Projets Industriels – Elec4

Année scolaire 2014-2015

Rapport Bibliographique

Création d'un sismomètre pédagogique de qualité professionnelle

Version 1

Etudiants : Jonathan David, Clément Le Marquis, Mickaël Renault

Encadrant : Luc Deneire, Enseignant Chercheur à Polytech'Nice Sophia

Industriel : Jean-Luc Berenguer, Professeur de Sciences de la Vie et de la Terre, au Centre International de Valbonne

Ecole Polytechnique Universitaire de Nice Sophia-Antipolis, Département électronique

930 route des Colles, 06410 BIOT

REMERCIEMENTS

Avant de commencer cette bibliographie, nous tenions à remercier certaines personnes, qui nous ont permis de nous mettre sereinement au travail, en nous apportant aide, conseils et connaissances quant à la sismologie.

Nous remercions donc en premier lieu monsieur Luc DENEIRE, enseignant chercheur à l'école polytechnique universitaire Polytech Nice-Sophia Antipolis et encadrant de ce projet, pour la confiance qu'il a en nous et l'aide fournie jusqu'à présent. Nous remercions également monsieur Robert PILLET, sismologue à GeoAzur, de nous avoir accordé ces quelques heures pour nous expliquer le principe de la sismologie et de la sismométrie. Pour finir, nous remercions monsieur Jean-Luc BERENGUER, professeur de SVT au Centre International de Valbonne, qui a contribué au lancement du projet et nous a accueilli dans les locaux du CIV pour nous préciser les enjeux et organiser une rencontre avec les élèves.

SOMMAIRE

Introduction

Le principe de ce projet est de concevoir un sismomètre low-cost accessible et pédagogique, puisqu'il doit pouvoir être implanté dans d'autres écoles Erasmus+ partenaires tout autour du globe. Quatre écoles européennes sont en concurrence autour de ce projet : le CIV, une école italienne et deux écoles anglaises.

Notre projet s'articule autour de trois objectifs. Le premier est pédagogique. Les élèves qui suivent ce projet ont des cours de sismologie, ils doivent être capables de comprendre le fonctionnement de la chaine d'acquisition complète de notre sismomètre. Le second objectif concerne les coûts. Le but du projet est de faire un sismomètre low-cost, qui ne doit pas dépasser les 200 euros. Nos recherches quant au matériel et technologies qui seront utilisées prendront donc en compte ce critère. Le dernier objectif est un objectif de portabilité. En effet, le produit final ne devra être ni trop encombrant ni trop fragile, car il devra pouvoir être transportable.

A propos des personnes concernées par ce projet, on distingue trois groupes. Tout d'abord une équipe pédagogique composée de professeurs du CIV, d'un enseignant chercheur de Polytech, et d'un sismologue de GéoAzur. On retrouve ensuite un groupe d'une quinzaine d'élèves du CIV, en seconde pour la plupart, qui se sont intéressés au projet et vont passer toute une partie de l'année à étudier la sismologie. Enfin, nous, équipe de trois étudiants en quatrième année d'électronique à Polytech Nice-Sophia.

Nous avons choisi ce projet pour diverses raisons. C'est avant tout un sujet pluridisciplinaire comme nous le verrons au cours de cette bibliographie. Les parties mécaniques, automatiques, électroniques et informatiques sont toutes intrinsèquement liées, et c'est à nous de les faire fonctionner ensemble. Ensuite, il y a la partie pédagogie. Nous rencontrerons plusieurs fois les élèves du CIV qui vont suivre ce projet, nous devons donc être en mesure de leur expliquer notre avancée en faisant un effort de vulgarisation, ce qui implique que nous devons maitriser nous même le sujet.

C'est donc dans l'optique des points évoqués précédemment que nous avons établi notre plan. Dans un premier temps, nous nous intéressons ici à la partie sismologie/sismométrie, pour comprendre le fonctionnement de l'appareil et le principe de la mesure. Nous rentrerons ensuite plus dans le détail, à savoir comment la mesure se fait réellement, le pourquoi de la nécessité d'une rétroaction pour avoir des mesures plus précises, ainsi que la numérisation du signal, avant de nous intéresser à la partie traitement de données et interaction, et des outils que nous utiliserons pour cela. Mais avant cela, toute une partie sera consacrée à l'aspect gestion du projet, en évoquant notre calendrier, le cahier des charges, et une étude des coûts, de façon à savoir si nous rentrons ou non dans le budget lors de cette première approche.

Chapitre I : Gestion du projet

I.1. Calendrier du projet

Voilà le calendrier provisoire du projet que nous avons établi, et qui a été validé par les responsables M. Berenguer et M. Deneire. (L'explication des codes-couleurs se trouve au bas du tableau.)

