrons dans une boucle « do,while », c'est-à-dire que le chronomètre va tourner jusqu'à ce que la variable « d » soit à 1. Dans la boucle « do, while » il y aura l'incrémentation, de la variable « ms », qui incrémentera toutes les millisecondes grâce à la fonction « delay_ms(10); ». Une fois que « ms » sera strictement supérieur à 9, elle sera remise à 0, à ce moment c'est la variable « Usec » qui s'incrémentera de 1. Comme les actions se trouvent dans la boucle « do, while », les millisecondes s'incrémenteront de nouveau jusqu'à 9 pour faire incrémenter « Usec » encore de 1, et ainsi de suite, jusqu'à avoir « Usec » strictement supérieur à 9. Quand « Usec » sera à 9, c'est la variable « Dsec » qui s'incrémentera à 1 et « Usec » sera remis à 0. Ainsi le chronomètre sera réalisé. Après la réalisation du chronomètre, nous l'afficherons.

Quand le kart passe devant le premier détecteur de la borne d'arrivée, le programme relèvera la valeur du chronomètre au moment du passage, par la formule suivante :

X = ((Dsec*10)+Usec+(ms/10))

Exemple:

Si le chronomètre indique 28,5 (28 secondes et 5 mili-secondes), donc Dsec=2, Usec=8 et ms=5.

Pour trouver le chiffre des secondes : on prend Dsec=2 et on fait (2*10)=20, puis on lui additionne Usec=8, ce qui nous donne 28. Nous obtenons donc le chiffre des secondes.

Pour trouver les milli-secondes : on prend ms=5, puis on le divise par 10, ce qui donne 0,5 s.

Le résultat final du chronomètre sera l'addition des secondes et des milisecondes : 28+0,5=28,5 s.

Le chronomètre s'arrêtera quand le kart passera devant le deuxième détecteur de la borne d'arrivée, puisque la variable « d » sera mise à 0, se qui nous permettra de sortir de la boucle « do, while » et ainsi d'arrêter le chronomètre.

```
void Chrono(void)       //La fonction chronomètre
{
 ms=0;          //Permet de mettre les milisecondes à 0
 Usec=0;        //Permet de mettre les unités de seconde à 0
 Dsec=0;        //Permet de mettre les dizaines de seconde à 0


 do                         //On fait...
 {
 ms++;                      //Incrémente un compteur toutes les 0,1sec

 delay_ms(10);              //Permet d'incrémenter le compteur avec comme base de temps les milisecondes

 if(ms>9)                   //Si la variable miliseconde arrive à 9
  {
  ms=0;                     //Miliseconde revient à 0 pour un nouveau cycle;
  Usec++;                   //Nouvelle variable incrémentée et représentant les unités de seconde
  }
 if(Usec>9)                 //Si la variable unité de seconde arrive à 9
  {
  Usec=0;                   //Unité de seconde revient à 0 pour un nouveau cycle
  Dsec++;                   // Nouvelle variable incrémentée et représentant les dizaines de seconde
  }

 lcd_gotoxy(3,0);               //Place le texte sur l'afficheur
 sprintf(tampon1,"%d",ms);      //Permet de voir le défilement des milisecondes
 lcd_puts(tampon1);             //Variable permettant le fonctionnement de la fonction « sprintf(); »
 delay_ms(10);                  //Ce délai permet le bon fonctionnement de la fonction « sprintf(); »

 lcd_gotoxy(2,0);               //Place le texte sur l'afficheur
 lcd_putsf(",");                //Permet d'écrire « , »

 lcd_gotoxy(1,0);               //Place le texte sur l'afficheur
 sprintf(tampon2,"%d",Usec);    //Permet de voir le défilement des unités de seconde
 lcd_puts(tampon2);             //Variable permettant le fonctionnement de la fonction « sprintf(); »
 delay_ms(10);                  //Ce délai permet le bon fonctionnement de la fonction « sprintf(); »
```

« PINB.2 » représente le premier détecteur de la borne d'arrivée et « PINB.3 » représente le deuxième détecteur de cette même borne. La détection du kart se fait à 0 pour la simulation, et inversement, quand le kart n'est pas détecter la détection est à 1.

La fonction « sprintf(); » permet d'afficher et de voir défiler le chronomètre ; pour cela il a fallu mettre à la suite une variable "tampon" puis un « delay_ms(); ».

5.2.5. La fonction « main » :

Nous découperons la fonction « main » en deux parties, pour faciliter son explication. La première partie de programmation portera sur la borne de départ et la seconde sur la borne d'arrivée. Cette fonction est réalisée dans une boucle infinie « while(1) ».

a) Programmation de la borne de départ :

La borne de départ a pour fonction de donner le top départ, de lancer le chronomètre et de détecter les faux départs. Ce début de fonction est essentiellement composé de structures « do, while ».

Pour commencer, tant que le kart n'est pas devant le premier détecteur, on appelle la fonction feux orange clignotant.

Ensuite, on rentre dans une boucle « do, while » et on met la variable « drapeau » à 0. Cette boucle « do, while » servira à détecter les faux départs du kart, au moment du feux orange. Le faux départ sera détecté quand nous sortirons de cette boucle, pour cela la variable « drapeau » devra être à 1.

Après nous rentrons dans une seconde boucle « do, while » qui servira à détecter un faux départ au moment du feux rouge. Dans cette seconde boucle, nous mettrons la variable « drap » et « x » à 0. Comme précédemment, pour détecter le faux départ nous devrons sortir de cette boucle et pour cela nous devrons avoir la variable drap à 1.

