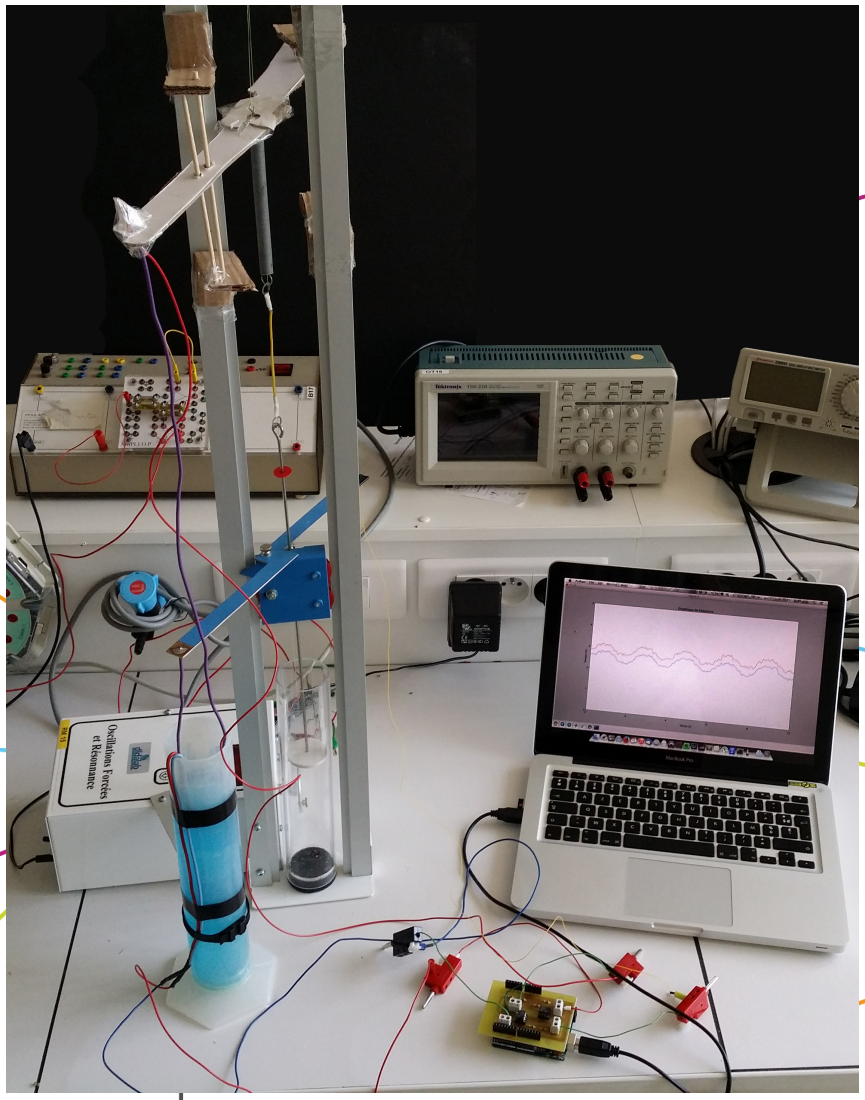
INSA INSTITUT NATIONAL DES SCIENCES APPLIQUÉES ROUEN

Projet de Physique P6
STPI/P6/2015-12

Mise en place d'une mesure de phase sur le TP de résonance mécanique



Enseignants responsables
Corentin JOUEN
Jérôme YON

Étudiants :
Gautier DARCHEN
Arthur LEBRÊNE
Baptiste ROUSSEAU
Alexis DURIEUX
Haochun MA
Hervé WAN

Date de remise du rapport : 15 juin 2015

Référence du projet : STPI [1] / P6 / 2015 - 12

Intitulé du projet : Mise en place d'une mesure de phase sur le TP de résonance mécanique

Type de projet : expérimental

Objectifs du projet :

Il existe pour les élèves-ingénieurs de première année un montage expérimental utilisé dans le cadre du cours « P2 - Mécanique du point ». Ce montage consiste en un mobile guidé verticalement en translation, en régime sinusoïdal forcé par un moteur. L'emploi d'un ressort entre ledit moteur et le mobile, couplé à des forces de frottements générées par une partie du mobile dans l'eau, entraîne un déphasage entre les oscillations du mobile et du moteur.

L'objectif de ce projet est donc d'améliorer le système, afin d'obtenir la phase avant ce ressort, c'est-à-dire la phase du moteur.

Remerciements

Nous tenons particulièrement à remercier M. Pascal WILLIAMS et Mme Hélène RADE pour leurs connaissances, leur disponibilité, leur écoute et leur compréhension, qui ont été des atouts décisifs dans la réalisation de ce projet.

Sans leur aide précieuse, notre travail n'aura pas été mené à terme.

Nota bene :

L'intitulé officiel de notre projet est « Mise en place d'une mesure de *déphasage* sur le TP de résonance mécanique ». Cependant, la mesure du déphasage étant déjà en place – il s'agit du TP en lui-même – et notre travail consistant à mesurer la phase, nous avons décidé, après aval de M. Corentin JOUEN, de remplacer le terme *déphasage* de notre intitulé par *phase*.

1. INSTITUT NATIONAL DES SCIENCES APPLIQUÉES DE ROUEN

Département Sciences et Techniques Pour l'Ingénieur
685 Avenue de l'Université BP 08- 76801 Saint-Étienne-du-Rouvray
Tél : +33 2 32 95 66 21 - Fax : +33 2 32 95 66 31

Table des matières

Introduction

Dans le cadre du deuxième semestre de la deuxième année au sein du département des Sciences et Techniques Pour l'Ingénieur, l'INSA de Rouen nous conviait à réaliser un projet de physique par groupe de six personnes afin de réaliser l'amélioration d'un système mécanique basé sur la notion de pendule élastique. Couramment utilisé en STPI 1 dans le cadre d'un TP portant sur la mécanique du point, l'amélioration du système permettrait une interprétation plus poussée des résultats.

L'objectif d'un tel projet est de concilier plusieurs compétences et savoir-faire de l'ingénieur parmi lesquels on trouve l'amélioration d'un produit, la planification d'un projet à moyen terme ainsi que le travail en équipe.

Cela est d'autant plus intéressant qu'un large ensemble de moyens techniques nous est mis à disposition afin de réaliser l'amélioration du système ainsi qu'un budget nous permettant de considérer la faisabilité des solutions envisagées.

La planification du projet, quant à elle, s'est découpée en deux phases. La première fut une phase de recherche permettant d'appréhender le problème étudié ainsi que de trouver des solutions envisageables à notre échelle et de finalement en retenir une seule. La seconde phase fut une phase de test de la solution retenue afin de vérifier sa faisabilité et sa validité quant aux critères d'attente de l'amélioration du système mécanique. Cette phase fut accompagnée d'une modélisation des différents composants à l'aide de logiciels spécifiques.

Ce projet nous permet finalement de mettre en application nos différentes connaissances concernant le domaine de la mécanique, de l'électricité et l'électronique au travers des solutions apportées au problème abordé.

Dans ce présent rapport, seront détaillés d'une part le problème étudié ainsi que les solutions envisagées, et d'autre part la solution retenue et mise en œuvre. Enfin, les différentes difficultés rencontrées au cours du projet seront mentionnées et détaillées.

Première partie

Méthodologie et organisation du travail

1.1 Suivi chronologique par séance

Lors de la première séance, M. JOUEN nous a donné le cahier des charges, et nous a expliqué que notre travail consistait à mettre en place une mesure de phase.

Pour ce faire, il a fallu lister les systèmes pouvant être mis en place. Durant la première semaine, chaque membre du groupe a donc fait ses propres recherches afin de trouver un ou plusieurs dispositifs pouvant répondre à l'objectif principal, sans se soucier du coût ni de la faisabilité technique.

Lors de la deuxième séance, les recherches ont été mises en commun afin de les comparer et de prendre en compte, cette fois-ci, leur coût et leur faisabilité. Deux systèmes se sont alors dégagés : l'ensemble bielle-manivelle et le code optique GRAY – ces dispositifs seront détaillés plus loin.

Afin d'approfondir nos connaissances de ces deux systèmes, les deux semaines suivantes, Alexis, Gautier et Hervé ont travaillé sur le code GRAY, tandis que Baptiste et Haochun se sont concentrés sur le dispositif bielle-manivelle.

Baptiste s'est également joint à Arthur afin de proposer de miniaturiser le circuit électrique, ce que M. JOUEN a accepté. Il leur a donc fallu effectuer des relevés sur le circuit afin de le comprendre en détail.

Au terme de ces 4 premières semaines, il est apparu évident que le système bielle-manivelle ne pouvait être mis en place. Il a donc fallu pousser plus avant les recherches concernant le code GRAY, ce dont Haochun et Hervé se sont occupés. Alexis et Gautier ont alors pu se concentrer sur le Arduino et son langage, dont nous avions besoin pour implémenter ce système.

Arthur et Baptiste ont quant à eux continué la miniaturisation du circuit.

Notre idée initiale concernant le GRAY était de le fabriquer nous-même, au moyen de DEL [2] et de photodiodes. Les recherches ont montré qu'ils existent des systèmes industriels prêts à l'emploi. Haochun et Hervé ont alors cherché un dispositif « bon marché ».

Ne connaissant pas le montant exact de notre budget, ils ont soumis à M. JOUEN leurs résultats, mais celui-ci nous a affirmé que toutes ces propositions restaient trop chères.

Nous avons donc abandonné ce dispositif à la 7e séance. Nous avons alors choisi de développer un autre système : mettre un second mobile, fonctionnant sur le même principe que le précédent, sur le montant.

C'est également pendant cette 7e séance que M. JOUEN nous a annoncé qu'il cesserait d'être notre tuteur, pour raisons personnelles. Il nous a cependant assuré qu'il continuerait d'être disponible par e-mail et que M. YON viendrait régulièrement nous superviser.

Nous avons alors choisi de conserver le Arduino, puisque cela représentait un atout indéniable pour améliorer ce TP – les avantages seront détaillés plus loin. Cela a aussi permis à Alexis et Gautier de proposer un nouveau logiciel d'analyse des tensions récupérées, développé par leurs soins.

Durant les 5 séances restantes, ces derniers se sont occupés du développement dudit logiciel, tandis qu'Arthur a terminé la réalisation du circuit électronique.

Baptiste, Haochun et Hervé, quant à eux, se sont concentrés sur la modélisation physique du montage, et la production des pièces mécaniques nécessaires au TP.

2. Diode ÉlectroLuminescente.

Enfin, la rédaction du rapport a été transversale et a occupé tous les membres durant toute la durée du projet.

1.2 Suivi des tâches par exécutant

Nous présentons ici sous forme graphique l'ensemble des tâches réalisées, et ce pour chacun des membres du projet.

Nous mentionnons également les membres du personnel de l'INSA qui nous ont aidé.

Nota : plus le cadre Tâche est long, plus le travail a demandé de temps.

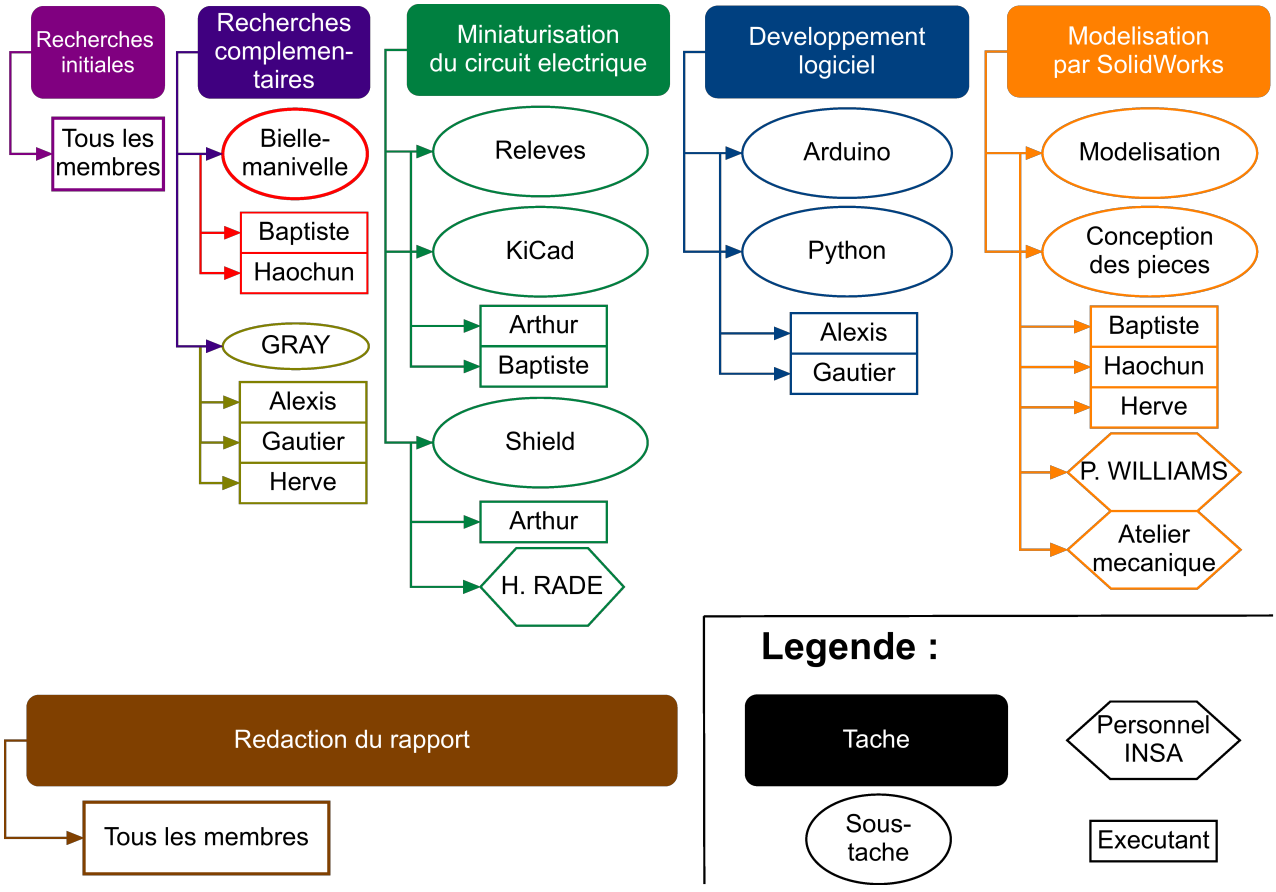


FIGURE 1.1 – Organigramme des tâches

Deuxième partie

Travail réalisé et résultats

Chapitre 2

Présentation du système initial

Ce chapitre s'attache à décrire du mieux possible les constituants du système qu'il nous a été proposé d'améliorer, ceci afin de re-contextualiser notre travail et d'apporter au lecteur suffisament d'éléments pour appréhender les chapitres suivants.

Il présente également le cahier des charges tel qu'il nous a été donné par l'INSA, ce qui a constitué le point de départ des recherches de notre projet.