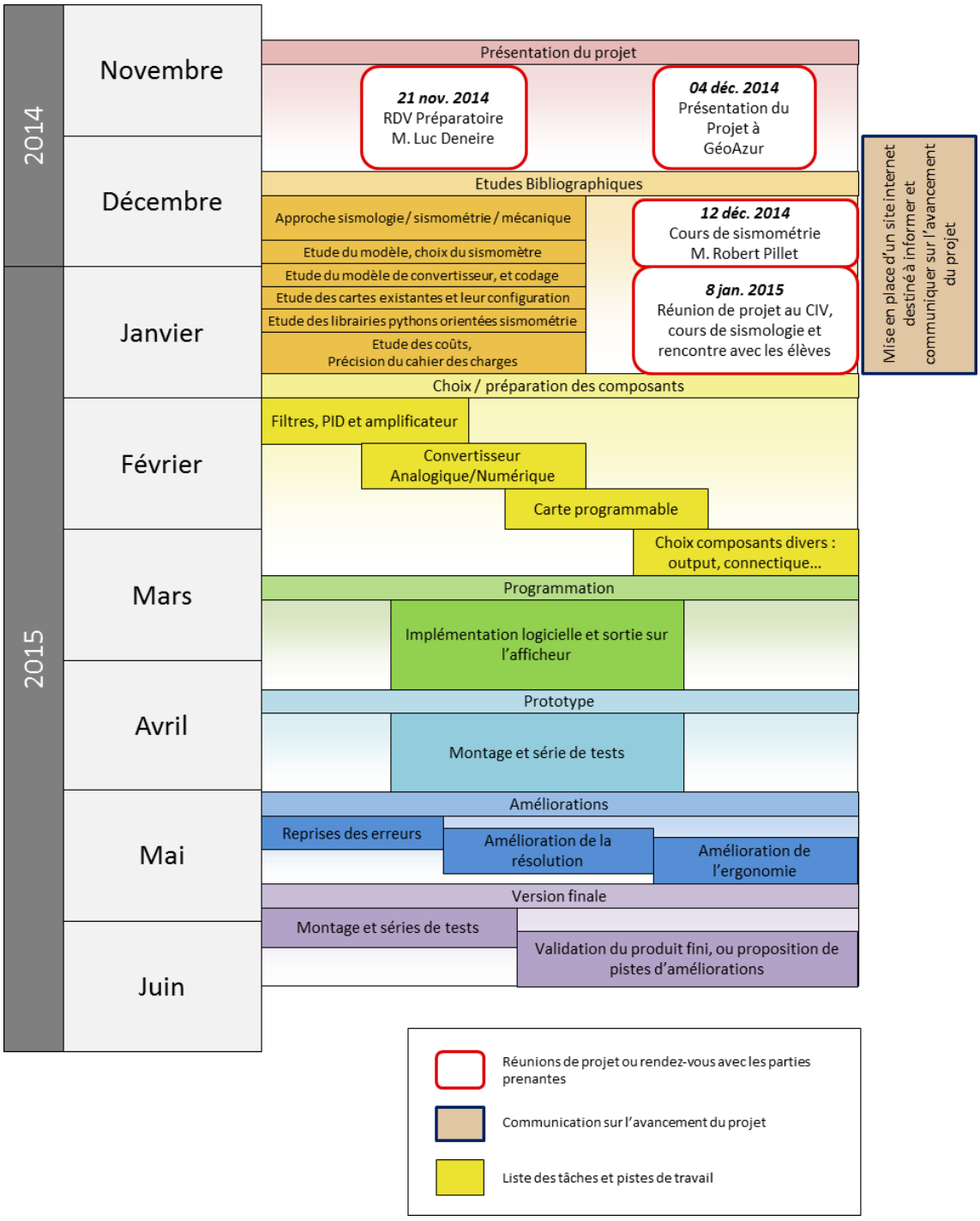


Tableau I.1 *Calendrier provisoire proposé pour le projet*

Nous avons, à travers ce calendrier, souhaité organisé notre temps en faveur de la préparation, la réflexion scientifique, et parfois la simulation pour ne pas gaspiller les moyens engagés dans ce projet. Nous avons jugé important également, de prévoir une période de prototypage au mois d'avril, afin de donner la possibilité aux lycéens travaillant à nos côtés de disposer d'un produit fonctionnel pour leur départ à Naples. En effet, un voyage est organisé dans le cadre de leur investissement dans ce projet, pour leur permettre de partager leur expérience avec les autres écoles Erasmus+ partenaires. Enfin, le produit fini doit être disponible pour la fin du mois de mai, voire début juin pour que cela s'associe à nos contraintes horaires associées.

I.2. Présentation du cahier des charges

Nous rappellerons tout d'abord le cahier des charges général de notre projet, qui nous a été communiqué avant même son commencement, puis nous verrons les précisions qui lui ont été apportées après cette première étape de recherche.

I.2.1. Cahier des charges global

Le voici tel qu'il a été rédigé et transmis à l'école Polytech'Nice Sophia par M. Jean-Luc Berenguer, professeur agrégé de Science de la Vie et de la Terre au Centre International de Valbonne, et responsable « industriel » du projet Sismomètre :

Cahier des charges :

1. Prendre connaissance des bases de sismologie et de techniques de mesure associées avec Monsieur Bérenguer et un chercheur du laboratoire Geoazur (ce sont les mêmes bases qui sont enseignées aux élèves de seconde).

2. Définition du cahier des charges de l'interface entre le capteur (type Slinky Spring ou TC1, voir ci-dessous) et la carte Raspberry PI, en fonction des caractéristiques données ci-dessous (Digitizer). Sélection et achat de cette interface.

3. Définition du cahier des charges du programme d'enregistrement sur la plateforme. Cette définition se fera avec les élèves du C.I.V.

4. Programmation de la plateforme pour finaliser le produit.

Références :

Slinky Spring Seismometer - http://www.mindsetsonline.co.uk

Digitizer - one per seismometer will be needed http://www.mindsetsonline.co.uk/

TC1 - http://tc1seismometer.wordpress.com/

I.2.2. Précision du cahier des charges

Au regard des éléments apportés dans la suite du document, conséquences mêmes de nos recherches sur le plan sismologique, sismométrique, électronique et informatique, nous pouvons préciser le cahier des charge portant sur la chaine d'acquisition des données du capteur sismométrique de la manière suivante :

Capteur sismométrique

Le LVDT que nous évoquerons par la suite est la solution apportant le meilleur compromis précision/coût, nous devrons donc implémenter et calibrer ce capteur directement sur le sismomètre du projet.