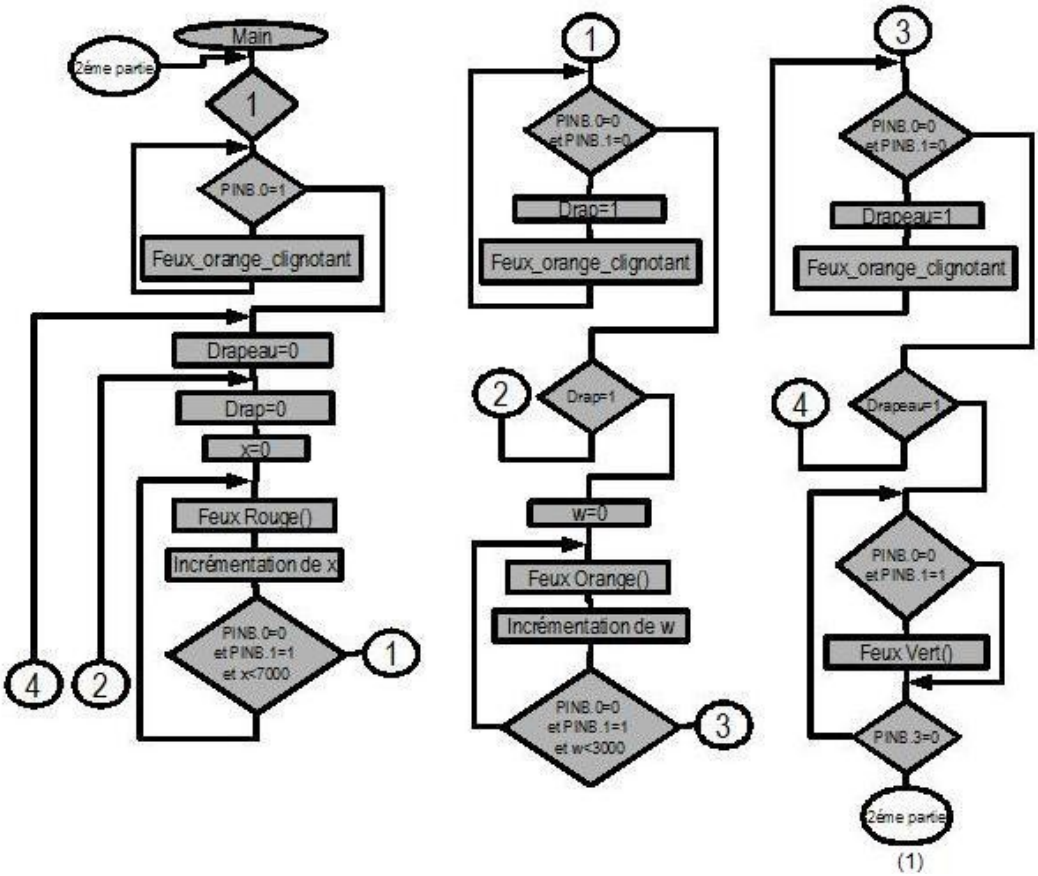


Illustration 13: Ordinogramme de la 1èr partie du" main"

Puis nous rentrons dans une troisième boucle, celle du feux rouge, dans laquelle on appellera la fonction « feux rouge », afin d'allumer la simulation du feux, puis on incrémente la variable x. Tant que le kart sera devant le premier détecteur, sans être détecté par le second, il n'y aura pas de faux départ. La variable « x » pourra s'incrémenter jusqu'à la valeur 7000 et le feux passera du rouge au orange.

```
while (1)                          //Fonction main
   {
    while ( PINB.0 == 1 )          //Tant que le kart n'est pas devant le premier détecteur de la borne de départ...
    {
     Feux_Orange_Clignotant();              //On appelle la fonction Feux_Orange_Clignotant
    }

    do                             //On fait...
    {
     drapeau=0;                    //On met la variable drapeau à 0
     do                            //On fait...
     {
     drap=0;                       //On met la variable drap à 0
     x=0;                          //On met la variable x à 0
     do                            //On fait...
     {
     Feux_R();                     //On appelle la fonction Feux_R
     x++;                          //Incrémente la variable x
     }
     while(PINB.0 == 0 && PINB.1 == 1 && x<7000 );    //...Tant que le kart est devant le 1er détecteur et n'est pas
                                                       devant le second de la borne de départ et que la variable
                                                       x est strictement inférieur à 7000

     while( PINB.0 == 0 && PINB.1 == 0 )               //Tant que le kart est devant les deux capteurs...
     {
       drap=1;                                         //On met la variable « drap » à 1
       Feux_Orange_Clignotant();                       //On appelle la foncton « Feux_Orange_Clignotant »
     }
    }
    while(drap==1);                                    //...Tant que la variable « drap » est à 1
```

Par contre si le kart passe devant le deuxième détecteur il sortira de la troisième boucle « do, while » et rentrera dans une boucle « while » qui fera appel à la fonction « feux_orange_clignotant » et la variable drap sera mise à 1. Le feux orange clignotera tant que le kart ne se sera pas replacé. Une fois le kart replacé et grâce au forçage de mise à 1 de la variable drap, nous pourrons revenir au feux rouge de départ et ainsi reprendre une tentative de départ.