2.1 Fonctionnement des différents éléments du système

2.1.1 Structure mécanique

Le système mécanique se compose d'un moteur (avec la fréquence sur un écran), un ressort, un fil en nylon qui relie le ressort et le moteur, un oscillateur dont la direction d'oscillation est fixée verticalement, et une tige qui connecte le système mécanique avec le circuit électrique et le détecteur d'oscillation au bout. Avec le roulement du moteur, l'oscillateur peut vibrer périodiquement et nous pouvons trouver la courbe de vibration de l'oscillateur sur ordinateur.

2.1.2 Principe de la mesure de hauteur du mobile

Pour mesurer les variations de hauteur du mobile, une lame en époxy [1] est fixée à celui-ci de manière horizontale. Cette lame comporte une piste en cuivre reliant ses deux extrémités.

L'une d'entre elles est reliée à une tige formée par un câble électrique rigide, ainsi décentrée par rapport au montant, qui évolue selon un axe vertical de la même manière que le mobile.

À l'autre extrémité se trouve un fil électrique relié au système d'acquisition.

Fixé au bloc moteur se trouve une éprouvette contenant une solution aqueuse de sulfate de cuivre – que nous appellerons « solution électrolytique » dans la suite du document.

En haut de cette éprouvette, au contact de la solution, se trouve une borne électrique, la cathode, par laquelle arrive une tension de 0,5V. En bas de l'éprouvette, au contact de la solution, se trouve une autre borne électrique, l'anode, reliée à la masse.

Ce dispositif permet de créer des équipotentiels au sein de la solution : plus on descend vers le fond de l'éprouvette, plus la tension diminue.

1. Polymère rigide.

La tige électrique précédemment citée est dénudée à son extrémité libre, laquelle baigne dans la solution. Le fil relié à l'autre extrémité de la lame époxy permet alors de transmettre la tension relevée.

Le mobile évoluant périodiquement de haut en bas, il en va de même pour la tension récupérée : si le mobile est au plus bas, on lit une tension minimale ; si le mobile est au plus haut, on lit une tension maximale.

2.1.3 Circuit électrique

Le circuit initialement présent sur les montages comprenait une plaquette pédagogique avec 2 AOP [2], 4 résistances et des fils reliant ladite plaquette aux bornes de la solution électrolytique, au mobile ainsi qu'à la rosace d'acquisition Synchronie. Le schéma est donné en figure 2.1 – pour plus de clarté, l'anode et la borne ⊖ de Synchronie, toutes deux reliées à la masse, ne sont pas représentées [3].

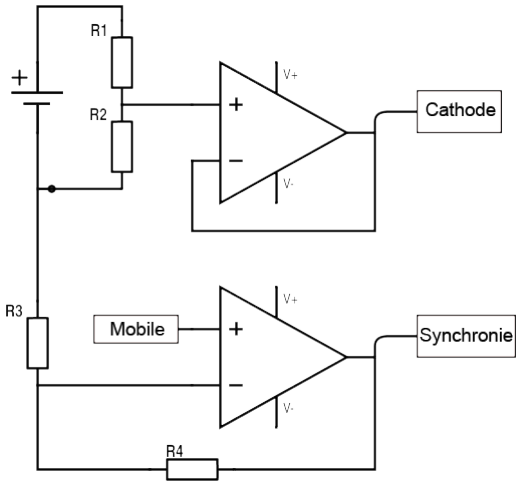


FIGURE 2.1 – Schéma d'ensemble du circuit électrique initial

Le but de ce montage est double. Nous l'avons vu, la solution électrolytique est composée de sulfate de cuivre dans l'eau. Une tension trop forte risquerait de rapidement dégrader cette solution, un pont diviseur de tension est alors utilisé pour passer d'une alimentation de 5V à une tension de 0,5V. Cette tension est ensuite stabilisée par un AOP en montage suiveur, et la sortie de cet AOP est reliée à la cathode [4].

Le deuxième objectif est d'améliorer la tension récupérée par le fil rigide. En effet, cette valeur peut varier entre 0 et 0,5V. Or ces variations sont trop faibles pour que Synchronie puisse les détecter sans bruit, on les amplifie donc grâce à ce montage.

2. Amplificateur OPérationnel.
3. La version complète et normalisée est donnée en annexe A.
4. Pour l'analyse exhaustive du circuit, voir l'annexe B.

2.2 Parties à améliorer et cahier des charges

Avec le système déjà existant, mesurer le déphasage engendré par le ressort nécessitait la réalisation de deux manipulations.

La première consistait à remplacer le ressort par un fil rigide. Le résultat obtenu par acquisition des tensions rend alors compte de la phase du mouvement transmis initialement par le moteur.

La deuxième consistait à réaliser une acquisition toujours selon le mode opératoire décrit dans la partie précédente, en intercalant le ressort. Le signal obtenu rend donc compte de la phase du mouvement après ressort.

En superposant les deux signaux sous un logiciel d'analyse (ici Synchronie), il devient alors possible de mesurer la période du signal, ainsi que le décalage temporel entre les deux signaux.

Notre objectif à travers ce projet est donc la création d'un système permettant de réaliser l'acquisition de ces deux signaux en une seule manipulation. Il s'agit donc ici de réaliser une seule acquisition permettant une mesure de déphasage et donc la mesure de phase. La réalisation d'autres améliorations annexes facilitant cette mesure de phase nous a été permise et même conseillée.

Chapitre 3

Catalogue des dispositifs envisagés

Afin de mesurer le déphasage engendré par le ressort sur l'oscillateur mécanique, il paraissait évident que nous devions penser à un système à installer avant le ressort et à partir duquel il serait possible de réaliser une série de relevés de tensions électriques permettent de tracer le signal de la phase du mouvement avant ressort.

Les premières recherches du projet nous ont conduit à considérer deux propositions :
— utilisation d'un système bielle manivelle avec laser ;
— utilisation d'un système de capteur optique utilisant l'encodage binaire GRAY.

Puis, sous l'initiative de M. JOUEN, une troisième proposition consistant à rajouter un mobile avant le ressort, similaire au système déjà existant, a finalement été étudiée.

Dans ce chapitre nous détaillerons chaque procédé en nous attardant sur les avantages et inconvénients de chacun, conduisant finalement à la sélection d'une seule des trois propositions afin de satisfaire le cahier des charges.

3.1 Système bielle-manivelle

Pourquoi utiliser un tel système ?

L'oscillation du système mécanique étant provoqué par le couple délivré par un moteur, il nous est alors paru logique de réutiliser ce couple. En effet, le système bielle-manivelle permettrait alors de transformer le mouvement de rotation de l'excentrique (point B sur la figure 3.1) situé en sortie du moteur, en mouvement de translation, par l'intermédiaire de deux liaisons pivot aux points B et C de la figure 3.1.

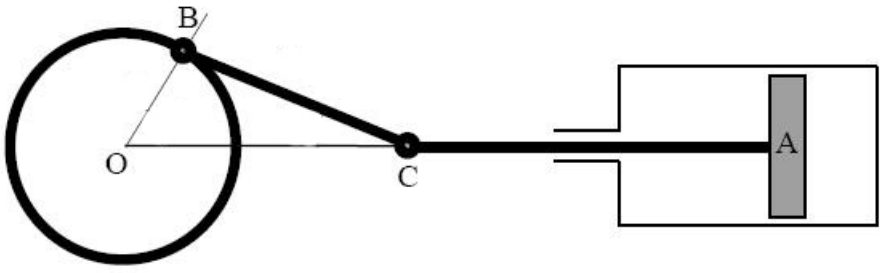


FIGURE 3.1 – Principe du système bielle-manivelle

Notons alors que ce système aurait la même fréquence que le mouvement engendré par le moteur. La partie translatant ferait par la suite office d'étude puisqu'elle serait donc représentative de la phase située avant ressort.

Comment effectuer les relevés de position du système au cours du temps ?

Bien que le système soit purement mécanique, l'ajout de système de lasers et capteur optique permettrait de remédier à ce problème. Un capteur serait alors situé sur la partie translatant du système (point A de la figure 3.1) et différents lasers émettant chacun une lumière de longueur d'onde différente seraient fixés à des emplacements précis, sur la partie immobile servant de logement pour la partie translatant. Le capteur serait donc mobile, suivant alors la périodicité du mouvement tandis que les lasers seraient immobiles. À chaque longueur d'onde relevée par le capteur correspondrait alors une position du capteur qui serait enregistrée par un logiciel configuré. Une courbe représentant la hauteur du mobile en fonction du temps pourrait alors être tracée.

Avantages

— Système visuel et pédagogique ;
— précision variable en fonction du nombre de lasers.

Inconvénients

— Prix élevé ;
— sensible à l'usure mécanique ;
— nécessite un entretien régulier ;
— encombrant ;
— nécessite une grande précision lors du calibrage ;
— difficile à mettre en place par nos soins.

3.2 Système à capteur optique et code GRAY

Un codeur absolu est un capteur de position angulaire qui est lié mécaniquement à un arbre qui l'entraîne, son axe fait tourner un disque qui lui est solidaire.

— Le disque comporte une succession de parties opaques et transparentes.
— Une lumière émise par des DEL traverse les fentes de ce disque créant sur les photodiodes réceptrices un signal analogique.
— Ce signal est amplifié électroniquement puis converti en signal carré, qui est alors transmis à un système de traitement.
— Chaque position élémentaire du mobile correspond à un code numérique, le codeur absolu délivre ainsi une valeur numérique codée en binaire correspondant à la position instantanée du mobile.

Obtention de la position

Le disque comporte n pistes, le système de lecture comporte donc n DEL et n photorécepteurs.

— La 1re piste possède une moitié opaque et une moitié transparente. On peut donc déterminer dans quel demi-tour on est ; c'est la piste MSB ou « bit de poids fort ».

— La piste suivante, en allant vers l'extérieur, est divisée en 4 quarts alternativement opaques et transparents. On détermine donc dans quel quart de tour on se situe.
— La piste suivante permet de savoir dans quel huitième de tour on est, la suivante dans quel seizième de tour, etc.
— La dernière piste, autrement dit la piste extérieure, est la piste LSB « bit de poids faible ». C'est elle qui donne la précision de la mesure, ou la résolution. Cette piste comporte 2^n zones noires ou blanches. Un tour de disque permet donc de coder 2^n positions.

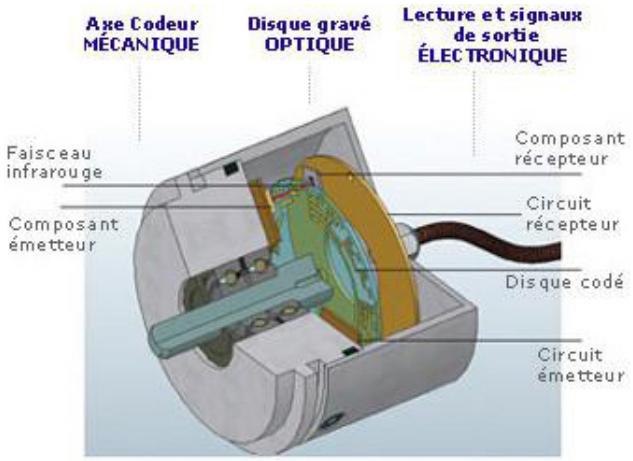


FIGURE 3.2 – Principe de fonctionnement d'un capteur optique (issue de [6])

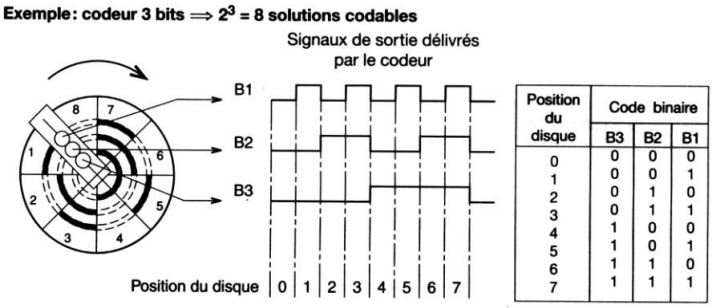


FIGURE 3.3 – Codeur à 3 bits (issue de [6])

Les différents codes délivrés

Le code binaire délivré par un codeur absolu peut être du binaire naturel ou du binaire réfléchi (code GRAY).

— Le binaire naturel est utilisable directement par l'unité de traitement mais il peut y avoir un changement simultané de plusieurs bits d'une position à la suivante si l'alignement des têtes de lecture n'est pas parfait. Il y aura apparition d'un code erroné.
— Le code GRAY ne fait intervenir qu'un changement de bit à la fois dans sa progression. Les codes erronés sont donc impossibles. Mais un transcodage du code GRAY au binaire naturel est nécessaire afin d'exploiter ce code.

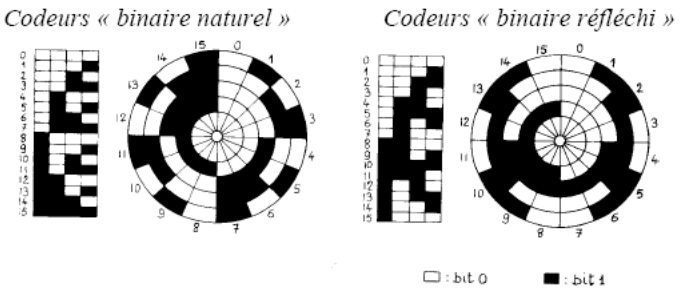


FIGURE 3.4 – Les deux différents codes (issue de [6])

Avantages

— Grande précision ;
— compacité (rentre dans le bloc moteur).

Inconvénients

— Prix élevé ;
— difficulté de mise en place : nécessite une grande précision pour le fixer à l'arbre moteur ;
— lecture non immédiate : nécessite une conversion d'un signal binaire en un signal continu (l'outil PWM [1] avait été envisagé).

3.3 Système à deux mobiles superposés

Cette méthode est identique à celle déjà présentée pour mesurer la phase du mobile après ressort ou mobile 2. Pour rappel, elle se constitue d'un mobile sur lequel est fixé une lame. La lame est constituée à ses deux extrémités d'une part, d'un fil électrique relié au circuit électrique central assurant la connexion du mobile avec le reste du circuit, et d'autre part d'un fil rigide dénudé trempant dans la solution électrolytique permettant les relevés de tension caractéristiques de la phase liée au mouvement et de la position du mobile. (cf. 2.1.2).

Seront alors suspendus au fil de haut en bas, le mobile 1 ou nouveau mobile, le ressort puis finalement le mobile 2.

Avantages

— Prix : matériaux à disposition dans les ateliers de l'INSA ;
— méthode opérationnelle car déjà mise en œuvre ;
— simple à expliquer et très visuel, dans un souci pédagogique.

Inconvénient

— Action de la masse du nouveau mobile sur le nylon importante.

Nota : des vues réalisées avec SolidWorks des deux derniers systèmes sont données en annexe C.

1. *Pulse Width Modulation*, ou Modulation de largeur d'impulsion.

Chapitre 4

Solution retenue et mise en œuvre

Après avoir défini et analysé les différentes propositions envisageables pour répondre à notre problème initial, nous avons opté dans un souci de temps, de complexité, de disponibilité mais surtout de budget, pour la mise en place d'un second mobile sur le montage mécanique.