Numérisation du signal et rétroaction

La numérisation doit être précise et ne générer qu'un bruit minimum. Pour cela, nous travaillerons sur la conversion de type sigma-delta.

La rétroaction par PID sera implémentée numériquement pour un meilleur calibrage et une meilleure flexibilité. Le processeur utilisé pourra être celui d'un micro-contrôleur ou d'un micro-ordinateur. La partie numérique traitant le signal de la boucle de rétroaction ne sera pas forcément la même que celle qui traite les données en sortie de chaine.

Acquisition et affichage des données

L'acquisition du flux de données doit être envisagée par interface I2C ou SPI (voir partie IV.2.3), selon le type de convertisseur Analogique Numérique.

Les signaux seront reçus par le port GPIO d'une carte Raspberry Pi de modèle B+, fonctionnant sous le système d'exploitation Rasbian Wheezy.

Le traitement des données sera programmé en Python sous IDLE3. Il permettra entre autre, de stocker le signal, de l'encoder au format international SEED, mais également de traiter numériquement le signal et d'afficher sous différentes formes les données du sismomètre.

L'affichage des données peut être, selon le budget, soit par un mini-écran intégré au produit, soit à la charge de l'utilisateur.

L'enregistrement des données se fait sur une micro-SD de classe 10 connectée à la Raspberry PI.

I.3. Analyse des coûts

Il est d'ores et déjà possible de prévoir le coût approximatif de la partie électronique du projet (c'est-à-dire sans prendre en compte la conception même du sismomètre). Cependant, il faut lire ces estimations avec une certaine distance, car les prix indiqués par les revendeurs peuvent être amenés à changer. Nous supposons donc judicieux de prévoir une marge de 10% sur les montants indiqués.

Par ailleurs, les revendeurs suggérés par l'école Polytech'Nice Sophia ne disposant pas systématiquement des composants recherchés, ou n'étant pas assez compétitifs, nous avons cherché les

revendeurs les plus accessibles. En effet, nous apporterons un soin tout particulier à respecter la volonté de proposer un produit final *low cost* à moins de 200€.

Composants nécessaires au projet :

Produit	**Prix**	**Site marchand**
Convertisseur Analogique Numérique AD7711ARZ	40€	www.farnell.com
Convertisseur Numérique Analogique ADAU1962WBSTZ	11,80€	www.farnell.com
Raspberry Pi B+	35€	www.farnell.com
Boitier transparent pour Raspberry Pi B+	3€	www.amazon.fr
Carte microSD classe 10 avec adaptateur SD	5,29€ *FPI*	www.amazon.fr
Hub USB (Alimente la Raspberry et permet de disposer de 3 ports USB supplémentaires), NLUSB2-222P	10,44€	www.modmypi.com
Carte RTC Haute précision DS3231	6,55€ *FPI*	www.amazon.fr

FPI : Frais de port inclus

Tableau I.2 Estimation des coûts des composants nécessaires au projet

Composants complémentaires possibles :

Produit	**Prix**	**Site marchand**
Récepteur GPS USB GlobalSat BU-353S4	29,40€ *FPI*	www.dx.com
Boitier 6 piles pour rendre la carte Raspberry PI autonome	4,81€	www.amazon.fr
Limiteur de tension LM2596	1,84 *FPI*	www.amazon.fr
Ecran LCD TFT 2.4'' tactile pour Raspberry Pi	25€ *FPI*	www.amazon.fr

FPI : Frais de port inclus

Tableau I.3 Estimation des coûts des composants accessoires au projet

I.4. Premier bilan

La première partie du projet nous permet d'établir une organisation, un processus général qui nous permet d'avancer dans de bonnes conditions pour notre travail collectif.

Nous avons transmis dès que possible notre calendrier provisoire pour qu'à tout moment nos collaborateurs puissent comprendre le déroulement de notre travail.

La communication a été établie de manière la plus claire possible, en prenant cependant soin de ne pas être trop fréquente non-plus. En effet, en complément du travail qui est attendu de notre part, nous avons souhaité mettre à disposition des parties prenantes un site dans le but d'informer, de partager, d'illustrer notre cheminement commun dans ce projet.

Les rencontres ont été productives et enrichissantes, nous avons pu cibler de manière plus claire les besoins et redéfinir les moyens d'y répondre. Par ailleurs, et il est difficile de le faire apparaitre scientifiquement dans ce rapport, nous avons compris les enjeux pédagogiques pour les étudiants du CIV et nous assurerons notre rôle de « coopérateur » dans le cadre de leur propre aventure.

Le travail qui nous attend semble maintenant bien défini, nous apporterons sans-doute quelques modifications ou ajustement d'implémentation si nécessaire, dans le souci de se rapprocher au plus près des exigences qui nous ont été transmises.

Chapitre II : Sismologie et sismométrie

II.1. Domaine de travail en sismologie

Un champ sismologique se caractérise par deux composantes.

II.1.1. Des fréquences de l'ordre de la seconde

La première est la fréquence. Celle-ci se mesure en Hertz (Hz), et son inverse, la période, en seconde (s).