Quand le feux passe au orange, on a auparavant mis la variable « w » à 0, et nous rentrons dans la quatrième boucle « do, while », celle du feux orange, dans laquelle la fonction « feux orange » sera appelée. La suite se passe comme pour le feux rouge, une variable sera incrémentée pour donner une durée au feux. Dans le même cas que pour le feux rouge, un faux départ pourra être détecté, et par le même processus on pourra revenir au feux rouge afin de retenter sa chance. Une fois la variable « w » à 3000, le feux passe au vert.

```
w=0;                                   //On met la variable w à 0
do                                     //On fait...
{
Feux_O();                              //On appelle la fonction « Feux orange »
w++;                                   //On incrémente la variable w
}
while (PINB.0 == 0 && PINB.1 == 1 && w<3000);    //...Tant que le kart est devant le 1er détecteur et n'est pas
                                                  devant le second de la borne de départ et que la variable
                                                  w est strictement inférieur à 3000

while( PINB.0 == 0 && PINB.1 == 0 )    //Tant que le kart est devant les deux capteurs...
{
 drapeau=1;                            //On met la variable « drapeau » à 1
 Feux_Orange_Clignotant();             //On appelle la foncton « Feux_Orange_Clignotant »
}
}
while(drapeau==1);                     //...Tant que la variable « drapeau » est à 1

do                                     //On fait...
{
if (PINB.0 == 0 && PINB.1 == 1 )       //Si le kart n'est vu que par le premier détecteur de la
                                         borne de départ...
 {
 Feux_V();                             //On appelle la fonction « Feux vert »
 }
}
while( PINB.3 == 0 );                  //...Tant que le kart n'est pas vu par le premier détecteur
                                         de la borne d'arrivée
```

Le feux passera au vert si le kart n'est pas détecté en faux départ, ainsi nous rentrerons dans la dernière boucle « do, while », celle du feux vert, et nous appellerons la fonction « feux vert » afin de l'allumer. Le chronomètre s'enclenchera dans la fonction feux vert, et une fois que le kart passe devant le dernier détecteur de la borne d'arrivée, nous sortirons de la boucle du feux vert. Le feux passera du vert au orange clignotant et le chronomètre s'arrêtera.

b) Programmation de la borne d'arrivée :

La borne d'arrivée a pour fonction de calculer la vitesse du kart et d'arrêter le chronomètre. Cette partie sera composée d'une seule boucle « do, while ».

La première partie du programme se termine par le passage du kart sur le dernier détecteur. A se passage, la valeur (T2) du chronomètre est relevée. Le moyen de relever cette valeur (T2) est exactement la même que pour relever (T1) (expliqué dans la partie de programmation de la fonction « chronomètre »). Cette valeur sera égale à « Y ».

Ensuite le calcul de la vitesse s'effectuera comme expliquer dans la partie 4.1.2. . La vitesse du kart sera égale à « V ». La formule informatique sera sous la forme suivante :

V = 0.0003/((Y-X)/3600)

La variable « Y » représente le temps (T2) et la variable « X » représente le temps (T1).

Pour finir nous rentrerons dans le dernière boucle « do, while » de la fonction « main ». A l'intérieur de celle-ci nous afficherons le chronomètre arrêté et la valeur de la vitesse du kart. Ces valeurs resterons affichées jusqu'à la présentation d'un nouveau kart au départ, ce qui activera le premier détecteur de la borne d'arrivée et donc nous fera sortir de la dernière boucle. La présence du nouveau kart enclenchera le feux ; pour démarrer une nouvelle course.