Dans ce chapitre nous nous attellerons à la description précise et exhaustive de la solution retenue à l'unanimité, l'ajout d'un second mobile. Nous verrons comment nous avons conçu, fabriqué et intégré ce système, sur les plans mécanique, électronique et informatique.

4.1 Mise en place d'un deuxième mobile

4.1.1 Conception et problèmes à prendre en compte

Le nouveau mobile est sensiblement identique au mobile déjà existant. Toutefois, comme évoqué dans la partie précédente de ce rapport, ce premier doit avoir une faible masse pour assurer le bon fonctionnement du système tout en étant guidé en translation. Réduire la taille du mobile afin de l'alléger nous a paru la seule solution envisageable. Ceci impliquait alors l'impossibilité d'utiliser des poulies de guidage en translation comme pour le premier mobile puisque ces dernières n'étaient pas assez grandes et ajoutaient une masse supplémentaire.

Pour résoudre ce problème la solution présentée en figure 4.1 et figure C.3 de l'annexe C a été envisagée.

Afin de réaliser un guidage en translation n'incombant pas de masse supplémentaire sur le fil, l'utilisation de 2 tiges rigides en inox fixées sur le bâti a été proposée et approuvée par M. WILLIAMS. La lame du mobile 1 serait alors percée de deux trous dans lesquels passeraient les deux tiges en inox. Au cours de l'actionnement du mécanisme, le mouvement vertical de la lame serait alors contraint par les deux tiges, ce qui assurerait la translation rectiligne du mobile. Ces tiges en inox sont par ailleurs mobiles sur un plan horizontal, cela dans le but d'ajuster la position des tiges en fonction des trous de la lame dans lesquels elles doivent passer. Cette rotation est permise par un système simple de deux pièces en contact l'une avec l'autre par l'intermédiaire d'une liaison pivot.

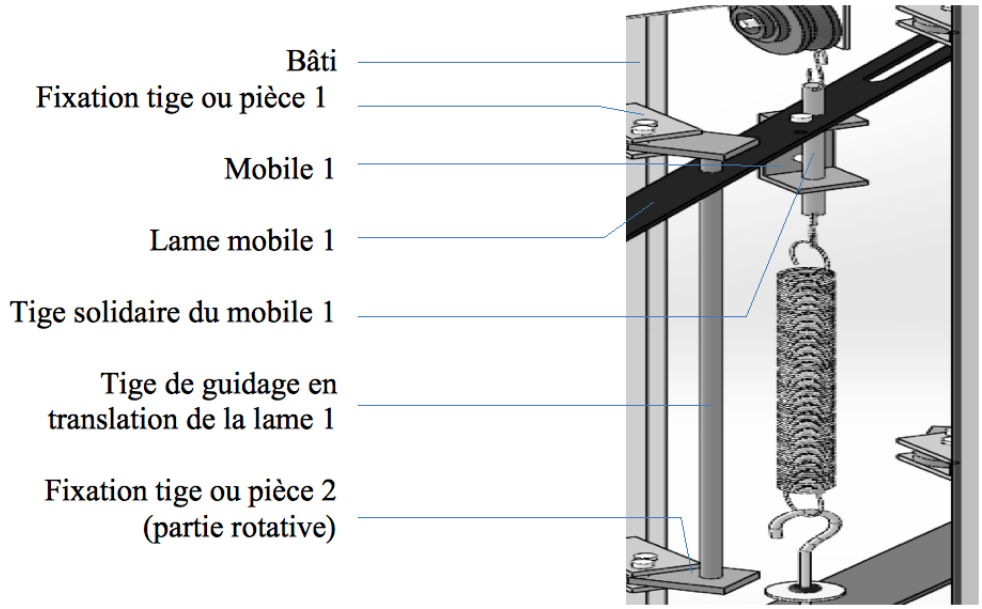


FIGURE 4.1 – Détail du système utilisé

Un deuxième problème s'est par la suite posé concernant le matériau dans lequel la lame du mobile est faite. Afin que le courant puisse passer entre ses deux extrémités, qui rappelons-le, sont reliées au circuit et à la solution électrolytique, cette lame est faite de cuivre soutenu par un support en époxy. Toutefois, lors du mouvement du mobile 1, les tiges en inox auraient alors pu être amenées, pour guider en translation la lame, à toucher le cuivre de la lame. L'inox étant un métal, il est donc conducteur et risquerait de provoquer un pont diviseur du courant traversant la lame. Bien que cela n'influe pas les résultats des mesures puisqu'elles sont en termes de tension, il nous a paru intéressant de traiter ce problème.

Dans le cas de la conception de notre première lame, le courant ne passait pas correctement dans la lame pour faute d'un défaut supposé dans le cuivre. M. WILLIAMS a donc proposé la réalisation d'un via, c'est à dire de relier les deux extrémités de la lame directement par un nouveau fil électrique.

Afin de résoudre ce problème mineur de division de l'intensité du courant tout en assurant le passage du courant électrique d'une extrémité à une autre, nous avons finalement proposé, dans le cadre de la généralisation du système, l'utilisation d'une lame plastique classique dont les deux extrémités seraient reliées par un via.

4.1.2 Fabrication des pièces

Par ailleurs, dès lors que la conception de ce nouveau système fut terminée, sa fabrication s'est réalisée en deux temps.

La plupart des pièces étant en métal, et devant être usinées, ces dernières ont été fabriquées par l'intermédiaire de l'atelier mécanique par les soins de MM. MOUARD et LESOBRE. Notez qu'initialement, nous souhaitions fabriquer ces pièces par le biais de l'imprimante 3D du laboratoire DUMONT D'URVILLE ou de celle du pôle mécanique afin qu'elles soient plus légères car en plastique. Elles auraient été toutefois trop fragiles.

Afin que la fabrication débute, nous avons pris contact avec M. Thomas BRETEAU à qui nous avons transmis les plans techniques des pièces voulant être usinées [1]. Dès lors certains problèmes pratiques se sont révélés – tel que celui de la non-correspondance des taraudages de vis (les vis M2 ont été remplacées par des M3 [2]) – qui ont été rapidement résolus.

Par ailleurs, la lame du mobile, elle, a été conçue au laboratoire DUMONT D'URVILLE par M. WILLIAMS ainsi que la soudure des fils sur celle-ci (plan donné en annexe E).

4.2 Acquisistion des tensions et analyses

4.2.1 Adaptation du circuit au montage à deux mobiles et Arduino

L'ensemble étant relativement peu esthétique et les élèves pouvant manipuler à loisir un circuit préparé par les techniciens, nous nous sommes proposé de miniaturiser le circuit.

Le choix initial du capteur GRAY ayant nécessité un micro-contrôleur Arduino, nous avons, sur conseil de M. JOUEN, modélisé le circuit grâce à la suite logicielle KiCad [3], dans le but d'obtenir un *shield* [4] comprenant les éléments électroniques greffable sur le Arduino.

La première étape consistait à faire des relevés sur le circuit – connexions, tensions, intensités, résistances – afin de le comprendre précisément et ainsi pouvoir faire la schématique en toute quiétude.

Le contrôleur Arduino délivrant une tension de 5V, il nous a fallu conserver le montage {pont diviseur + suiveur}. Les tests effectués avec une *breadboard* [5] ont montré que le Arduino était capable d'accepter en entrée des tensions de l'ordre de la centaine de millivolts, néanmoins nous avons décidé de conserver le montage amplificateur dans un but pédagogique, afin d'avoir des tensions plus explicites pour les étudiants, puisque l'amplitude de tension est ainsi augmentée.

Nous avons alors dû rajouter un montage amplificateur pour le mobile qui est placé avant le ressort. Le montage est identique à celui décrit plus haut et dans l'annexe B.

La première étape fût de réaliser la schématique du montage grâce au logiciel Eeschema de la suite KiCad [6]. Le schéma est donné en annexe F.

Après schématisation du circuit, il a fallu associer les composants à des empreintes réelles – i.e. des références industrielles – afin de prévoir le matériel à sortir de l'atelier ou à acheter. Nous avons pour cela utilisé le logiciel CvPCB de la suite KiCad.
Une *netlist* a ainsi été générée, contenant les références industrielles de tous nos composants. Nous la donnons en figure 4.2.
Notez qu'ici, les AOP ne sont plus alimentés en $\pm$10V mais en 5V / GND. En effet, le Arduino ne délivre pas d'alimentation symétrique. Ainsi, au lieu d'utiliser des TL082 comme sur les plaquettes pédagogiques, nous utilisons des LM358 [7].

1. Les plans sont donnés en annexe D.
2. De manière générale, une vis Mn est une vis de $n$mm.
3. Voir [7].
4. Circuit imprimé permettant l'extension du Arduino, enfichable sur celui-ci.
5. Plaque permettant de connecter des composants sans soudure, afin d'effectuer un prototypage.
6. Pour le tutoriel utilisé, voir [8].
7. Références industrielles des AOP.

```
 1       P1 -          Mobile1 : Connect:bornier2
 2       P2 -          Mobile2 : Connect:bornier2
 3       P3 -          Cathode : Connect:bornier2
 4       P4 -            Anode : Connect:bornier2
 5       R1 -              1kΩ : Discret:R3-LARGE_PADS
 6       R2 -              1kΩ : Discret:R3-LARGE_PADS
 7       R3 -              1kΩ : Discret:R3-LARGE_PADS
 8       R4 -             10kΩ : Discret:R3-LARGE_PADS
 9       R5 -             10kΩ : Discret:R3-LARGE_PADS
10       R6 -             10kΩ : Discret:R3-LARGE_PADS
11  SHIELD1 - ARDUINO_UNO_REV3 : Shield_Arduino:ARDUINO_SHIELD
12       U1 -            LM358 : Housings_DIP:DIP-8__300_ELL
13       U2 -            LM358 : Housings_DIP:DIP-8__300_ELL
```

FIGURE 4.2 – Netlist (capture d'écran CvPCB)

L'ouverture de cette netlist dans le logiciel Pcbnew nous a ensuite permis de tracer les pistes de cuivre sur le shield pour le Arduino, c'est-à-dire que nous avons réalisé le PCB [8] nécessaire à notre projet.

Nous avons ensuite envoyé le positif de ce PCB, comme donné en annexe G, à M. WILLIAMS, qui s'est occupé de sa réalisation – le processus requérant savoir-faire et produits toxiques, nous n'avons pu assister qu'à une explication sommaire du procédé par M. JOUEN.

Les intérêts de cette miniaturisation sont multiples. Tout d'abord, cela permet de rendre l'ensemble de travail plus compact, la plaquette pédagogique étant d'une taille imposante.

Ensuite, cela offre une consommation électrique réduite : toutes les tensions sont gérées par le Arduino, alimenté en 7V *via* sa prise USB. Le seul outil branché sur les prises secteur est donc l'ordinateur.

De plus, les éléments électriques précédemment utilisés – plaquette, AOP, résistances – redeviennent utilisables pour d'autres TP, et les techniciens n'auront plus à se soucier de câbler un circuit que les élèves pourraient s'amuser à inverser.

Enfin, cela réduit de manière drastique le coût global du TP. Certes l'INSA dispose déjà des plaquettes pédagogiques, l'élément le plus cher, mais si une de celles-ci venait à tomber hors-service, nous n'osons imaginer le prix du remplacement, mais il serait sans aucun doute plus élevé que le prix du Arduino Uno (20€ à la rédaction de ce rapport).

4.2.2 Récupération des tensions

Compte tenu du montage choisi pour mesurer la première acquisition (celle de la partie avant ressort), nous avons opté pour l'option de développer un logiciel adapté à notre problème.

Afin de tracer les courbes des deux acquisitions, il nous a été nécessaire d'utiliser un micro-contrôleur Arduino. Nous avons alors créé un programme en language Arduino (donné en annexe H), permettant de récupérer des tensions sur les ports analogiques `A1` et `A2`. Les tensions captées sont des nombres binaires compris entre 0 et 1023, et nous les avons donc convertis en tension comprises entre 0 et 5V pour faciliter la lecture des données aux étudiants.

8. *Printed Circuit Board*.

4.2.3 Logiciel de mesure du déphasage

Afin de concevoir un logiciel de mesure maintenable, il nous fallait trouver un language de programmation qui soit à la fois lisible, relativement pédagogique et performant. Le language Python nous a alors semblé répondre au maximum des critères que nous recherchions. Une autre raison pour laquelle nous avons choisi de programmer en Python est le fait que ce dernier, lors de son installation, contient un certain nombre de librairies natives, qu'il nous aurait fallu installer manuellement avec la plupart des autres languages de programmation. Python ne nécessitant pas de compilation, il nous a certainement fait gagner du temps tout au long de l'élaboration de ce programme.

Lors d'une acquisition, le micro-contrôleur Arduino renvoie des tensions comprises entre 0 et 5V, comme nous l'avons dit précédemment. Ainsi, il nous faut associer ces tensions à un temps pour pouvoir tracer une courbe d'acquisition.

Dans un premier onglet, on retrouve différents *widgets* nous permettant de lancer une nouvelle acquisition, de choisir le temps de la mesure, de quitter l'application...

La fonction analyse (donnée en annexe I, lignes 66 à 88) créé une communication entre le micro-contrôleur et le logiciel. Elle associe ensuite à chaque mesure le temps correspondant, ce qui permet ensuite de ressortir un graphique, tracé grâce à la librairie Matplotlib.

Le temps de l'acquisition est laissé au choix de l'utilisateur, entre 10 et 60 secondes.

Dans un second onglet, on retrouve le tableau des valeurs mesurées. L'utilisateur doit naviguer sur plusieurs pages, du fait du nombre important de mesures effectuées.

Enfin, l'utilisateur a également la possibilité d'exporter ces valeurs dans un fichier `.csv` (*Comma Separated Values*) qu'il pourra ensuite ouvrir avec un tableur et effectuer des calculs variés sur ces valeurs.

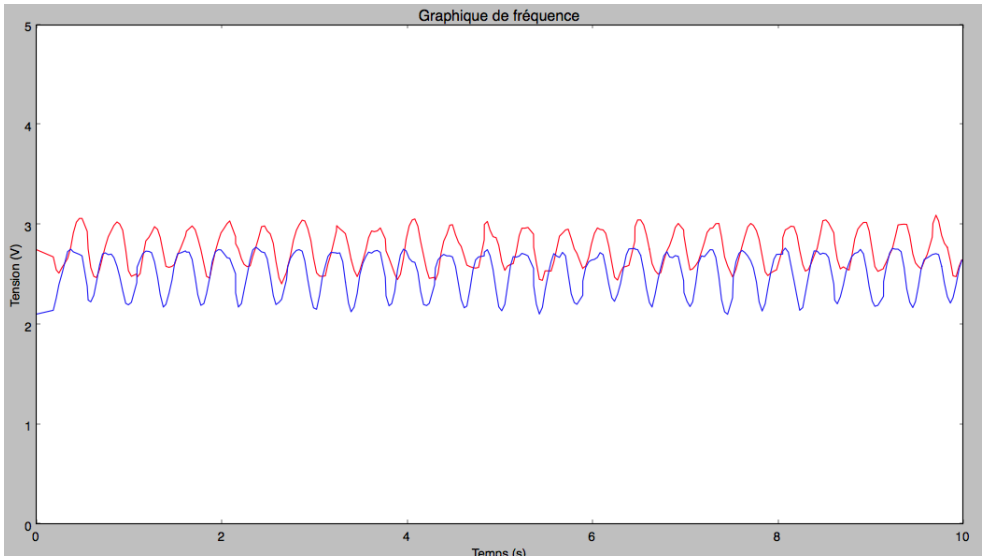


FIGURE 4.3 – Capture d'écran du logiciel avec les deux signaux

La figure 4.3 montre le résultat que nous obtenons dans notre logiciel d'analyse. L'acquisition a été faite sur 10s, avec le moteur à 2,52Hz (selon son écran intégré).