On observera dans l'étude des tremblements de terre des fréquences variant de 10^{-5} à 100 Hz. Chaque sismomètre ayant ses caractéristiques propres, ils n'ont bien entendu pas tous la même bande passante, ni la même période propre. Un sismomètre amateur captera une moins large gamme de fréquences, tandis qu'un appareil de qualité professionnelle pourra capter une plus large bande passante.

Cette bande de fréquences correspond bien évidemment à différents types d'ondes sismiques. Ce sont des ondes élastiques, qui peuvent traverser un milieu sans le modifier. Les particules vont « pousser » les particules voisines avant de revenir à leur place, traduisant ainsi le déplacement de l'onde. On peut distinguer deux types d'ondes de volume : les ondes P (pression), et les ondes S (cisaillement).

Ces deux types d'ondes se distinguent sur les enregistrements, et permettent d'estimer à quelle distance du sismomètre se trouve l'épicentre d'un séisme.

On distingue trois catégories : les ondes de volume de séismes, les ondes de surface, et les vibrations propres de la Terre (également appelées modes normaux). On retrouve les ondes de volume entre 0,2 Hz et 10 Hz, les ondes de surface entre 0,002 Hz et 0,2 Hz, et les modes normaux entre 0,3 mHz et 20 mHz. Pour illustrer au mieux, voici un graphique de synthèse :

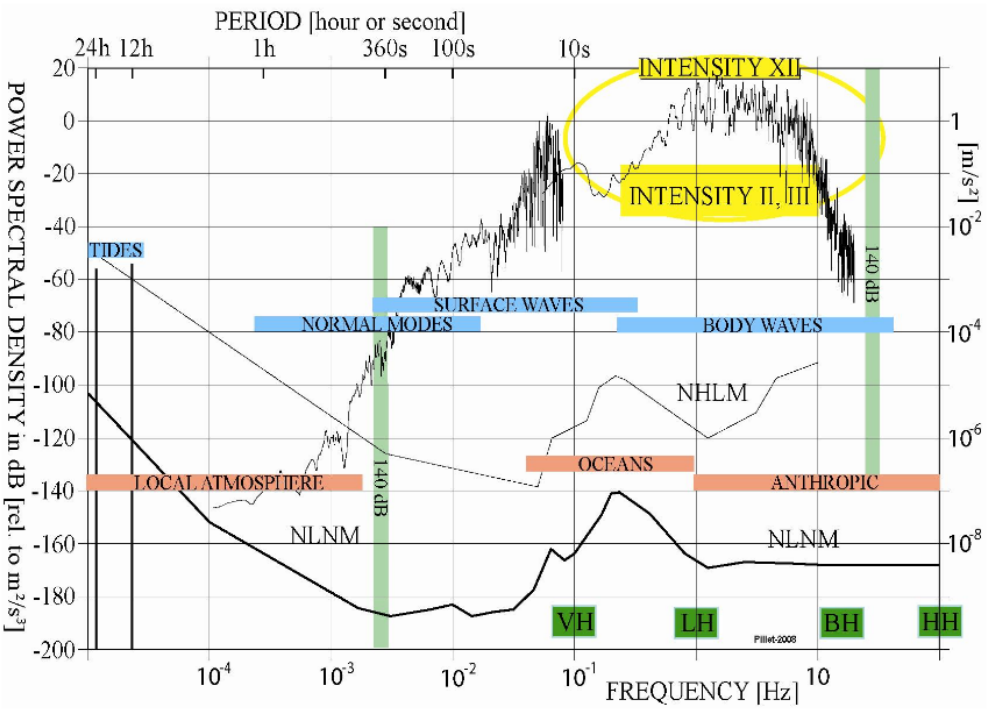


Figure II.1. *Amplitudes des signaux sismiques en fonction de la fréquence et du temps.*

Les vibrations propres de la Terre sont créées par les ondes de surface de longue période, donc de faible fréquence. Une fois qu'elles ont fait le tour de la Terre, ces ondes peuvent se retrouver en phase avec elle-même. Le mode le plus grave à une période de 54 minutes, soit de 0,3 mHz.

II.1.2. Une mesure de l'amplitude

La deuxième caractéristique d'un champ sismologique est l'amplitude de la vibration. Sur la Figure II.1, l'amplitude des différentes ondes selon la fréquence est représentée sur les ordonnées, en échelle logarithmique, en décibel sur l'axe de gauche, et en m/s² sur celui de droite.

En effet, le mouvement de vibration du sol peut s'exprimer soit en déplacement, en vitesse, ou encore en accélération. Ainsi, le bruit de fond (nous expliquerons plus loin ce que c'est) d'un site calme est de l'ordre de 10^{-10} m/s². Il est aisé de passer d'une grandeur (déplacement, vitesse ou accélération) à l'autre avec les dérivations et intégrations temporelles, et chacune d'elles permet de représenter l'amplitude du séisme, selon l'information que l'on souhaite mettre en avant.

Par rapport à toutes les secousses enregistrées à l'aide d'un sismomètre, seules celles ayant des accélérations supérieures à 10^{-2} m/s² sont ressenties par l'homme. Ainsi, de nombreux séismes ont lieu chaque jour, mais leurs amplitudes étant tellement faibles, ils passeraient inaperçus si nous n'avions pas des moyens fiables pour les détecter.

II.1.3. Le bruit de fond sismique

Un problème qui se pose lorsque l'on veut détecter des séismes est ce que l'on appelle le bruit de fond. Ce bruit de fond est généré, entre autre, par les activités humaines et les océans. C'est pour cela que les sismomètres sont généralement installés dans des zones calmes. On peut ainsi réduire les bruits générés par l'homme (voiture, vibrations dues à une machine, etc…). En revanche, on ne peut éviter le bruit de fond des océans, même dans des sites très éloignés des côtes.