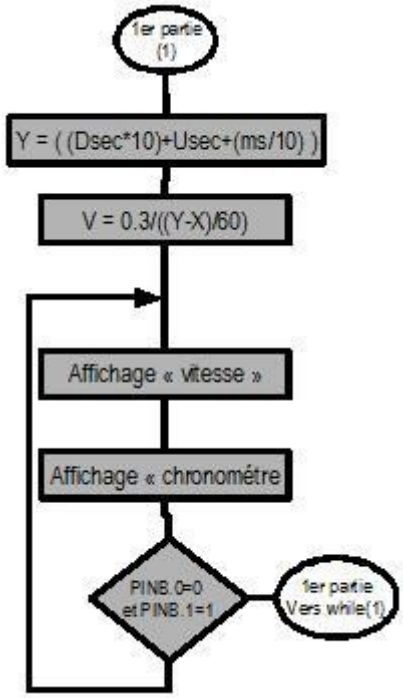
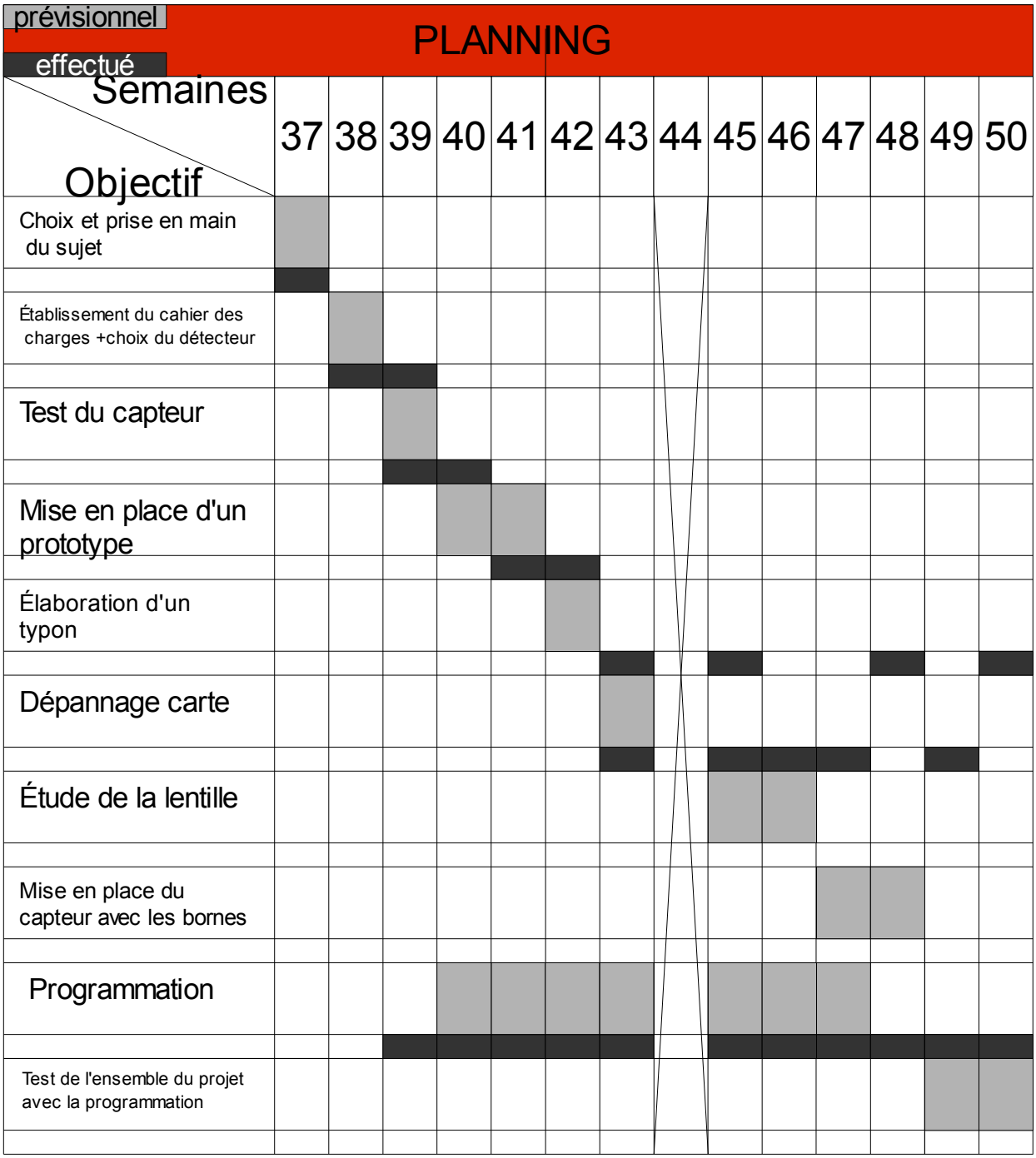


Illustration 14: Ordinogramme de la 2ème partie du" main"

```
    Y = ( (Dsec*10)+Usec+(ms/10) );          //La variable Y va retenir la deuxième valeur du chronométre
    V = 0.3/((Y-X)/60);                      //La variable V va correspondre à la vitesse du kart

    do                                       //On fait...
    {
    lcd_gotoxy(4,0);                         //Place le texte sur l'afficheur
    lcd_putsf("Vitesse=");                   //Permet d'écrire « Vitesse= »

    lcd_gotoxy(13,0);                        //Place le texte sur l'afficheur
    sprintf(tampon4,"%f",V);                 //Permet de voir la vitesse du kart
    lcd_puts(tampon4);                       //Variable permettant le fonctionnement de la fonction« sprintf(); »
    delay_ms(10);                            //Ce délai permet le bon fonctionnement de la fonction « sprintf(); »

    lcd_gotoxy(3,1);                         //Place le texte sur l'afficheur
    sprintf(tampon1,"%d",ms);                //Permet de voir le défilement des milisecondes
    lcd_puts(tampon1);                       //Variable permettant le fonctionnement de la fonction« sprintf(); »
    delay_ms(10);                            //Ce délai permet le bon fonctionnement de la fonction « sprintf(); »
    lcd_gotoxy(2,1);                         //Place le texte sur l'afficheur
    lcd_putsf(",");                          //Permet d'écrire « , »
    lcd_gotoxy(1,1);                         //Place le texte sur l'afficheur
    sprintf(tampon2,"%d",Usec);              //Permet de voir le défilement des unités de seconde
    lcd_puts(tampon2);                       //Variable permettant le fonctionnement de la fonction « sprintf(); »
    delay_ms(10);                            //Ce délai permet le bon fonctionnement de la fonction « sprintf(); »
    lcd_gotoxy(0,1);                         //Place le texte sur l'afficheur
    sprintf(tampon3,"%d",Dsec);              //Permet de voir le défilement des dizaines de seconde
    lcd_puts(tampon3);                       //Variable permettant le fonctionnement de la fonction « sprintf(); »
    delay_ms(10);                            //Ce délai permet le bon fonctionnement de la fonction « sprintf(); »
    }
    while(PINB.0 == 0 && PINB.1 == 1);       //...Tant que le kart est devant le 1er détecteur et n'est pas
                                               devant le second de la borne de départ


   };
}
```