En bleu est tracée la courbe avant ressort, autrement dit la phase. La courbe rouge représente l'acquisition après ressort, c'est-à-dire le mobile déphasé.

Nous avons pu observer qu'au vu du montage – notamment un ressort particulièrement rigide – il a fallu monter à plus de 2Hz avant d'observer un déphasage, si faible soit-il.

Conclusion

Ce dossier en atteste, nous avons mené notre projet à bien. Les objectifs donnés par l'INSA, à savoir améliorer le dispositif afin de le rendre plus compact et mesurer la phase, ont été validés. Les dimensions temporelle et budgétaire ont elles aussi été respectées, puisque nous n'avons dépassé ni le temps ni le coût alloués.

Ce projet a largement fait appel à nos connaissances techniques et scientifiques, qu'elles soient d'ordre électrique, mécanique ou informatique, et ce tant sur la réalisation du projet que sur la résolution des problèmes auxquels nous avons fait face. Il nous aura aussi permis d'approfondir lesdites connaissances. Nous avons ainsi amélioré notre compréhension de la physique.

Cela a également été l'occasion pour nous de découvrir un vrai travail d'équipe. La répartition en groupes de 2 nous a permis d'avancer efficacement sur plusieurs fronts, et la cohésion de notre équipe a permis d'avancer vers un même horizon.

Se mettre en relation avec les personnes compétentes afin de nous procurer le matériel nécessaire, usiner ou obtenir des pièces conçues par nos soins a également été un point important de notre projet.

L'abandon de notre première solution, le code GRAY, au bout de presque 2 mois de travail, nous est apparu comme frustrant au premier abord. Cependant, le recul dont nous avons su faire preuve nous a montré que sans cette décision, nous n'aurions pas été confrontés à l'apprentissage de la résolution des problèmes techniques et budgétaires que nous sommes sûrs de rencontrer dans notre futur métier d'ingénieur.

Enfin, le fait que le sujet puisse être réellement mis en place et puisse servir pour les générations futures de STPI 1 nous a donné une impulsion nécessaire pour mener à terme de la meilleure des manières ce projet.

Pour conclure, nous pensons que les améliorations à apporter à ce projet résident essentiellement dans le placement du Arduino avec son shield dans le bloc moteur, avec une vitre transparente afin que les élèves puissent voir le circuit.

Bibliographie

Monographies

[1] BARTMANN Erik, *Le Grand livre d'Arduino*
Éditions Eyrolles, 2014

[2] LE GOFF Vincent, *Apprenez à programmer en Python*
Collection Le Livre du Zéro
Éditions Simple IT, 2011

[3] MARGOLIS Michael, *La boîte à outils Arduino*
Éditions Dunod, 2012

[4] MAYÉ Pierre, *Composants électroniques*, 5e édition
Collection Aide-Mémoire
Éditions Dunod, 2015

[5] TAVERNIER Christian, *Arduino – Maîtrisez sa programmation et ses cartes d'interface* (shields), 2e édition
Éditions Dunod, 2014

Sites internet

[6] Document de cours de la CPGE du lycée Pierre Paul RIQUET, 04 avril 2015
`http://pedagogie.ac-toulouse.fr/lyc-riquet-saint-orens/CPGESII/2-Acquerir%20l'information/Les%20capteurs/Les%20codeurs.pdf`

[7] Site officiel de KiCad, 4 mars 2015
`http://www.kicad-pcb.org`

[8] PIETTE Ferdinand, dossier *Créez vos PCB avec KiCad*, 13 mars 2015
`http://www.ferdinandpiette.com/blog/2012/04/un-nouveau-dossier-creer-vos-pcb-avec-kicad/`

[9] Question concernant l'amplification d'une tension continue sur le forum Arduino, 13 mars 2015
`http://forum.arduino.cc/index.php?topic=66236.0`

[10] Question concernant l'empreinte de composants sur le forum Futura-Sciences, 13 mars 2015
`http://forums.futura-sciences.com/electronique/635134-realisation-dune-carte-kicad.html`

[11] Question concernant un AOP LM324N sur le forum d'OpenClassroom, 13 mars 2015
`http://openclassrooms.com/forum/sujet/amplificateur-operationnel-lm324n`

[12] Documentation technique de l'AOP LM358 de Texas Instruments, 8 juin 2015
`http://www.ti.com/lit/ds/symlink/lm158-n.pdf`

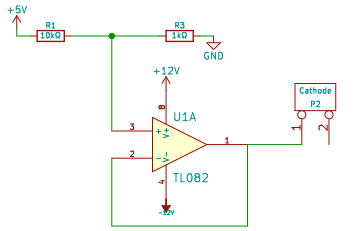
Annexes

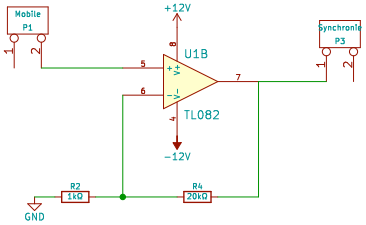
Table des annexes

Annexe A

Circuit originel

Voir page suivante.





Arthur LEBRÊNE / Baptiste ROUSSEAU
INSA de Rouen

Sheet: /
File: circuit arduino projet p6.sch

Title: Circuit originel

Size: A4	Date: 10 mar 2015	**Rev: 2**
KiCad E.D.A. kicad (2015-03-28 BZR 5547)-product		Id: 1/1

Annexe B

Analyse exhaustive du circuit électrique

Nota : Cette étude est basée sur le livre *Composants électroniques*, voir [4], ainsi que les cours d'électricité (P3) de STPI1.

Partant d'une alimentation de 5V et 0,45mA délivrée par la plaquette pédagogique [1], il faut diviser la tension par 10 afin de ne pas dégrader la solution électrolytique, ou du moins de la dégrader moins rapidement, puis stabiliser cette tension. Pour cela, on utilise un pont diviseur de tension comme montré sur le schéma B.1.

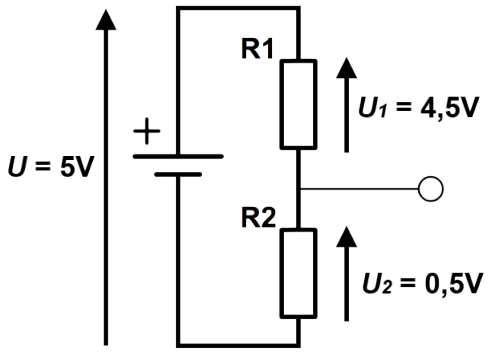


FIGURE B.1 – Pont diviseur de tension tel qu'utilisé

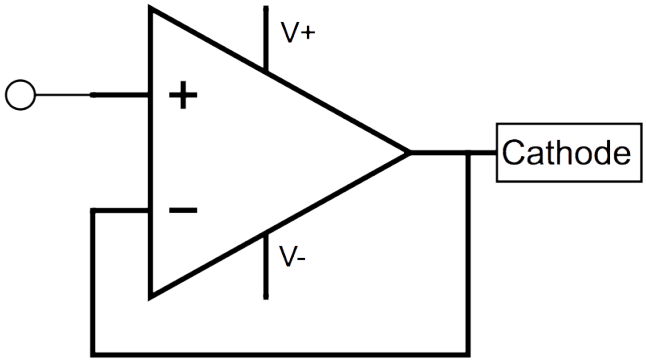


FIGURE B.2 – AOP en montage suiveur relié à la cathode

On a bien $U = U_1 + U_2$. Il nous reste à déterminer les valeurs des résistances R_1 et R_2. D'après la loi d'OHM $R = U/I$:

$$R_1 = \frac{U_1}{I} = \frac{4,5}{0,45 \times 10^{-3}} = \boxed{10\text{k}\Omega} \tag{B.1}$$

$$R_2 = \frac{U_2}{I} = \frac{0,5}{0,45 \times 10^{-3}} \simeq \boxed{1\text{k}\Omega} \tag{B.2}$$

Cette tension de 0,5V est ensuite stabilisée par un AOP alimenté en $\pm$10V et branché en montage suiveur comme sur le schéma B.2 – le point relié à la borne positive de l'AOP est relié au point du schéma B.1. La sortie de l'AOP est reliée à la cathode de la solution électrolytique, autrement dit la borne la plus haute dans l'éprouvette, tandis que l'anode est directement reliée à la masse du circuit.

1. Relevés effectués sur la plaquette.

Le mobile trempant dans la solution électrolytique, la tension peut varier de 0 à 0,5V, du moins en théorie. Les relevés montrent que la variation est moindre : pour le montage témoin dont nous disposions, la tension relevée était comprise entre 180 et 225mV.

Ces variations sont trop faibles pour que Synchronie puisse les détecter sans bruit. Il faut donc les amplifier. Pour ce faire, on emploie le montage présenté figure B.3.

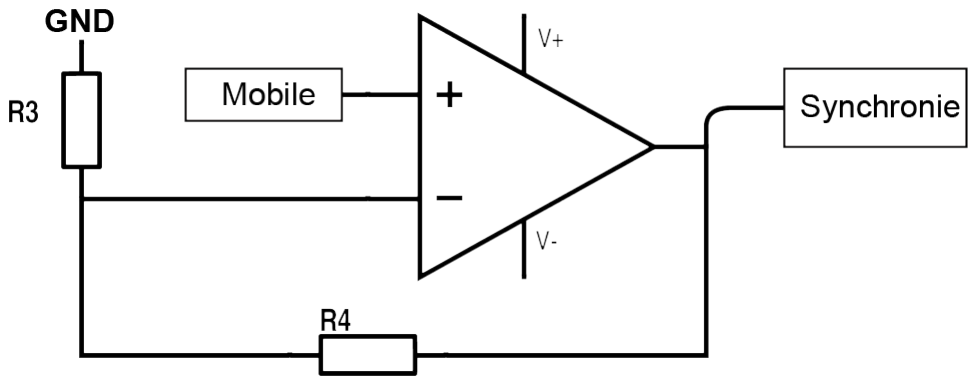


FIGURE B.3 – Montage amplificateur

On fixe $R_3 = 1\text{k}\Omega$. L'AOP étant alimenté en ± 10V, nous devons déterminer R_4 afin de ne pas le saturer, c'est-à-dire que la tension maximale de sortie ne dépassera pas la tension maximale d'alimentation.

Appelons V_e^+ (resp. V_e^-) la tension d'entrée à la borne + (resp. $-$) de l'AOP (non nulle), i_+ (resp. i_-) l'intensité à la borne + (resp. $-$), et V_s la tension de sortie. L'AOP étant supposé parfait, on a $V_e^+ = V_e^-$ et $i_+ = i_- = 0$. D'après le pont diviseur de tension, nous obtenons :

$$\frac{V_e^+}{R_3} = \frac{V_s}{R_3 + R_4} \tag{B.3}$$

$$\iff R_4 = \left(\frac{V_s}{V_e} - 1\right) R_3 \tag{B.4}$$

$$= \left(\frac{10}{0,5} - 1\right) \times 1000 \tag{B.5}$$

$$= \boxed{19000\Omega} \tag{B.6}$$

On prend alors une résistance de 20kΩ. Vérifions que cette valeur ne fait pas saturer l'AOP. D'après l'équation (B.3) :

$$V_s = \left(1 + \frac{R_4}{R_3}\right) V_e \tag{B.7}$$

$$= \left(1 + \frac{20000}{1000}\right) \times 0,5 \tag{B.8}$$

$$= \boxed{10,5\text{V}} \tag{B.9}$$

On dépasse légèrement la tension d'alimentation, cependant la probabilité que le fil relié au mobile délivre une tension de 0,5V, c'est-à-dire que ledit fil soit tout en haut de la solution, est quasi nulle. On peut donc prendre comme valeur de R_4 20kΩ.

Pour le circuit à deux mobiles cependant, nous n'utilisons plus des TL082, mais des LM358, lesquels sont alimentés en +5V / GND. Il faut donc refaire l'application numérique de (B.4) :

$$R_4^{'} = \left(\frac{5}{0,5} - 1\right) \times 1000 \tag{B.10}$$

$$= \boxed{9000\Omega} \tag{B.11}$$

Revérifions alors qu'on ne sature pas l'AOP par (B.7) :

$$V_s = \left(1 + \frac{10000}{1000}\right) \times 0,5 \tag{B.12}$$

$$= \boxed{5,5\text{V}} \tag{B.13}$$

Pour les mêmes raisons que précédemment, on peut effectivement prendre pour valeur de $R_4^{'}$ 10kΩ.

Rappelons les valeurs des résistances :
— R_1 = 10kΩ ;
— R_2 = 1kΩ ;
— R_3 = 1kΩ ;
— R_4 = 20kΩ ;
— $R_4^{'}$ = 10kΩ.