Les fréquences de l'activité humaine peuvent aller jusqu'à 1 Hz, tandis que celles de l'activité océanique se situent entre 0,08 Hz et 0,2 Hz. Il faut donc en tenir compte lors de mesures, notamment lorsque le sismomètre se trouve sur une île par exemple.

Il a donc été défini un bruit de fond minimum, calculé par le rapport entre le *New Low Noise Model* (NLNM) et le *New High Noise Model* (NHNM). Ces termes sont déterminés en superposant les relevés des meilleurs sismomètres disposés autour du globe. Ainsi, le NLNM correspond au minimum de bruit de fond relevé, tandis que le NHNM correspond au bruit de fond maximal observé jusqu'à présent.

II.2. Utilisation de l'inertie pour la mesure, différents types de sismomètres

II.2.1. L'inertie

De nombreux moyens ont été cherchés pour mesurer les vibrations du sol. La première solution qui viendrait à l'esprit serait d'utiliser le *Global Positioning System*, plus connu sous le nom de GPS. On utilise les antennes GPS, qui sont aujourd'hui déployées à grande échelle autour du globe. Elles détectent très bien les très gros séismes, mais du fait de la qualité de leur mesure, et de leur fréquence d'échantillonnage (de l'ordre d'une mesure par seconde), il est impossible de détecter et de mesurer le bruit de fond de la Terre. Les sismologues ont alors eu l'idée d'utiliser le principe d'inertie.

Le principe d'inertie, ou première loi de Newton, nous dit que si l'on souhaite déplacer une masse, il faut lui transmettre de l'énergie.

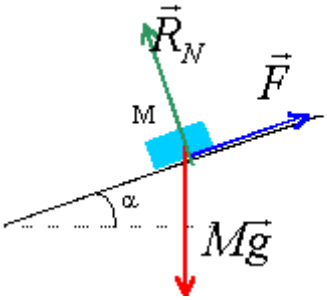


Figure II.2. *Illustration du principe d'inertie.*

Ainsi, si l'on prend un solide (en bleu sur la figure ci-dessus) posé sur un plan incliné, trois forces s'appliquent à lui : la résultante R_N, les frottements F, et son propre poids Mg. Tant que la force de frottement est suffisamment importante, le solide restera immobile, mais dès que le plan est trop incliné ou si la masse de notre solide est trop importante, celui-ci se mettra en mouvement.

C'est donc ce genre de repère que les sismologues ont mis au point. Si l'on remplace par exemple notre solide précédent par une boule de pétanque, qui a un point de contact associable à un point, elle ne suivra pas les mouvements horizontaux rapides du plan du fait de son poids, elle gardera sa position initiale. On crée ainsi un nouveau référentiel virtuel fixe. Les sismologues ont alors construit des appareils de mesure à partir de ces constatations.

Si on modélise notre système, on obtient :

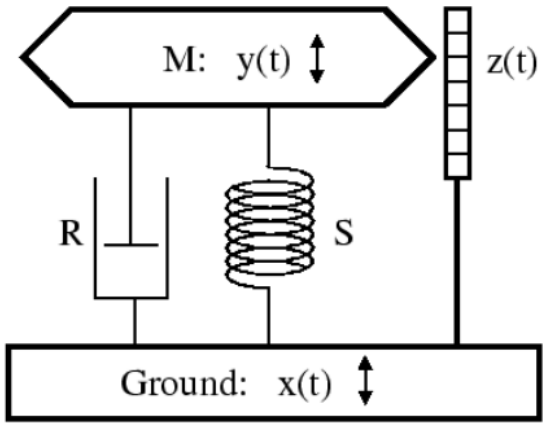


Figure II.3. *Schéma d'un oscillateur constitué d'une masse d'inertie, d'un amortisseur et d'un ressort.*

On y retrouve une masse M, liée au sol par un ressort de dureté S et un système de frottement R, et sur le côté une échelle z(t) qui permet de mesurer les oscillations de la masse par rapport au référentiel terrestre. Le système de frottement est quant à lui présent pour atténuer les vibrations de la masse après la fin d'une secousse. En effet, si on pose le système sur une table, que l'on relie la masse et la table uniquement par un ressort, et que l'on tape sur la table pour simuler une vibration, la masse va se mettre en mouvement, et va osciller jusqu'à retrouver son équilibre, et ce bien après la fin des vibrations. Un tel système nous permettrait alors de détecter le début d'un séisme, mais non la fin, et on ne pourrait mesurer l'amplitude des vibrations. D'où la nécessité de cette partie.

La mise en équation et la résolution de ce système étant un peu longue, nous vous invitons à aller voir les travaux de Mr. PILLET soit dans son livre, soit à la fin de cette bibliographie en Annexe A où une partie des calculs sont repris.

II.2.2. Sismomètres horizontal et vertical

Les sismomètres sont donc des appareils destinés à évaluer les mouvements du sol. Pour cela, ils sont composés de la partie mécanique détaillée dans le paragraphe précédent, et sont équipés d'un capteur mécanique et d'un transducteur. On retrouve ensuite le sismographe, composé du sismomètre, d'un amplificateur et d'un enregistreur, que ce soit papier ou électronique. Regardons tout ceci d'un peu plus près.