6. Planning :



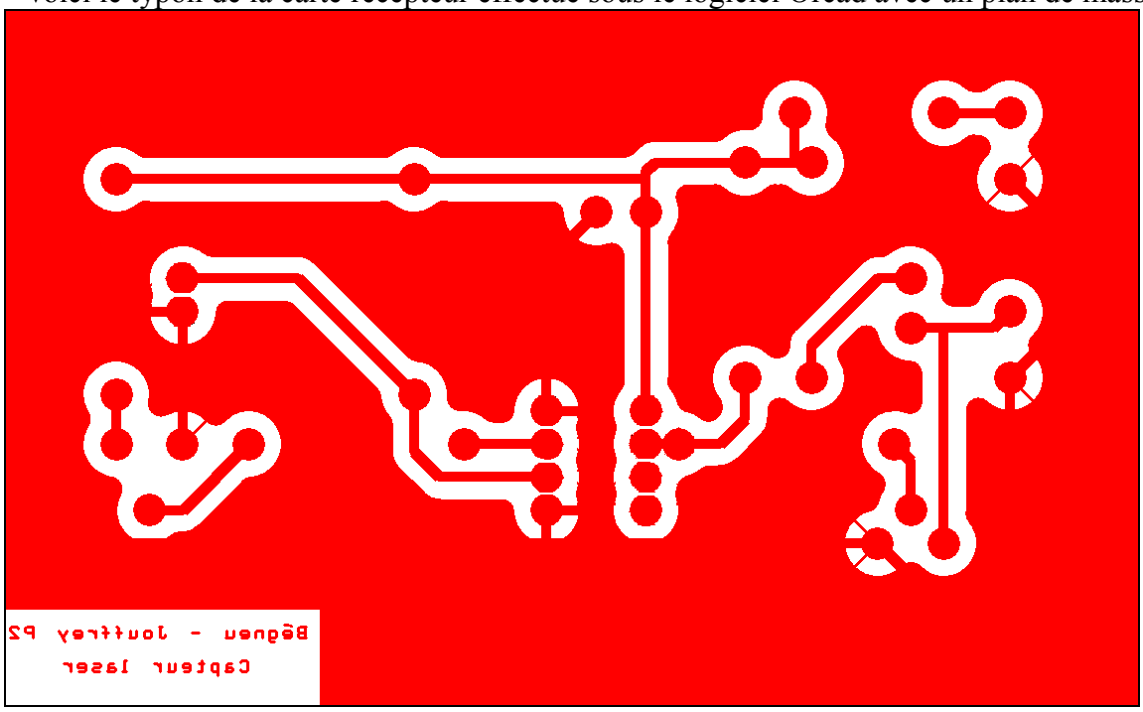
Nous n'avons pas pu créer un typon correct comme le planning le prévoyait, dû à de nombreux problèmes au niveau de la partie dépannage de la carte et de l'élaboration du typon. Nous avons donc décidé de travailler plus en profondeur la programmation pour avoir un programme le plus fidèle au cahier des charges.

Annexe :

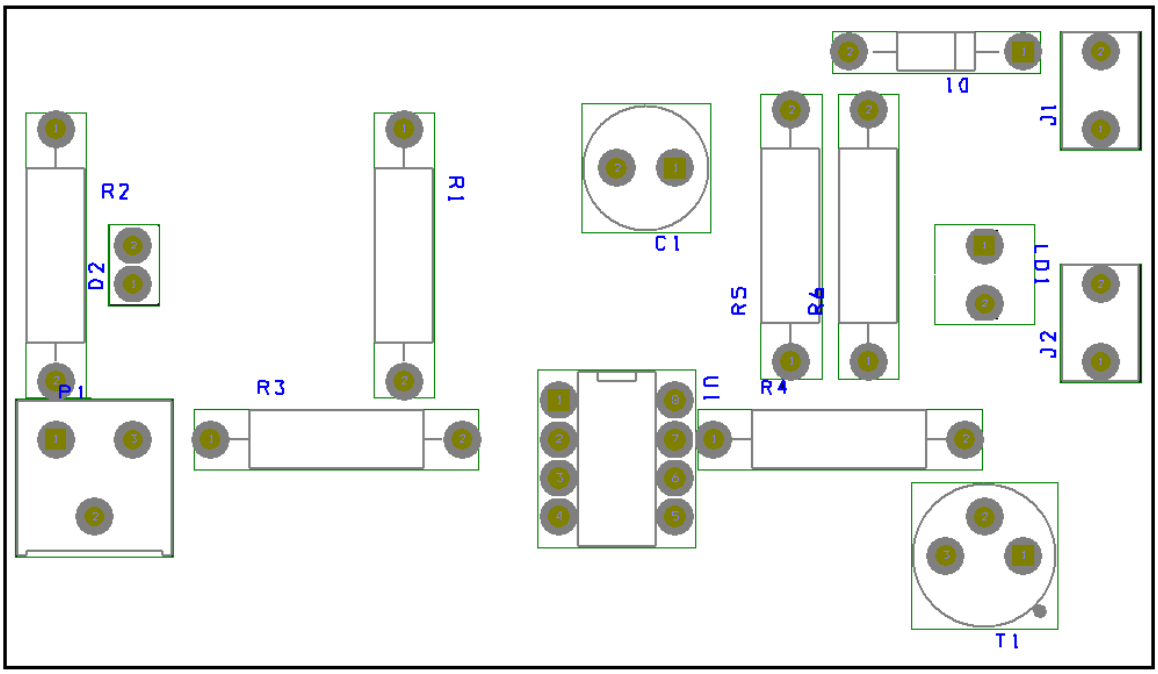
La documentation des composants se trouve sur les URL suivantes:

- LM311 : http://www.datasheetcatalog.com/datasheets_pdf/L/M/3/1/LM311.shtml
- BPW34 : http://www.datasheetcatalog.com/datasheets_pdf/B/P/W/3/BPW34.shtml
- 2N2222 : http://www.datasheetcatalog.com/datasheets_pdf/2/N/2/2/2N2222.shtml

Voici le typon de la carte récepteur effectué sous le logiciel Orcad avec un plan de masse.