Annexe C

Vues en 3D des systèmes envisagés

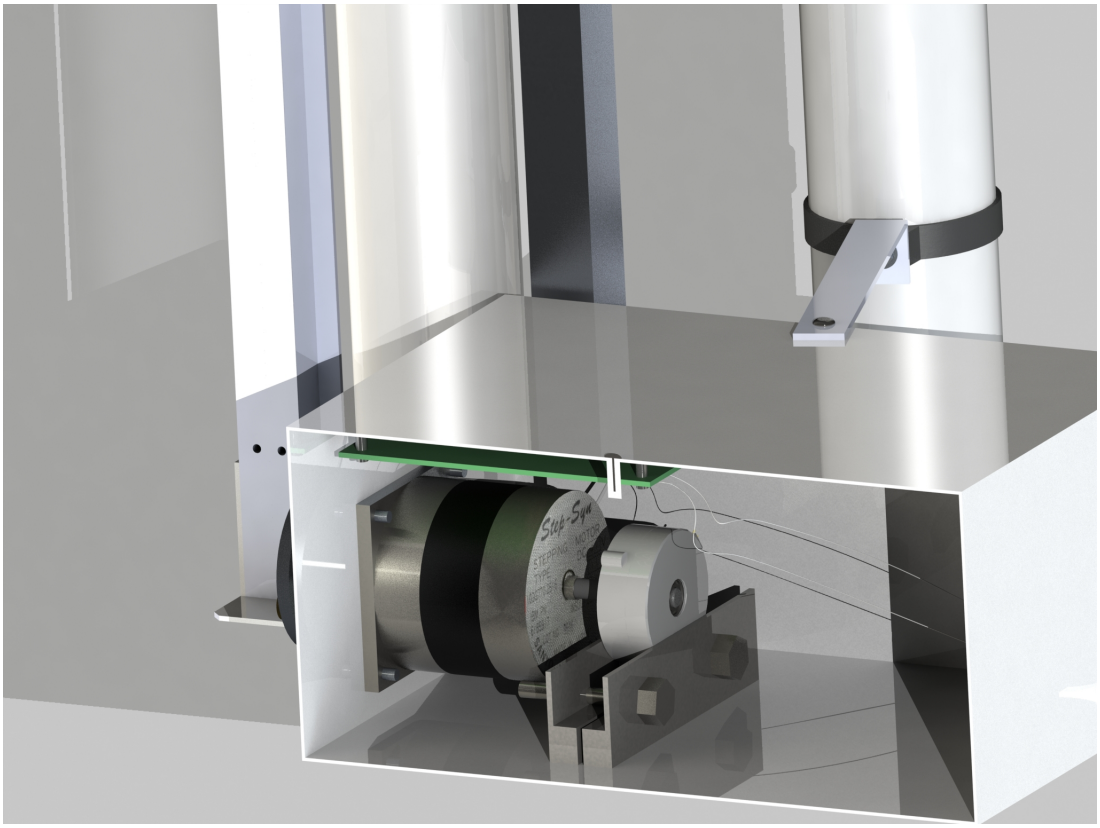


FIGURE C.1 – Système à capteur optique et code GRAY

FIGURE C.2 – Système à deux mobiles superposés

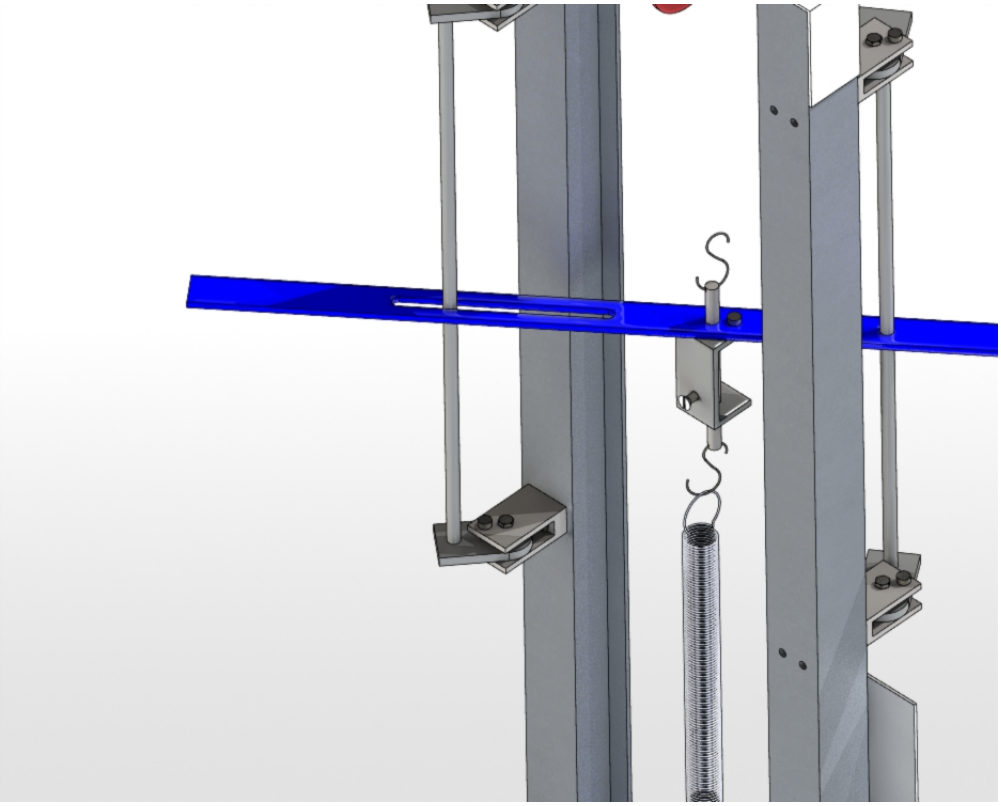
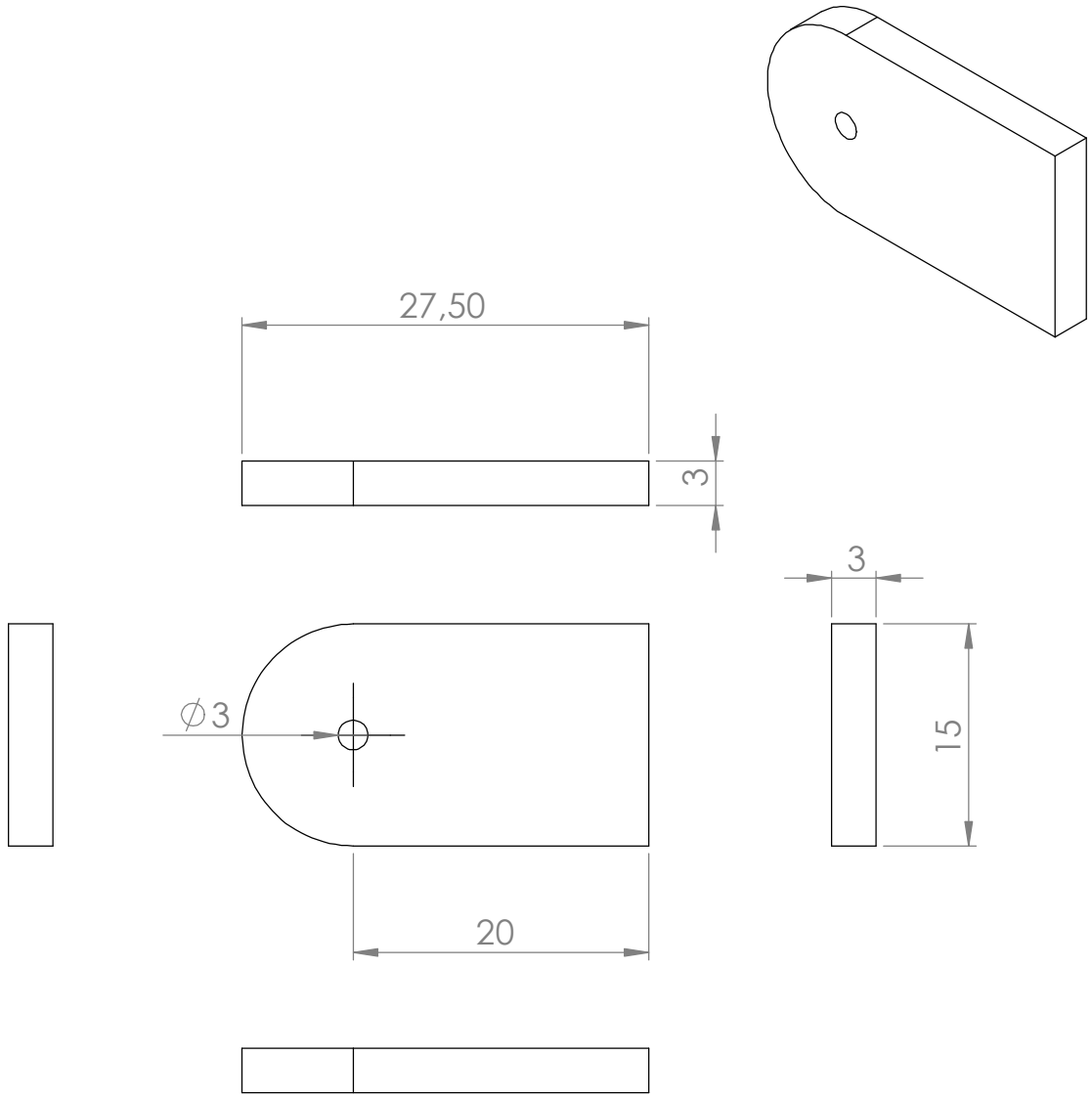


FIGURE C.3 – Gros plan sur le nouveau mobile

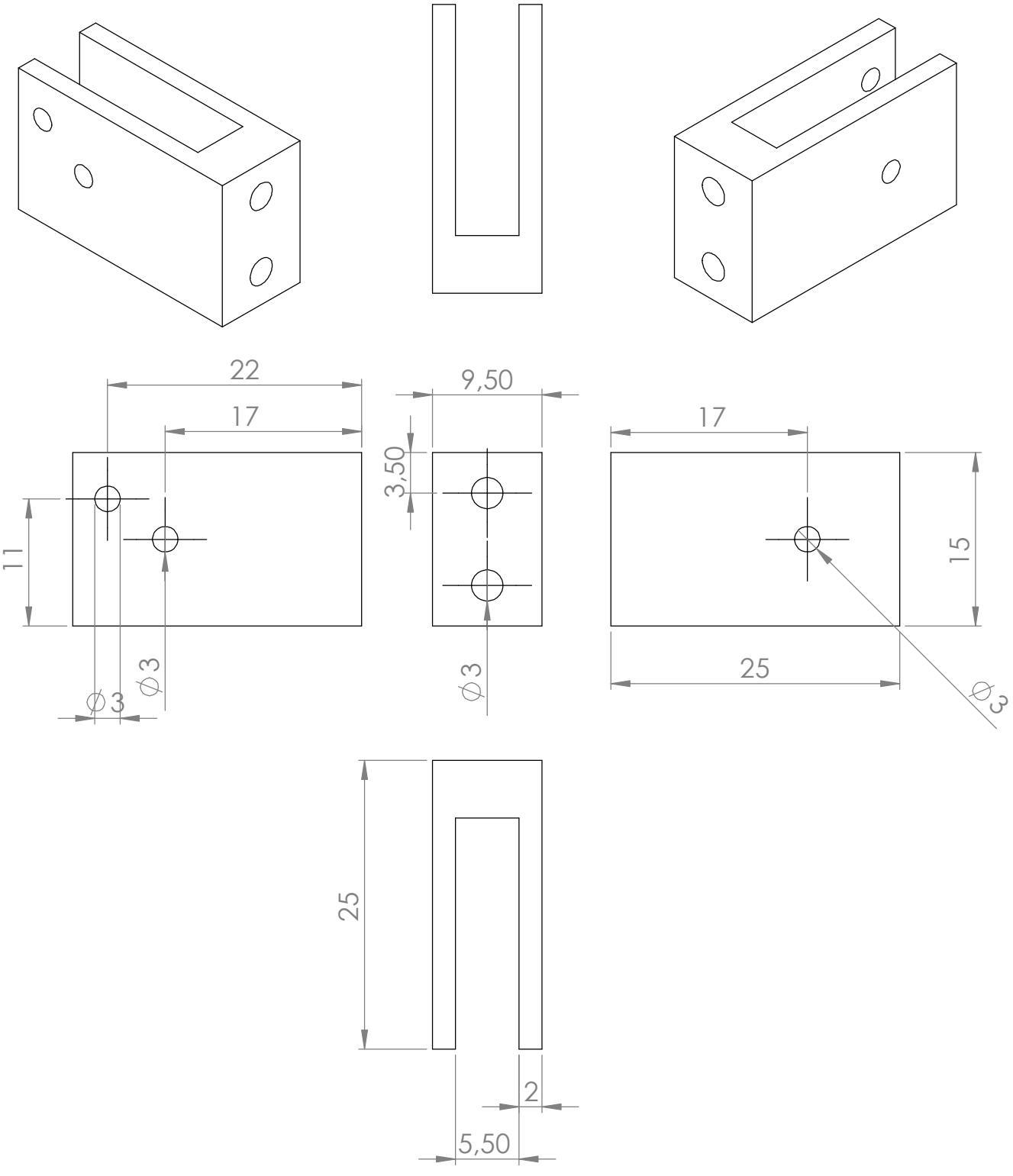
Annexe D

Plans des pièces

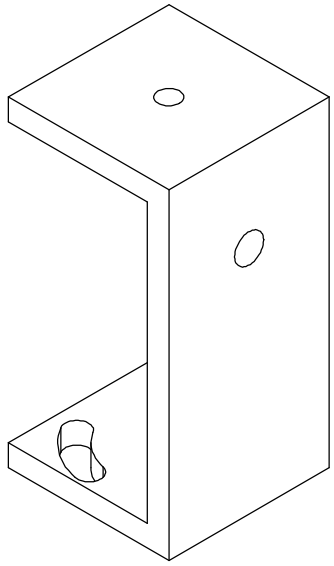
Voir les 5 pages suivantes.

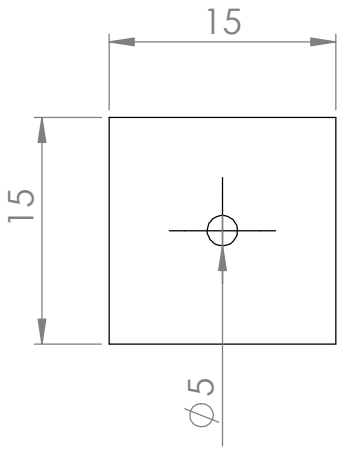


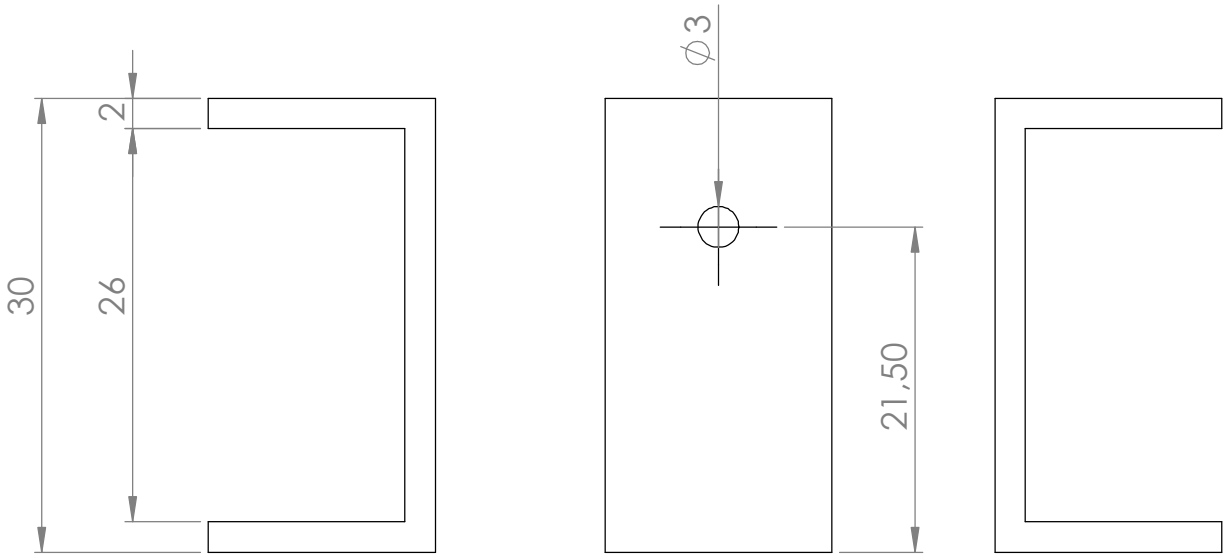
SAUF INDICATION CONTRAIRE: LES COTES SONT EN MILLIMETRES ETAT DE SURFACE: TOLERANCES: LINEAIRES: ANGULAIRES:	FINITION:		CASSER LES ANGLES VIFS	NE PAS CHANGER L'ECHELLE	REVISION	
	NOM	SIGNATURE	DATE	TITRE:		
AUTEUR	Wan/Ma/Rousseau		14/06/2015	Fixation entre la tige de translation et le montant (partie rotative)		
VERIF.						
APPR.						
FAB.						
QUAL.						
MATERIAU:				No. DE PLAN	Fixation 2	A4
ECHELLE:2:1				FEUILLE 1 SUR 1		