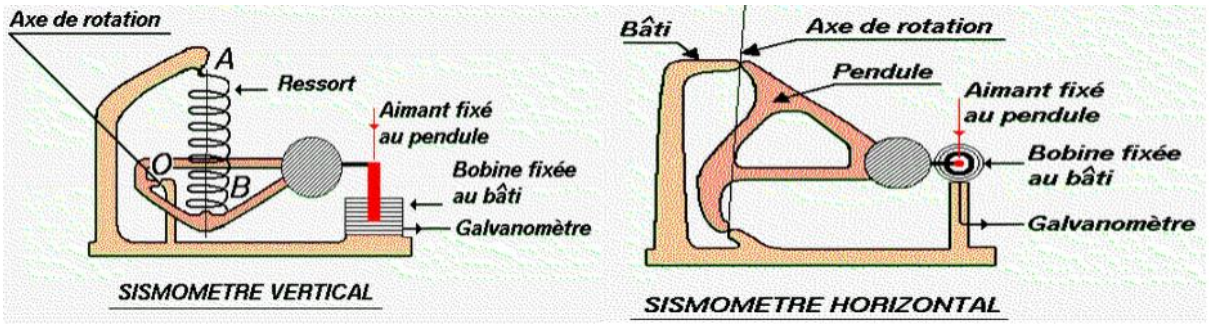


Figure II.4. *Schémas d'un sismomètre vertical gauche et d'un sismomètre horizontal à droite*.

Une combinaison des deux appareils ci-dessus permet alors de détecter aussi bien les vibrations verticales que les vibrations horizontales. On peut également trouver des sismomètres hybrides, prenant en compte les trois degrés de liberté en translation, comme le sismomètre SEIS, qui comporte six axes.

Revenons au sismographe, qui est l'objet de ce projet. On peut le schématiser de la façon suivante :

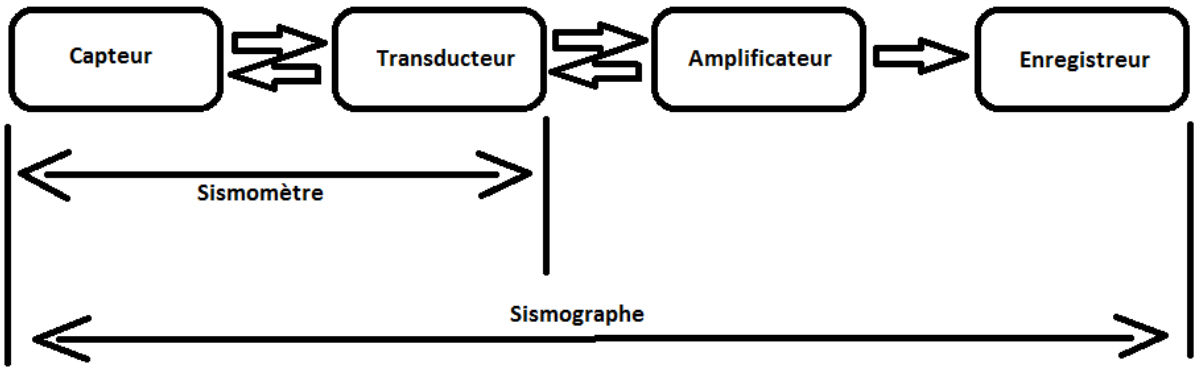


Figure II.5. *Représentation schématique d'un sismographe*.

Le capteur mécanique permet de déterminer le déplacement du sol, le transducteur va transformer ce déplacement en une tension, qui sera amplifiée par l'amplificateur, tension qui sera ensuite exploitée pour déterminer l'amplitude de chaque vibration. La rétroaction entre capteur et transducteur et entre transducteur et amplificateur est expliquée en partie III de cette bibliographie.

II.3. Transformation du déplacement de la masse en tension électrique et précision des mesures

II.3.1. Présentation du problème

Prenons le cas d'un sismomètre vertical, puisque c'est celui qui va nous intéresser au cours de ce projet. Lors d'un séisme, l'appareil va vibrer, il va donc y avoir un mouvement de créé. C'est ce mouvement qui va nous intéresser, puisqu'il va falloir le transformer pour qu'il devienne exploitable par un ordinateur. C'est là le rôle du transducteur.

Le transducteur va transformer le mouvement de la masse en une grandeur mesurable, une tension généralement, qui pourra ensuite être convertie et exploitée par la partie numérique (partie III et IV). L'arrivée du transducteur a révolutionné le monde de la sismologie, car il a permis de mettre au point une technique de substitution au relevé sur papier. Il devint rapidement électromagnétique, le phénomène d'induction permettant au mieux cette transformation mouvement/tension.

II.3.2. Le couple bobine/aimant

Ce phénomène d'induction est créé ici par un couple bobine/aimant. L'aimant est lié à la masse, la bobine au support, et lors de vibration, la bobine voit un champ magnétique qui se déplace. Redonnons quelques explications quant à l'induction électromagnétique.