L'implantation des composants sur le typon précédent.



Programmation :

```
/*****************************************************
This program was produced by the
CodeWizardAVR V1.24.2c Professional
Automatic Program Generator
© Copyright 1998-2004 Pavel Haiduc, HP InfoTech s.r.l.
http://www.hpinfotech.ro
e-mail:office@hpinfotech.ro

Project :
Version :
Date    : 03/10/2008
Author  : F4CG
Company : F4CG
Comments:


Chip type           : ATmega8535
Program type        : Application
Clock frequency     : 16,000000 MHz
Memory model        : Small
External SRAM size  : 0
Data Stack size     : 128
*****************************************************/
#include <mega8535.h>

// Alphanumeric LCD Module functions
#asm
   .equ __lcd_port=0x15 ;PORTC
#endasm
#include <lcd.h>
#include<stdio.h>
#include <delay.h>

//#asm("cli")
void Feux_R(void);
void Feux_O(void);
void Feux_V(void);
void Feux_Orange_Clignotant(void);
void Chrono(void);

int x,w,y,d,Dsec,ms,Usec,drap,drapeau;
float X,Y,V;
unsigned char tampon1[10];
unsigned char tampon2[10];
unsigned char tampon3[10];
unsigned char tampon4[10];

void main(void)
{
// Declare your local variables here

// Input/Output Ports initialization
// Port A initialization
// Func7=In Func6=In Func5=In Func4=In Func3=In Func2=In Func1=In Func0=In
// State7=T State6=T State5=T State4=T State3=T State2=T State1=T State0=T
PORTA=0x00;
DDRA=0x00;
```

```
// Port B initialization
// Func7=In Func6=In Func5=In Func4=In Func3=In Func2=In Func1=In Func0=In
// State7=T State6=T State5=T State4=T State3=T State2=T State1=T State0=T
PORTB=0x00;
DDRB=0x00;

// Port C initialization
// Func7=In Func6=In Func5=In Func4=In Func3=In Func2=In Func1=In Func0=In
// State7=T State6=T State5=T State4=T State3=T State2=T State1=T State0=T
PORTC=0x00;
DDRC=0x00;

// Port D initialization
// Func7=In Func6=In Func5=In Func4=In Func3=In Func2=In Func1=In Func0=In
// State7=T State6=T State5=T State4=T State3=T State2=T State1=T State0=T
PORTD=0x00;
DDRD=0x00;

// Timer/Counter 0 initialization
// Clock source: System Clock
// Clock value: Timer 0 Stopped
// Mode: Normal top=FFh
// OC0 output: Disconnected
TCCR0=0x00;
TCNT0=0x00;
OCR0=0x00;

// Timer/Counter 1 initialization
// Clock source: System Clock
// Clock value: Timer 1 Stopped
// Mode: Normal top=FFFFh
// OC1A output: Discon.
// OC1B output: Discon.
// Noise Canceler: Off
// Input Capture on Falling Edge
TCCR1A=0x00;
TCCR1B=0x00;
TCNT1H=0x00;
TCNT1L=0x00;
ICR1H=0x00;
ICR1L=0x00;
OCR1AH=0x00;
OCR1AL=0x00;
OCR1BH=0x00;
OCR1BL=0x00;

// Timer/Counter 2 initialization
// Clock source: System Clock
// Clock value: Timer 2 Stopped
// Mode: Normal top=FFh
// OC2 output: Disconnected
ASSR=0x00;
TCCR2=0x00;
TCNT2=0x00;
OCR2=0x00;

// Timer/Counter 0 initialization
// Clock source: System Clock
// Clock value: 15,625 kHz
```

```
// Mode: Normal top=FFh
// OC0 output: Disconnected
TCCR0=0x05;
TCNT0=0x03;
OCR0=0x00;

// External Interrupt(s) initialization
// INT0: Off
// INT1: Off
// INT2: Off
MCUCR=0x00;
MCUCSR=0x00;

// Timer(s)/Counter(s) Interrupt(s) initialization
TIMSK=0x00;

// Analog Comparator initialization
// Analog Comparator: Off
// Analog Comparator Input Capture by Timer/Counter 1: Off
// Analog Comparator Output: Off
ACSR=0x80;
SFIOR=0x00;

#asm("sei")

// LCD module initialization
lcd_init(16);

x=0;                                    //On met la variable x à 0
y=0;                                    //On met la variable y à 0
w=0;                                    //On met la variable w à 0

while (1)                               //Fonction main
    {
     while ( PINB.0 == 1 )              //Tant que le kart n'est pas devant le premier détecteur de la borne de départ...
     {
      Feux_Orange_Clignotant();         //On appelle la fonction  Feux_Orange_Clignotant
     }

     do                                 //On fait...
     {
      drapeau=0;                        //On met la variable drapeau à 0
      do                                //On fait...
      {
       drap=0;                          //On met la variable drap à 0
       x=0;                             //On met la variable x à 0
       do                               //On fait...
       {
        Feux_R();                       //On appelle la fonction  Feux_R
        x++;                            //Incrémente la variable x
       }
       while(PINB.0 == 0 && PINB.1 == 1 && x<7000 );     //...Tant que le kart est devant le 1er détecteur et n'est pas
                                                          devant le second de la borne de départ et que la variable
                                                          x est strictement inférieur à 7000

       while( PINB.0 == 0 && PINB.1 == 0 )               //Tant que le kart est devant les deux capteurs...
        {
         drap=1;                                         //On met la variable « drap » à 1
         Feux_Orange_Clignotant();                       //On appelle la foncton « Feux_Orange_Clignotant »
        }
```

```
 }
 while(drap==1);                                    //...Tant que la variable « drap » est à 1