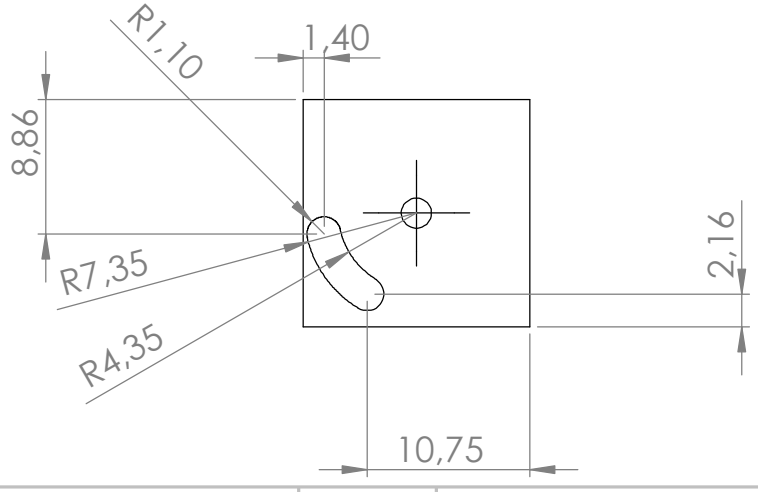


SAUF INDICATION CONTRAIRE: LES COTES SONT EN MILLIMETRES ETAT DE SURFACE: TOLERANCES: LINEAIRES: ANGULAIRES:	FINITION:		CASSER LES ANGLES VIFS	NE PAS CHANGER L'ECHELLE	REVISION	
	NOM	SIGNATURE	DATE	TITRE:		
AUTEUR	Wan/Ma/Rousseau		14/06/2015	Fixation entre la tige de translation et le montant (partie immobile)		
VERIF.						
APPR.						
FAB.						
QUAL.						
MATERIAU: nium				No. DE PLAN	fixemontant	A4
Licence étudiante de SolidWorks **Utilisation universitaire uniquement**				ECHELLE:2:1	FEUILLE 1 SUR 1	









SAUF INDICATION CONTRAIRE: LES COTES SONT EN MILLIMETRES ETAT DE SURFACE: TOLERANCES: LINEAIRES: ANGULAIRES:	FINITION:	CASSER LES ANGLES VIFS	NE PAS CHANGER L'ECHELLE / REVISION

	NOM	SIGNATURE	DATE
AUTEUR	Wan/Ma/Rousseau		14/06/2015
VERIF.			
APPR.			
FAB.			
QUAL.			

TITRE: Mobile 1

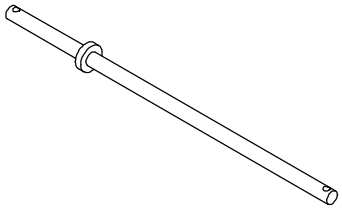
MATERIAU:

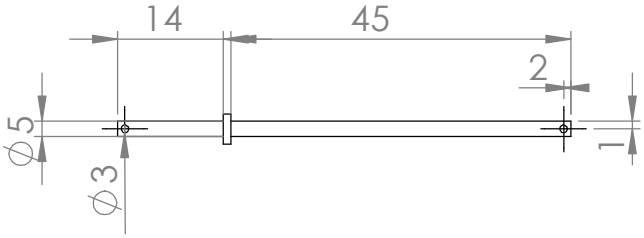
No. DE PLAN: mobile1

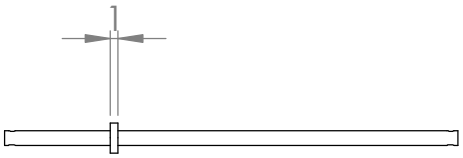
A4

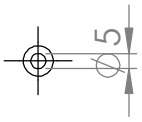
ECHELLE:2:1

FEUILLE 1 SUR 1

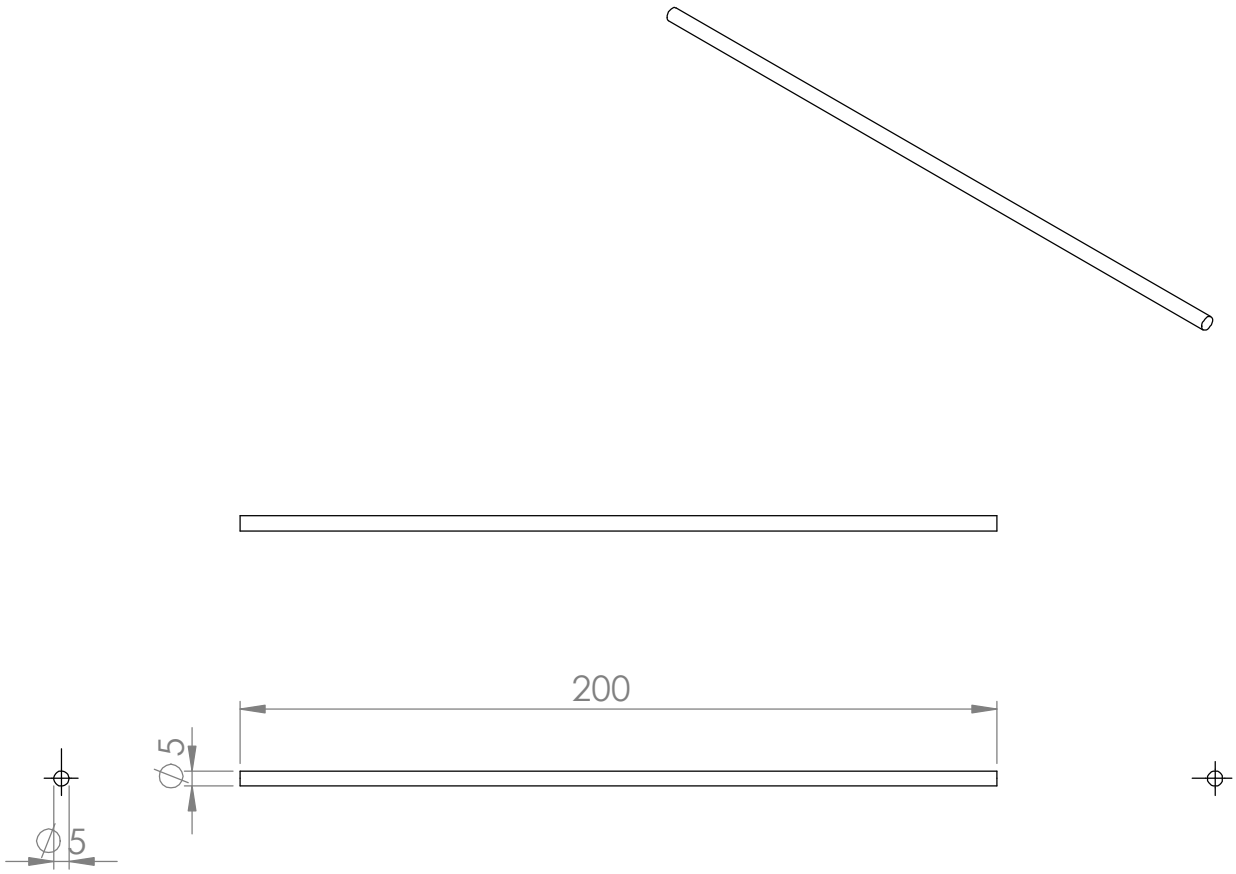








SAUF INDICATION CONTRAIRE: LES COTES SONT EN MILLIMETRES ETAT DE SURFACE: TOLERANCES: LINEAIRES: ANGULAIRES:	FINITION:				CASSER LES ANGLES VIFS	NE PAS CHANGER L'ECHELLE / REVISION
						Le bourrelet de 1 mm d'épaisseur n'est plus a usiner (la tige doit être lisse)
	NOM	SIGNATURE	DATE			TITRE:
AUTEUR	Wan/Ma/Rousseau		14/06/2015			Tige solidaire du mobile 1
VERIF.						
APPR.						
FAB.						
QUAL.				MATERIAU:		No. DE PLAN: tige mobile 1 — A4
Licence étudiante de SolidWorksium **Utilisation universitaire uniquement**						ECHELLE:1:1 — FEUILLE 1 SUR 1

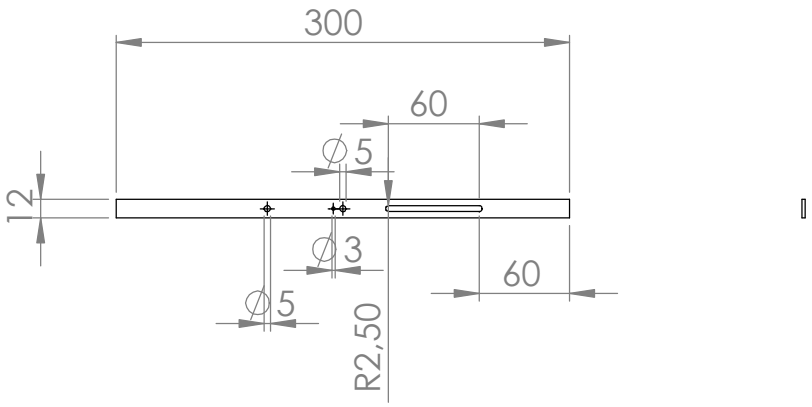


SAUF INDICATION CONTRAIRE: LES COTES SONT EN MILLIMETRES ETAT DE SURFACE: TOLERANCES: LINEAIRES: ANGULAIRES:	FINITION:		CASSER LES ANGLES VIFS	NE PAS CHANGER L'ECHELLE	REVISION	
	NOM	SIGNATURE	DATE	TITRE:		
AUTEUR	Wan/Ma/Rousseau		14/06/2015	Tige de guidage en translation pour la lame du mobile 1		
VERIF.						
APPR.						
FAB.						
QUAL.				MATERIAU:		
Licence étudiante de SolidWorks **Utilisation universitaire uniquement**				No. DE PLAN	SWBA60	A4
				ECHELLE:1:2	FEUILLE 1 SUR 1	

Annexe E

Plan de la lame du mobile

Voir page suivante.

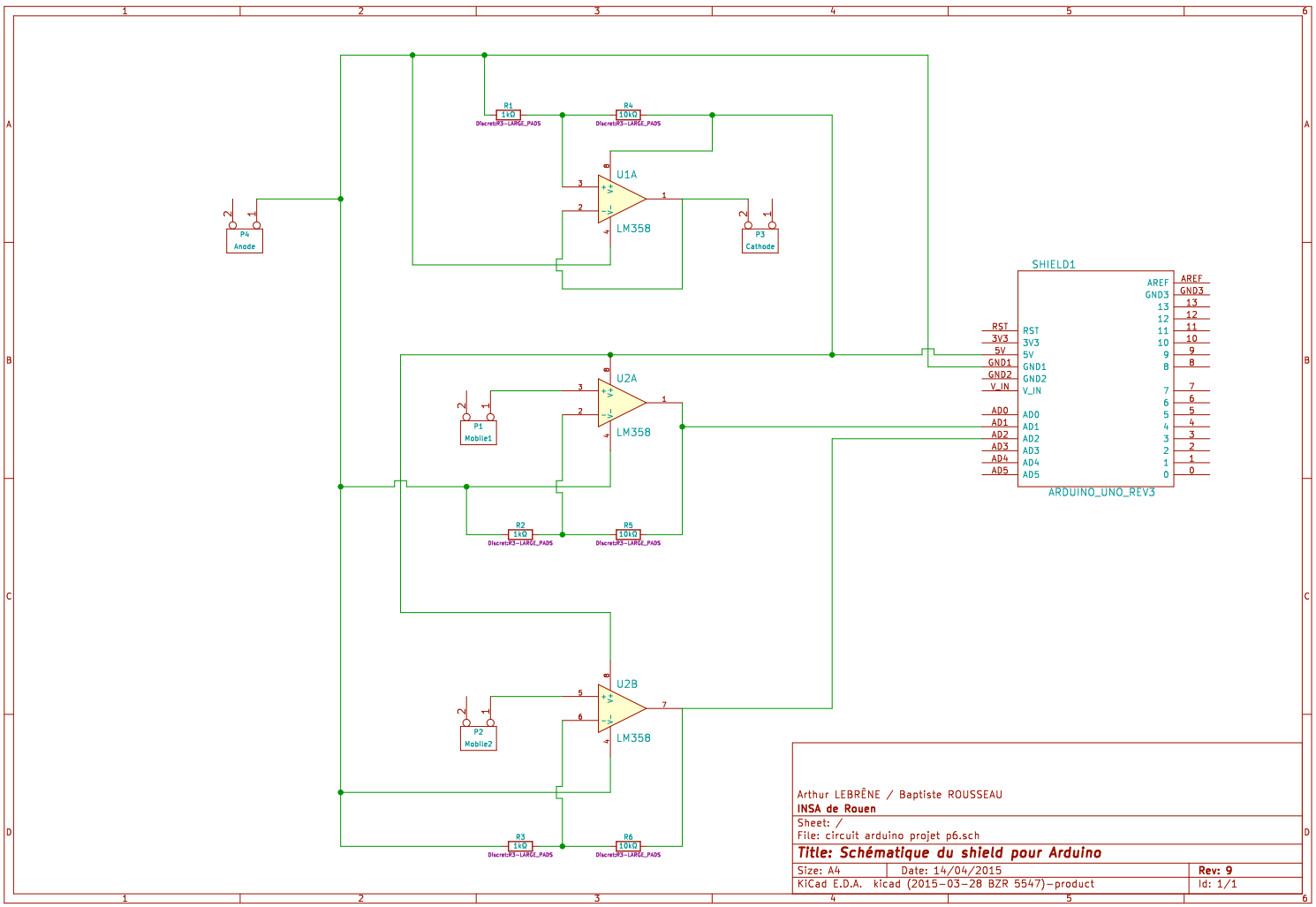


SAUF INDICATION CONTRAIRE: LES COTES SONT EN MILLIMETRES ETAT DE SURFACE: TOLERANCES: LINEAIRES: ANGULAIRES:	FINITION:		CASSER LES ANGLES VIFS	NE PAS CHANGER L'ECHELLE	REVISION	
	NOM	SIGNATURE	DATE	TITRE: Lame associée au mobile 1		
AUTEUR	Wan/Ma/Rousseau		14/06/2015			
VERIF.						
APPR.						
FAB.						
QUAL.				MATERIAU:		
Licence étudiante de SolidWorks **Utilisation universitaire uniquement**				No. DE PLAN lamemobile1		A4
				ECHELLE:1:5	FEUILLE 1 SUR 1	

Annexe F

Schématique du shield pour Arduino

Voir page suivante.