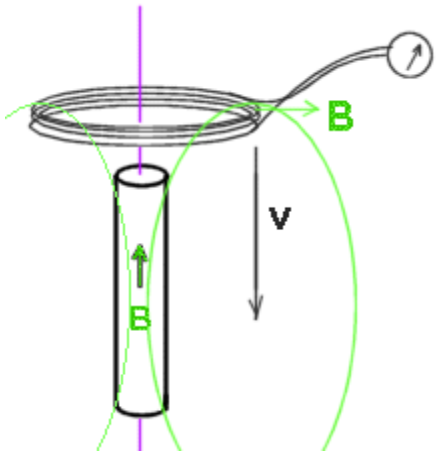


Figure II.6. *Schémas issus du site marmet.org, schématisant le déplacement d'un aimant dans une bobine.*

L'aimant est de base polarisé, et génère donc un champ magnétique. Lorsque la bobine va passer autour de lui, un courant d'induction va donc se créer dans la bobine en fonction de la vitesse, d'après l'induction de Lorentz :

$$\vec{F} = q * \vec{v} \wedge \vec{B} \qquad \text{(II.1)}$$

où F est la force de Lorenz (en Newton), q la charge de la particule (en Coulomb), v la vitesse du déplacement de l'aimant dans la bobine (en mètre par seconde), et B le champ magnétique (en Tesla). Cette équation peut également s'écrire :

$$F = \sigma * i \qquad \text{(II.2)}$$

où σ est le coefficient du couple aimant/bobine (en Newton par Ampère) et i le courant (en Ampère). On en déduit alors l'équation suivante :

$$i = \frac{q * \vec{v} \wedge \vec{B}}{\sigma} . \vec{u} \qquad \text{(II.3)}$$

où u est un vecteur unitaire, colinéaire au produit vectoriel.

L'équation (II.3) nous montre bien que le courant induit, donc la tension aux bornes de la bobine, est directement proportionnel à la vitesse de déplacement de l'aimant dans la bobine. Il faudra bien évidemment étalonner le système.

Le couple aimant/bobine possède deux caractéristiques : un transducteur de mesure, et un forceur. Le premier donne une tension proportionnelle à la vitesse de déplacement entre aimant et bobine, sans perturber le pendule, tandis que le second produit une force proportionnelle à l'intensité du courant qui circule dans la bobine.

II.3.3. Précision des mesures

Un sismomètre n'a de réel intérêt (excepté dans le cadre pédagogique) que s'il peut fournir des mesures fiables. Pour cela, il faut qu'il possède une période « infinie », ce qui correspond généralement à T = 30s. Or le seul élément sur lequel on peut jouer, c'est le ressort. L'équation de la tension du ressort nous est donnée par la formule suivante :

$$T = -k * (l - l0) \qquad \text{(I.4)}$$

où T est la force du ressort, k sa constante, l sa longueur effective et l0 sa longueur à vide. Mais si on parvient à créer un ressort de longueur nulle (nous verrons plus loin comment), la tension ne dépend plus que de la longueur effective :

$$T = -k * l \qquad \text{(I.5)}$$

On simplifie alors grandement les équations du sismomètre. Voyons maintenant l'application de cette méthode à un sismomètre vertical :

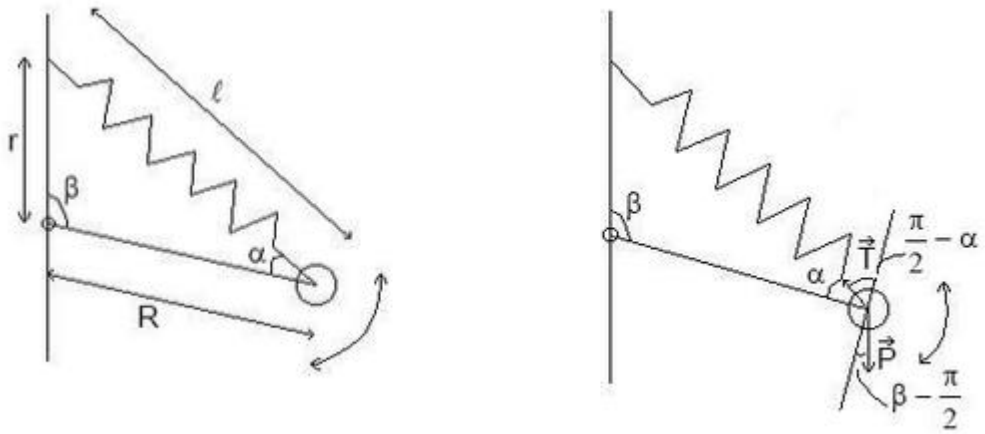


Figure II.7. *Schémas issus du musée de sismologie de Strasbourg.*

Deux forces s'appliquent à la masse : son poids et la tension du ressort, que l'on a pris de longueur nulle. Si on projette ces forces sur la tangente de la trajectoire, on obtient :

$$Pp = m*g*\cos(\beta - \pi/2) = m*g*\sin(\beta) \quad \text{(II.6)}$$

$$Tp = k*l*\cos(\pi/2 - \alpha) = k*l*\sin(\alpha) \quad \text{(II.7)}$$

soit à l'équilibre :

$$m*g*\sin(\beta) = k*l*\sin(\alpha) \Leftrightarrow m*g = k*l*\frac{\sin(\alpha)}{\sin(\beta)} \quad \text{(II.8)}$$

De plus, la loi des sinus nous donne :

$$\frac{l}{\sin(\beta)} = \frac{r}{\sin(\alpha)} \Leftrightarrow \frac{\sin(\alpha)}{\sin(\beta)} = \frac{r}{l} \quad \text{(II.9)}$$

En injectant dans l'équation (I.8), on obtient donc :

$$m*g = k*r \quad \text{(II.10)}$$

Dans cette nouvelle équation, on ne retrouve plus que des constantes. Les angles n'interviennent plus, donc la position d'équilibre ne dépend plus d'eux. On a donc un système complètement stable, et on passe d'une position d'équilibre à une autre. On vient donc de créer un sismomètre de période « infinie ». De plus, la masse et le châssis sont indépendants, les mouvements sismiques du châssis ne seront donc pas ressentis par la masse d'inertie.