 w=0;                                               //On met la variable w à 0
 do                                                 //On fait...
 {
  Feux_O();                                         //On appelle la fonction « Feux orange »
  w++;                                              //On incrémente la variable w
 }
 while (PINB.0 == 0 && PINB.1 == 1 && w<3000);      //...Tant que le kart est devant le 1er détecteur et n'est pas
                                                      devant le second de la borne de départ et que la variable
                                                      w est strictement inférieur à 3000
 while( PINB.0 == 0 && PINB.1 == 0 )                //Tant que le kart est devant les deux capteurs...
 {
  drapeau=1;                                        //On met la variable « drapeau » à 1
  Feux_Orange_Clignotant();                         //On appelle la foncton « Feux_Orange_Clignotant »
 }
}
while(drapeau==1);                                  //...Tant que la variable « drapeau » est à 1

do                                                  //On fait...
{
 if (PINB.0 == 0 && PINB.1 == 1 )                   //Si le kart n'est vu que par le premier détecteur de la
                                                      borne de départ...
  {
   Feux_V();                                        //On appelle la fonction « Feux vert »
  }
}
while( PINB.3 == 0 );                               //...Tant que le kart n'est pas vu par le premier détecteur
                                                      de la borne d'arrivée

Y = ( (Dsec*10)+Usec+(ms/10) );                     //La variable Y va retenir la deuxième valeur du chronométre
 V = 0.3/((Y-X)/60);                                //La variable V va correspondre à la vitesse du kart

do                                                  //On fait...
{
 lcd_gotoxy(4,0);                                   //Place le texte sur l'afficheur
 lcd_putsf("Vitesse=");                             //Permet d'écrire « Vitesse= »

 lcd_gotoxy(13,0);                                  //Place le texte sur l'afficheur
 sprintf(tampon4,"%f",V);                           //Permet de voir la vitesse du kart
 lcd_puts(tampon4);                                 //Variable permettant le fonctionnement de la fonction« sprintf(); »
 delay_ms(10);                                      //Ce délai permet le bon fonctionnement de la fonction « sprintf(); »

 lcd_gotoxy(3,1);                                   //Place le texte sur l'afficheur
 sprintf(tampon1,"%d",ms);                          //Permet de voir le défilement des milisecondes
 lcd_puts(tampon1);                                 //Variable permettant le fonctionnement de la fonction« sprintf(); »
 delay_ms(10);                                      //Ce délai permet le bon fonctionnement de la fonction « sprintf(); »
 lcd_gotoxy(2,1);                                   //Place le texte sur l'afficheur
 lcd_putsf(",");                                    //Permet d'écrire « , »
 lcd_gotoxy(1,1);                                   //Place le texte sur l'afficheur
 sprintf(tampon2,"%d",Usec);                        //Permet de voir le défilement des unités de seconde
 lcd_puts(tampon2);                                 //Variable permettant le fonctionnement de la fonction « sprintf(); »
 delay_ms(10);                                      //Ce délai permet le bon fonctionnement de la fonction « sprintf(); »
 lcd_gotoxy(0,1);                                   //Place le texte sur l'afficheur
 sprintf(tampon3,"%d",Dsec);                        //Permet de voir le défilement des dizaines de seconde
 lcd_puts(tampon3);                                 //Variable permettant le fonctionnement de la fonction « sprintf(); »
 delay_ms(10);                                      //Ce délai permet le bon fonctionnement de la fonction « sprintf(); »
}
```

```
      while(PINB.0 == 0 && PINB.1 == 1);            //...Tant que le kart est devant le 1er détecteur et n'est pas
                                                      devant le second de la borne de départ

    };
}

void Feux_R(void)    //Fonction feux rouge
 {
  lcd_gotoxy(0,1);              //Place le texte sur l'afficheur
  lcd_putsf("feux rouge");    //Permet d'écrire « feux rouge »

  lcd_gotoxy(0,2);          //Place le texte sur l'afficheur
  lcd_putsf("          ");    //Permet d'effacer l'écriture « feux orange »
  lcd_gotoxy(0,3);          //Place le texte sur l'afficheur
  lcd_putsf("          ");  //Permet d'effacer l'écriture « feux vert »
 }

void Feux_O(void)          //Fonction feux orange
 {
 lcd_gotoxy(0,2);               //Place le texte sur l'afficheur
 lcd_putsf("feux orange");    //Permet d'écrire « feux orange»

 lcd_gotoxy(0,1);             //Place le texte sur l'afficheur
 lcd_putsf("          ");        // Permet d'effacer l'écriture « feux rouge »
 }

void Chrono(void)        //La fonction chronométre
{
 ms=0;            //Permet de mettre les milisecondes à 0
 Usec=0;          //Permet de mettre les unités de seconde à 0
 Dsec=0;          //Permet de mettre les dizaines de seconde à 0