Arthur LEBRÊNE / Baptiste ROUSSEAU

INSA de Rouen

Sheet: /

File: circuit arduino projet p6.sch

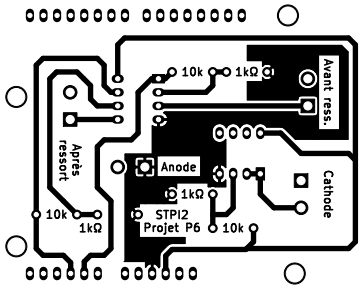
Title: Schématique du shield pour Arduino

Size: A4	Date: 14/04/2015	**Rev: 9**
KiCad E.D.A. kicad (2015-03-28 BZR 5547)-product		Id: 1/1

Annexe G

PCB du shield pour Arduino

Voir page suivante.



Échelle 1:1
Arthur LEBRÊNE / Baptiste ROUSSEAU
INSA de Rouen

Sheet:
File: circuit arduino projet p6.kicad_pcb

Title: PCB du shield pour Arduino

Size: A4	Date: 26/04/2015	**Rev: 5**
KiCad E.D.A. kicad (2015-03-28 BZR 5547)-product		Id: 1/1

Annexe H

Code Arduino

```
/*
  ReadAnalogVoltage
  Auteurs : Alexis et Gautier
  Lit les entrees analogiques des pins 1 et 2, les convertit en tension, et les affiche a l'ecran.
 */

// La routine setup s'execute quand on presse le bouton reset :
void setup() {
  Serial.begin(9600);// Initialise la communication a 9600 bits par seconde
}

void loop() {// Cette routine s'execute indefiniment
  int sensor1 = analogRead(A1);// Recupere la valeur en entree A1
  int sensor2 = analogRead(A2);// Recupere la valeur en entree A2
  // Convertit les valeurs recuperees (entre 0 et 1023) en tension (entre 0 et 5V) :
  float voltage1 = sensor1 * (5.0 / 1023.0);
  float voltage2 = sensor2 * (5.0 / 1023.0);
  // Affiche les tensions :
  Serial.print(voltage1);
  Serial.print(" ### ");// Separe les 2 tensions affichees sur une meme ligne
  Serial.println(voltage2);
}
```

Annexe I

Code du logiciel Python

```
1  # -*- coding: utf-8 -*-
2  """
3  Created on Mon Mar 23 09:04:32 2015
4
5  @author: Alexis
6  """
7
8  from time import time
9  import sys
10 from tkinter import *
11 from tkinter.ttk import *
12 from tkinter.filedialog import *
13 from glob import *
14 from serial import Serial, SerialException
15 from time import time
16 from matplotlib import pyplot as plt
17 from matplotlib.figure import Figure
18 from re import findall
19
20 class Arduino(object): #Cette classe definit l'objet Arduino
21     "Objet Arduino"
22     def serial_ports(self): #L'objectif de cette methode est de
   detecteur automatiquement le port USB sur lequelle l'Arduino est
    branche !
23         """Lists serial ports
24
25         :raises EnvironmentError:
26             On unsupported or unknown platforms
27         :returns:
28             A list of available serial ports
29         """
30
31         if sys.platform.startswith('win'):
32             ports = ['COM' + str(i + 1) for i in range(256)]
33
34         elif sys.platform.startswith('linux') or sys.platform.
   startswith('cygwin'):
35             ports = glob('/dev/tty[A-Za-z]*')
36
```

```
37          elif sys.platform.startswith('darwin'):
38              ports = glob('/dev/tty.usbmodem*')
39
40          else:
41              raise EnvironmentError('Unsupported platform')
42
43          result = []
44          for port in ports:
45              try:
46                  s = Serial(port)
47                  s.close()
48                  result.append(port)
49              except (OSError, SerialException):
50                  pass
51          return result
52
53      def autoDetectArduino(self):
54          return self.serial_ports()[0]
55
56      def __init__(self):
57          self.serialName = self.autoDetectArduino()
58
59  class Acquisition(object): #Cette classe correspond e l'acquisition
60      "Nouvelle acquisition"
61      def __init__(self, dureeAcquisition):
62          self.time, self.voltageRessort, self.voltageSansRessort =
        [], [], [] #3 tableaux contenant les differentes valeur d'
        acquisition
63          self.arduino = Arduino() #On cree un objet arduino
64          self.dureeAcquisition = dureeAcquisition #Duree d'
        acquisition recuperee via l'interface graphique (transmise en
        parametre lors de la creation d'une acquisition)
65
66      def analyse(self):
67          "Retourne un tuple contenant les x et les y de la courbe"
68          ser = Serial(self.arduino.serialName, 9600) #Serial est une
         fonction issue de la librairie Serial qui fait le lien entre
        Python et l'Arduino
69          graph = Courbe(self.dureeAcquisition) #Creation d'un objet
        courbe
70          for i in range(100): #On lis quelques valeurs dans le vide
        afin que l'acquisition demarre correctement, sinon il y a un
        leger decallage e t = 0
71              ser.readline()
72              i += 1
73          ts = time() #On recupere le timestamp avant de commencer l'
        acquisition dans la variable ts (Data de debut de l'acquisition)
74          graph.show() #On affiche le graphique (cf class Courbe)
75          graph.timer = ts #Cette variable permet de stocker la date
        de la derniere actualisation de la courbe. Ainsi on va pouvoir l
        'actualiser toutes les 0.5 secondes
76          while ((time() - ts) <= self.dureeAcquisition): #On rentre
        dans la boucle tant que la duree d'acquisition n'est pas ecoulee
```

```
77              try:
78                  graph.updateData(float(time() - ts), float(findall(
    "\d+.\d+", str(ser.readline()))[0]), float(findall("\d+.\d+",
    str(ser.readline()))[1])) #On actualise les donnees de notre
    graphique (cf classe Courbe); On doit changer les types de nos
    variables en float et on utilise pour les tensions des
    expressions regulieres car les valeurs envoyees dans le serial
    sont sous la forme (Tension1 ### Tension2)
79                  if ((time() - graph.timer) > 0.5): #Si 0.5 secondes
     sont ecoulees depuis la derniere actualisation du graphe on
    actualise
80                      graph.update()
81                      graph.timer = time() #La date de la derniere
    actualisation devient la datae actuelle
82              except IndexError:
83                  pass
84
85          graph.update() #On update l'affichage du graphe e la fin
86          self.time = graph.time  #On recupere les valeurs d'
    acquisition
87          self.voltageRessort = graph.voltageLisseRessort
88          self.voltageSansRessort = graph.voltageLisseSansRessort
89
90      def returnData(self):
91          return (self.time, self.voltageRessort, self.
    voltageSansRessort)
92
93
94  class Courbe(object):
95      "Nouvelle Courbe -- Nouvelle fenetre"
96      def __init__(self, dureeAcquisition):
97          self.time = [] #Tableau contenant les valeurs de temps;
    dates d'acquisitions
98          self.voltageRessort, self.voltageSansRessort, self.
    voltageLisseRessort, self.voltageLisseSansRessort = [], [], [],
    [] #Les 2 premiers tableaux contiennent les valeurs d'
    acquisition alors que les deux suivants contiennent les valeurs
    lisses e l'aide de la moyenne mobile (cf methode lisser)
99          self.fig = plt.figure() #On cree une nouvelle figure
100         self.fig.canvas.set_window_title("Acquisition") #On renomme
     la fenetre de la figure
101         self.graphZone = self.fig.add_subplot(111) #On ajoute un
    subplot
102         self.graphZone.set_title("Tension en fonction du temps") #
    Titre du graphique
103         self.graphZone.set_xlabel("Temps (s)") #Titre de l'axe des
    abscisses
104         self.graphZone.set_ylabel("Tension (V)") #Titre de l'axe
    des ordonnees
105         self.graphZone.axis([0,dureeAcquisition, 0, 5]) #On fixe
    les valeurs maximales et minimum du graphiques. Prend un tableau
     en parametres contenant [xmin, xmax, ymin, ymax]
```

```
106         self.manager = plt.get_current_fig_manager() #On recupere
    la figure courante dans le manager.
107         self.data1 = self.graphZone.plot(self.time, self.
    voltageLisseRessort, 'r', label = "Avec Ressort") #On trace les
    deux courbes. Lors de l'initialisation les tableaux sont vides
    donc cela ne trace en rien.
108         self.data2 = self.graphZone.plot(self.time, self.
    voltageLisseSansRessort, 'b', label = "Sans Ressort") #En
    revanche on va actualiser self.data1 et self.data2 e chaque fois
     que l'on veut reactualiser le graphe et ainsi afficher le
    graphe. Les labels correspondent e la legende
109         self.graphZone.legend(bbox_to_anchor=(1, 1), loc=0,
    borderaxespad=0.) #Place la legendre
110         self.timer = 0 #La variable permettant de savoir si on a
    actualise le graphique.
111
112     def lisser(self):
113         "Cette methode permet de recuperer les tableaux de donnees
    et de les lisser e l'aide d'une moyenne mobile"
114         #On parcourt le tableau non lisse depuis la derniere valeur
     lisse soit la taille du tableau contenant les valeurs lissees
    jusqu'e la taille du tableau de donnees. Ainsi on ne va lisser
    que les valeurs ajoutees depuis la derniere actualisation du
    graphique
115         # Principe de la moyenne mobile:
116         # Soit un signal 0 0 0 a b c  0 0 0
117         # Avec un coefficient de 1 sur la valeur centree et 1/2 sur
     les valeurs adjacentes on obtient:
118         # 0 0 (a/2/2) (a + b/2)/2 (b + (a/2) + (c/2))/2 (c + b/2)/2
     (c/2/2) 0 0
119         for i in range(len(self.voltageLisseRessort), len(self.
    voltageRessort)):
120             if (i == 0):  #Cas oe l'on est en debut de tableau, il
    n'y a donc pas de valeurs avant
121                 self.voltageLisseRessort.append((self.
    voltageRessort[i] + self.voltageRessort[i + 1])/2)
122                 self.voltageLisseSansRessort.append((self.
    voltageSansRessort[i] + self.voltageSansRessort[i + 1])/2)
123             elif (i == (len(self.voltageRessort) - 1)): #Cas oe l'
    on est en fin de tableau il n'y a donc pas de valeurs apres
124                 self.voltageLisseRessort.append((self.
    voltageRessort[i] + self.voltageRessort[i - 1])/2)
125                 self.voltageLisseSansRessort.append((self.
    voltageSansRessort[i] + self.voltageSansRessort[i - 1])/2)
126             else: #Cas classique
127                 self.voltageLisseRessort.append(((self.
    voltageRessort[i - 1]/2) + self.voltageRessort[i] + (self.
    voltageRessort[i + 1]/2))/2)
128                 self.voltageLisseSansRessort.append(((self.
    voltageSansRessort[i - 1]/2) + self.voltageSansRessort[i] + (
    self.voltageSansRessort[i + 1]/2))/2)
129
130     def show(self):
```

```
131         "Affiche le graphique"
132         self.fig.show()
133 
134     def update(self):
135         "Actualise le graphique"
136         self.lisser() #D'abord on lisse les valeurs bruts.
137         self.data1[0].set_data(self.time, self.voltageLisseRessort)
     #Puis on change les donnees du graphique
138         self.data2[0].set_data(self.time, self.
    voltageLisseSansRessort)
139         self.manager.canvas.draw() #On affiche ensuite.
140 
141     def addTime(self, time):
142         """Ajoute une valeur dans le tableau contenant les dates d'
    acquisition"""
143         self.time.append(time)
144 
145     def addVoltage(self, voltageRessort, voltageSansRessort):
146         "Ajoute les valeurs mesurees dans les tableaux
    voltageRessort et voltageSansRessort"
147         self.voltageRessort.append(voltageRessort)
148         self.voltageSansRessort.append(voltageSansRessort)
149 
150     def updateData(self, time, voltageRessort, voltageSansRessort):
151         "On appelle cette fonction pour actualiser les donnees
    brutes de nos tableaux"
152         #print(time, voltageRessort, voltageSansRessort) #Affichage
     dans la console. Utile pour debugger
153         self.addVoltage(voltageRessort, voltageSansRessort) #On
    ajoute les valeurs de tension
154         self.addTime(time) #On ajoute les dates d'acquisition
155         self.time[len(self.time) - 1] = self.time[len(self.time) -
    1] - self.time[0] #On fait un decallage de self.time[0]
    correspodnant e la premiere date d'acquisition car il y a un
    petit temps entre la determination de la date de debut d'
    acquisition et l'acquisition de la premiere valeur. On s'assure
    ainsi que la premiere valeur est prise e la date t = 0
156 
157 class TableauDeDonnees(object): #Cree un tableau de donnees dans un
     widget parent. Ici on va creer deux tableaux dans les deux
    onglets correspondants.
158     "Tableau de donnees -- Onglet Tableau de donnees"
159     def __init__(self, parent):
160         self.parent = parent #On initialise no variables de notre
    objet Tableau de donnees
161         self.time, self.voltage = [0], [0]
162         self.labelsNum, self.labelsTime, self.labelsVoltage = [],
    [], []
163         self.navigationFrame = Frame(self.parent) #On cree une
    Frame contenue dans notre parent
164         self.canvas = Canvas(self.parent, width = 250, height =
    500) #On place un Canvas dans cette Frame afin de pouvoir
    scroller. Car on ne peut pas scroller une Frame
```

```
165         self.mainFrame = Frame(self.canvas) #On cree ensuite une
    nouvelle Frame contenue dans le canvas
166         self.navigationFrame.pack() #On affiche la premiere frame
167         self.labelTime = Label(self.mainFrame, text = "Temps (s)",
    background = "white", foreground = "black",borderwidth = 3,
    relief = RIDGE) #On cree les differents labels de notre tableaux
    . Ici les noms de nos colonnes
168         self.labelTime.grid(column = 1, row = 0, sticky = NSEW) #.
    grid permet de placer les elements d'affichage sous la forme d'
    element dans un tableau le oe .pack() place l'element selon les
    widgets parents et adjacents automatiquement
169         self.labelVoltage = Label(self.mainFrame, text = "Tension (
    V)", background = "white", foreground = "black",borderwidth = 3,
     relief = RIDGE)
170         self.labelVoltage.grid(column = 2, row  = 0, columnspan =
    5, sticky = NSEW)
171
172
173         #On cree les boutons permettant de naviguer entre les
    onglets
174         self.boutonRetour = Button(self.navigationFrame, text = "
    Retour", command = lambda: self.move(-2))
175         self.quickNav = Entry(self.navigationFrame, width = 3)
176         self.quickNav.bind('<Return>', lambda x: self.move(self.
    quickNav.get()))
177         self.boutonSuivant = Button(self.navigationFrame, text = "
    Suivant", command = lambda: self.move(0))
178
179         #On fixe au depart le nombre de pages e 0
180         self.pages = 0
181         self.position = Label(self.navigationFrame, text='Page 0
    sur ' + str(self.pages)) #Ce label permet d'afficher oe l'on est
     situe dans le tableau de donnees
182         self.pageCourante = 0
183
184         #On place les boutons via la methode grid
185         self.boutonRetour.grid(row = 0, column = 0)
186         self.quickNav.grid(row = 0, column = 1)
187         self.boutonSuivant.grid(row = 0, column = 2)
188         self.position.grid(row = 0, column = 3)
189
190         #On initialise la scrollbar
191         self.scroll = Scrollbar(self.parent, orient = VERTICAL,
    command = self.canvas.yview)
192         self.canvas.configure(yscrollcommand = self.scroll.set)
193         self.scroll.pack(side = RIGHT, fill='y')
194         self.canvas.pack(side = LEFT, fill = 'both')
195         self.canvas.create_window((4,4), window = self.mainFrame,
    anchor = "nw", tags = "frame")
196         self.canvas.configure(yscrollcommand = self.scroll.set)
197         self.mainFrame.bind("<Configure>", self.update)
198         self.canvas.configure(scrollregion = self.canvas.bbox("all"
    ))
```

```
199
200     def update(self, event):
201         self.canvas.configure(scrollregion = self.canvas.bbox("all"
        ))
202
203     def rafraichir(self, data):
204         """
205         Permet d'actualiser le tableau de donnees lorsque l'on
        renouvelle l'acquisition.
206         Consiste en fait e l'affichage de la premiere page du
        tableau de donnees
207         """
208         self.time, self.voltage = data[0], data[1]
209         self.labelsTime, self.labelsVoltage = [], [] #Contient les
        labels de notre tableau. Chaque label correspond e une valeur d'
        acquisition
210         for row in range(100): #100 correspond au nombre d'elements
         affiches par page
211             self.labelsNum.append(Label(self.mainFrame, text = str(
        row), background = "white", foreground = "black",borderwidth =
        3, relief = RIDGE))
212             self.labelsNum[row].grid(column = 0, row = row + 1,
        sticky = NSEW)
213             self.labelsTime.append(Label(self.mainFrame, text = "
        {:10.4f}".format(self.time[row]), background = "white",
        foreground = "black",borderwidth = 3, relief = RIDGE))
214             self.labelsTime[row].grid(column = 1, row = row + 1,
        sticky = NSEW)
215             self.labelsVoltage.append(Label(self.mainFrame, text =
        "{:10.4f}".format(self.voltage[row]), background = "white",
        foreground = "black", borderwidth = 3, relief = RIDGE))
216             self.labelsVoltage[row].grid(column = 2, row = row + 1,
         sticky = NSEW, columnspan = 5)
217
218         temp  = divmod(len(self.time), len(self.labelsTime))
219         self.pages = temp[0] + (1 if temp[1] else 0) #Permet de
        recuperer le nombre de pages necessaires e l'affichage des
        donnees
220         self.position.config(text='Page 1 sur ' + str(self.pages))
221         self.pageCourante = 1
222
223
224
225     def move(self, direction):
226         """
227         Permet de se changer de page dans le tableau de donnees. En
         fonction de la page dans laquelle on est l'on affiche pas les
        memes valeurs:
228         1-99 sur la premiere page; 99-200 sur la deuxieme etc...
229         """
230         if (self.pageCourante == 1 and direction == -2) or (self.
        pageCourante == self.pages and direction == 0): #La valeur du
        parametre suivant est determinee selon que le bouton suivant ou
```

```
    retour soit clique
231             return
232         if direction in (-2, 0):
233             self.pageCourante += direction + 1
234         else:
235             try:
236                 temp = int(direction)
237                 if temp not in range(1, self.pages + 1):
238                     return
239             except ValueError:
240                 return
241             else:
242                 self.pageCourante = temp
243         for row in range(len(self.labelsTime)):
244             try:
245                 locationTime = "{:10.4f}".format(self.time[len(self
    .labelsTime)*(self.pageCourante - 1) + row]) #On stocke les
    nouvelles valeurs du tableau e afficher
246                 locationVoltage = "{:10.4f}".format(self.voltage[
    len(self.labelsVoltage)*(self.pageCourante - 1) + row])
247             except IndexError:
248                 location = ''
249             self.labelsTime[row].config(text = locationTime) #Puis
    on modifie le texte des labels avec les nouvelles valeurs.
250             self.labelsVoltage[row].config(text = locationVoltage)
251         self.position.config(text = 'Page ' + str(self.pageCourante
    ) + ' sur ' + str(self.pages)) #Enfin, on affiche sur quelle
    page l'on se trouve
252
253
254 class Application(object):
255     "Application principale"
256     def __init__(self, parent):
257         "Mise en place de l'affichage du programme"
258         self.parent = parent #Correspond e la fenetre
259         self.parent.title("Acquisition") #On change le titre de la
    fenetre
260         self.parent.resizable(width = FALSE, height = FALSE) #On
    enpeche l'utilisateur de modifier la taille de la fenetre.
    Passer les arguments e TRUE pour autoriser
261         self.onglets = Notebook(self.parent) #On cree un notebook.
    Correspond e la mise en place des onglets
262         self.premierOnglet = Frame(self.onglets) #On cree un
    premier onglet (l'onglet general)
263         self.frameGeneral = Frame(self.premierOnglet) #On cree une
    frame dans le premier anglais
264         self.frameGeneral.pack() #On pack le Frame general
265         self.frameAcquisitionRessort = Frame(self.onglets) #On cree
     l'onglet contenant le tableau de donnees de l'acquisition avec
    Ressort
266         self.frameAcquisitionSansRessort = Frame(self.onglets) #On
    cree l'onglet contenant le tableau de donnees de l'acquisition
    sans ressort
```

```
267
268         #On affiche les onglets
269         self.onglets.add(self.premierOnglet, text = "General")
270         self.onglets.add(self.frameAcquisitionRessort, text = "
    Acquisition avec Ressort")
271         self.onglets.add(self.frameAcquisitionSansRessort, text = "
    Acquisition sans Ressort")
272
273         #Creation du bouton temps d'acquisition
274         self.frameGeneral.labelTemps = Label(self.frameGeneral,
    text = "Temps d'acquisition: ")
275         self.frameGeneral.labelTemps.grid(row = 0, column = 0,
    sticky = NSEW)
276         self.frameGeneral.spinBoxDureeAcquisition = Spinbox(self.
    frameGeneral, from_ = 10, to = 60, increment = 5, width = 3,
    justify = CENTER)
277         self.frameGeneral.spinBoxDureeAcquisition.grid(column = 1,
    row = 0, sticky = NSEW, padx = 10)
278         self.frameGeneral.labelSecondes = Label(self.frameGeneral,
    text = "secondes")
279         self.frameGeneral.labelSecondes.grid(row = 0, column = 3,
    sticky = NSEW, padx = 10)
280
281         #Creation des differents boutons
282         self.frameGeneral.boutonAnalyse = Button(self.frameGeneral,
     text = "Nouvelle analyse", command = self.acquisition)
283         self.frameGeneral.boutonAnalyse.grid(row = 1, columnspan =
    4, sticky = NSEW)
284         self.frameGeneral.boutonExporter = Button(self.frameGeneral
    , text = "Exporter les donnees au format .csv (Excel)", command
    = self.exporter)
285         self.frameGeneral.boutonExporter.grid(row = 2, columnspan =
     4, sticky = NSEW)
286         self.frameGeneral.boutonQuitter = Button(self.frameGeneral,
     text = "Quitter", command = self.destroy)
287         self.frameGeneral.boutonQuitter.grid(row = 3, columnspan =
    4, sticky = NSEW)
288
289         self.frameGeneral.amplitudeRessortNom = Label(self.
    frameGeneral, text = "Amplitude avec ressort = ")
290         self.frameGeneral.amplitudeSansRessortNom = Label(self.
    frameGeneral, text = "Amplitude sans ressort = ")
291         self.frameGeneral.amplitudeRessortNom.grid(row = 4, column
    = 0, columnspan = 3, sticky = NSEW)
292         self.frameGeneral.amplitudeSansRessortNom.grid(row = 5,
    column = 0, columnspan = 3, sticky = NSEW)
293
294         #On grid les onglets
295         self.onglets.grid()
296
297         #On cree une Frame dans chaque autre onglet dans lequel
    sera contenu le tableau de donnees
298
```

```
299         self.navigationFrameRessort = Frame(self.
    frameAcquisitionRessort)
300         self.navigationFrameRessort.pack()
301         self.tabRessort = TableauDeDonnees(self.
    navigationFrameRessort)
302 
303         self.navigationFrameSansRessort = Frame(self.
    frameAcquisitionSansRessort)
304         self.navigationFrameSansRessort.pack()
305         self.tabSansRessort = TableauDeDonnees(self.
    navigationFrameSansRessort)
306 
307 
308     def exporter(self):
309         "Fonction gerant l'export au format CSV"
310         directory = askdirectory(initialdir = "/",title = "
    Choisissez le dossier dans lequel vous voulez exporter les
    donnees")
311         try:
312             exportedFile = open(directory + "/data.csv", 'w')
313             exportedFile.write("Temps(s); Tension sans ressort(V);
    Tension avec ressort\n")
314             for i in range(len(self.data[0])):
315                 #On formate nos donnees qu'on ecrit dans un fichier
     .csv. On doit remplacer les points par des virgules afin que
    nos donnees soit considerees comme des nombres lors de la
    lecture du fichier
316                 exportedFile.write(str(self.data[0][i]).replace('.'
    ,',') + ";" + str(self.data[1][i]).replace('.',',') + ";" + str(
    self.data[2][i]).replace('.',',') + "\n")
317         except:
318             pass
319         exportedFile.close()
320 
321 
322     def destroy(self):
323         "Quitte l'application"
324         self.parent.quit()
325 
326     def acquisition(self):
327         acquisition = Acquisition(int(self.frameGeneral.
    spinBoxDureeAcquisition.get())) #Lance une acquisition
328         acquisition.analyse() #Effectue une analyse
329         self.data = acquisition.returnData() #recupere les donnees
    de l'analayse pour les afficher dans le tableau
330         self.tabRessort.rafraichir((self.data[0], self.data[1])) #
    Mise e jour du tableau avec ressort
331         self.tabSansRessort.rafraichir((self.data[0], self.data[2])
    ) #Mise e jour du tableau sans ressort
332         self.frameGeneral.amplitudeRessortNom.config(text = "
    Amplitude avec ressort = " + "{:10.4f}".format(max(self.data[1])
     - min(self.data[1])))
```

```
333             self.frameGeneral.amplitudeSansRessortNom.config(text = "
    Amplitude sans ressort = " + "{:10.4f}".format(max(self.data[2])
     - min(self.data[2])))
334
335
336 if __name__ == '__main__':
337     root = Tk()
338     main = Application(root)
339     root.mainloop()
```