 do                           //On fait...
 {
  ms++;                       //Incrémente un compteur toutes les 0,1sec

  delay_ms(10);               //Permet d'incrémenter le compteur avec comme base de temps les milisecondes

  if(ms>9)                    //Si la variable miliseconde arrive à 9
   {
    ms=0;                     //Miliseconde revient à 0 pour un nouveau cycle;
    Usec++;                   //Nouvelle variable incrémentée et représentant les unités de seconde
   }
  if(Usec>9)                  //Si la variable unité de seconde arrive à 9
   {
    Usec=0;                   //Unité de seconde revient à 0 pour un nouveau cycle
    Dsec++;                   // Nouvelle variable incrémentée et représentant les dizaines de seconde
   }

  lcd_gotoxy(3,0);                  //Place le texte sur l'afficheur
  sprintf(tampon1,"%d",ms);         //Permet de voir le défilement des milisecondes
  lcd_puts(tampon1);                //Variable permettant le fonctionnement de la fonction « sprintf(); »
  delay_ms(10);                     //Ce délai permet le bon fonctionnement de la fonction « sprintf(); »

  lcd_gotoxy(2,0);                  //Place le texte sur l'afficheur
  lcd_putsf(",");                   //Permet d'écrire « , »

  lcd_gotoxy(1,0);                  //Place le texte sur l'afficheur
  sprintf(tampon2,"%d",Usec);       //Permet de voir le défilement des unités de seconde
```

```
  lcd_puts(tampon2);                  //Variable permettant le fonctionnement de la fonction « sprintf(); »
  delay_ms(10);                       //Ce délai permet le bon fonctionnement de la fonction « sprintf(); »

  lcd_gotoxy(0,0);                    //Place le texte sur l'afficheur
  sprintf(tampon3,"%d",Dsec);         //Permet de voir le défilement des dizaines de seconde
  lcd_puts(tampon3);                  //Variable permettant le fonctionnement de la fonction « sprintf(); »
  delay_ms(10);                       //Ce délai permet le bon fonctionnement de la fonction « sprintf(); »

  if(PINB.2 == 0)                     //Quand le kart est devant le premier détecteur de la borne d'arrivée
   {
    X = ( (Dsec*10)+Usec+(ms/10) );  //La variable X va retenir la première valeur du chronométre
   }

  if(PINB.3 == 0)                     //Quand le kart est devant le deuxième détecteur de la borne d'arrivée
   d=0;                               //La variable d passe à 0
 }
 while(d==1);                         //...Tant que la variable d est à 1, on reste dans la fonction chronométre
}

void Feux_V(void)             //Fonction feux vert
 {
   lcd_gotoxy(0,1);             //Place le texte sur l'afficheur
   lcd_putsf("          ");     // Permet d'effacer l'écriture «feux rouge»
   lcd_gotoxy(0,2);             //Place le texte sur l'afficheur
   lcd_putsf("          ");     //Permet d'effacer l'écriture «feux orange»
   lcd_gotoxy(0,3);             //Place le texte sur l'afficheur
   lcd_putsf("feux vert");      //Permet d'écrire «feux vert»
   d=1;                         //On met la variable d à 1
   Chrono();                    //On appelle la fonction Chrono
 }

void Feux_Orange_Clignotant (void)          //Fonction Feux orange clignotant
{
       if(y==0)          //Quand la variable y est à 0
        {
         lcd_gotoxy(0,2);              //Place le texte sur l'afficheur
         lcd_putsf("feux orange");     //Permet d'écrire «feux orange»

         lcd_gotoxy(0,1);              //Place le texte sur l'afficheur
         lcd_putsf("          ");      //Permet d'effacer l'écriture « feux rouge »
         lcd_gotoxy(0,3);              //Place le texte sur l'afficheur
         lcd_putsf("          ");      //Permet d'effacer l'écriture « feux vert »

        delay_ms(1000);                //Laisse afficher pendant un court temps « feux orange »
         y=1;                          //Met la variable y à 1
        }

       else                            //Quand la variable y est à 1
        {
         lcd_gotoxy(0,1);              //Place le texte sur l'afficheur
         lcd_putsf("          ");      //Permet d'effacer l'écriture « feux rouge »
         lcd_gotoxy(0,2);              //Place le texte sur l'afficheur
         lcd_putsf("          ");      //Permet d'effacer l'écriture « feux orange»
         lcd_gotoxy(0,3);              //Place le texte sur l'afficheur
         lcd_putsf("          ");      //Permet d'effacer l'écriture « feux vert »

        delay_ms(1000);                //Efface les écritures pendant un court temps
         y=0;                          //Met la variable y à 0
        }
}
```

Index des illustrations

Les images sont en partie tirés du site de M. LEQUEU www.**thierry-lequeu**.fr/ ou sont de composition personnel.

Bibliographie

[1] , "", [En ligne]. <www.alldatasheet.com/> (Page consultée le).
[2] **Thierry LEQUEU**, "", [En ligne]. <http://www.lequeuth.com/http://www.lequeuth.com/> (Page consultée le).

Les sources n'auront étaient que des sources internet, elles auront était consultées à différentes dates lors du déroulement du projet. Nous nous sommes inspiré d'un schémas, sur la notice d'utilisation d'un capteur laser.