Protocole d'installation à l'attention des techniciens
14 juin 2015

---

1 Contenu du CD fourni par le groupe

Veillez à ce que le support qui vous a été remis par l'équipe du projet #12 contienne les éléments suivants :

— Le protocole d'installation (ce fichier) ;
— la notice d'utilisation à l'attention des élèves ;
— le rapport de projet « Rapport_P6_2015_12.pdf » ;
— le dossier KiCad « circuit arduino projet p6 » ;
— le code Arduino `ReadAnalogVoltage.ino` ;
— le soft Python `analyse.pyw`.

2 Protocole

Afin de mettre en place le dispositif sur une nouvelle machine, nous vous conseillons de suivre les instructions suivantes :

2.1 Paramétrage de la plateforme Arduino

1. Se procurer un micro-contrôleur Arduino, de modèle *Uno* de préférence. Vous pourrez en trouver à l'adresse `http://www.arduino.cc/en/Main/ArduinoBoardUno`.
2. Installer le logiciel Arduino sur votre ordinateur. Ce logiciel est disponible en téléchargement gratuit à l'adresse `http://www.arduino.cc/en/Main/Software`.
3. Brancher le micro-contrôleur Arduino en USB à l'ordinateur et lancer le logiciel Arduino.
4. Ouvrir le fichier `ReadAnalogVoltage.ino` depuis le logiciel Arduino.
5. Dans ce même logiciel, dans la barre de menu, sélectionnez la rubrique « Outils », et dans la sous-rubrique « Type de carte », choisir le modèle de carte que vous avez branché (Uno si vous avez bien suivi ce tutoriel). Dans « Outils », la sous-rubrique « Port » vous permet de choisir le port USB sur lequel est branché votre carte Arduino (vous le repérez sûrement). Sélectionnez le bon port.
6. Une fois ces paramétrages effectués, cliquez sur le bouton « Vérifier » de la fenêtre principale du logiciel Arduino, puis « Téléverser ». Si tout s'est bien passé, le logiciel Arduino a été transféré sur la plateforme.

2.2 Au niveau du logiciel d'acquisition des tensions

1. Afin que le logiciel fonctionne, il faut dans un premier temps installer `python3.4` depuis le site `https://www.python.org/downloads/`
2. Ouvrez une « Invite de commande » sous Windows et tapez la commande `py` afin de vérifier que Python est correctement installé.
3. Il faut ensuite installer les librairies nécessaire au bon fonctionnement du logiciel à l'aide de la librairie `pip` native à Python 3.
4. Tapez les commandes suivantes dans le terminal : `py -m pip install matplotlib` et `py -m pip install pyserial`.
5. Maintenant, il vous faut équiper votre ordinateur de notre logiciel d'acquisition.
6. Si vous souhaitez apporter des modifications au logiciel de mesure, il vous suffit de l'ouvrir avec un éditeur de texte quelconque. Le logiciel est codé en language Python et est documenté de manière à vous en faciliter la compréhension. Nous vous conseillons d'effectuer une copie du logiciel avant d'y apporter une quelconque modification, au cas où il y aurait un problème.

Notice d'utilisation à l'attention des étudiants

14 juin 2015

---

1. Vérifier que la plateforme électronique Arduino est bien connectée à l'ordinateur *via* un port USB.
2. Lancer le logiciel de mesure `analyse.pyw`
3. Choisir la durée de l'acquisition dans le premier onglet du logiciel.
4. Lancer l'acquisition en cliquant sur « Nouvelle analyse ».
5. Une nouvelle fenêtre s'ouvre. Grâce aux nombreux outils disponibles dans la barre des tâches situées sous le graphique, il vous est possible de zoomer sur les courbes, de lire les coordonnées du point où se situe le curseur de votre souris...
6. Vous pouvez à présent avoir accès aux données mesurées dans les onglets « Acquisition avec ressort » et « Acquisition sans ressort », avec les tensions mesurées à un instant t associées à cet instant t.
7. Il vous est également possible d'accéder aux amplitudes de chaque signal en allant sur le premier onglet « Général ».
8. Vous pouvez relancer une acquisition à n'importe quel moment en cliquant à nouveau sur « Nouvelle analyse ». Une nouvelle fenêtre de courbe va alors apparaître et les valeurs des tableaux de données vont être actualisées avec celle de la nouvelle analyse. Il est alors conseillé de fermer la courbe de l'ancienne analyse afin d'éviter les confusions. Si vous souhaitez la conserver, il est néanmoins possible de l'enregistrer à l'aide du bouton « disquette » présent sous la courbe.
9. Si vous souhaitez extraire les données afin de les exploiter dans un tableur, utilisez la fonction « Exporter au format `.csv` » sur l'onglet « Général » du logiciel.
10. Une fois les données exportées, vous pourrez retrouver le fichier `.csv` [1] sous le nom data.csv dans le répertoire choisi. Attention, ce fichier écrasera le fichier précédent si un fichier du même nom est déjà présent dans le répertoire.

1. Comma Separated Values.