Un ressort de longueur nulle n'est bien évidemment pas réalisable en sens propre du mot. C'est un ressort dont toutes les spires sont jointives lorsque qu'aucune tension extérieure ne lui est appliquée. Il possède donc une tension qui lui est propre et qu'il faut dépasser pour l'allonger. Pour réaliser un tel ressort, on peut par exemple prendre un ressort conique que l'on va retourner. Une fois fait, le ressort cherchera à retourner dans sa position d'origine, mais s'il est correctement conçu, les spires se chevaucheront.

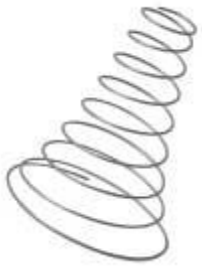


Figure II.8. *Ressort conique de longueur non nulle.*

En revanche, si le ressort n'est pas de longueur nulle, c'est-à-dire que les spires ne sont pas jointives, mais qu'elle est petite, on obtiendra une période finie, mais longue, suffisamment pour réaliser de bonnes mesures.

Chapitre III : Mesure, rétroaction et numérisation.

III.1. Capteurs.

III.1.1. Généralité sur les capteurs.

Le capteur, aussi appelé transducteur, est un des éléments qui composent une chaîne de mesure d'un système. Il permet de mesurer, et surveiller l'évolution d'un système physique complexe. Des grandeurs physiques sortent du système et le capteur permet d'en faire la mesure. L'objectif étant de pouvoir mémoriser et traiter ces informations relatives au système. Pour cela il est nécessaire de transformer les grandeurs physiques en grandeurs électriques proportionnelles qui seront par la suite adaptées par le conditionneur pour un traitement des données par le micro contrôleur. Toutefois les grandeurs mesurées peuvent être parasitées par des grandeurs dîtes d'influence, ce qui pose un problème sur la qualité du capteur.

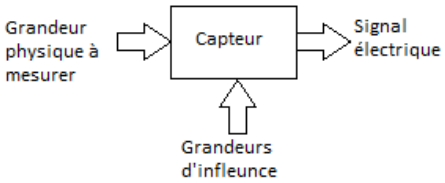


Figure III.1 *Schéma d'un capteur*

On peut distinguer deux grandes familles de capteurs, les capteurs actifs et les capteurs passifs.

Un capteur est dit actif si la transformation en grandeur électrique de la grandeur physique et directement faite par le capteur. « C'est la loi physique elle-même qui relie mesurande et grandeur électrique de sortie », *Capteurs: principes et utilisation*, F.Baudoin et M.Lavabre, édition Casteilla. Il s'agit par exemple d'une génératrice tachymétrique.

Un capteur est dit passif lorsque la modification d'un paramètre électrique tel que la résistivité, la permittivité électrique ou la perméabilité magnétique est due à l'influence d'un changement de la grandeur physique. Par exemple une mesure de résistance en fonction de la température. C'est donc par la variation de la résistivité du matériau que l'on accède à la variation de température.

Comme on peut le voir en figure III.1 le capteur est soumis à plusieurs contraintes issues de son environnement, tel que la température ou l'humidité. Pour notre étude nous ne prendrons pas en compte l'influence de ces paramètres. Cependant cela explique pourquoi les sismomètres sont souvent placés sous vide. En plus de ces problèmes externes, il existe des sources d'erreur issues du capteur lui-même c'est pourquoi on parle de qualité du capteur et qu'il est important de pouvoir étalonner un capteur en déterminant ces lois mathématiques de fonctionnement afin de pouvoir limiter les erreurs de mesure. Ces erreurs sont dues à la conception du capteur.

Dans le domaine de la sismométrie nous avons besoin d'un capteur capable de mesurer l'effet d'un séisme, nous allons nous intéresser uniquement à l'étude des ondes P (cf. partie II). Nous avons donc besoin d'un capteur capable de mesurer un déplacement.

III.1.2. Capteur de déplacement.

Les capteurs de déplacements sont essentiellement des capteurs passifs, ils vont donc travailler sur une modification de résistance, de capacité, ou bien de propriété magnétique.

Parmi les capteurs de déplacement et au vu des besoins de notre étude, nous nous limiterons à l'étude de capteurs capacitifs et inductifs. De plus ce sont des capteurs sans contact ce qui permet de s'absoudre de l'usure du capteur et d'augmenter la fiabilité des mesures de celui-ci. De plus ces types de capteurs permettent d'avoir une grande résolution de l'ordre du nanomètre.

Capteur capacitif

Ce type de capteur permet de mesurer des déplacements linéaires ou bien angulaires. Dans le cas du sismomètre nous avons besoin de pouvoir mesurer un déplacement linéaire donc il faut utiliser un capteur capacitif plan. Avec un tel capteur il est possible de mesurer des déplacements bien en dessous du millimètre.

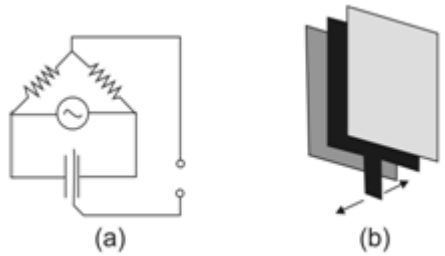


Figure III.2 *schéma capteur capacitif plan.*

On note *e* la distance entre les deux armatures (plaque grise et noire), une fixe (grise) et une mobile (noire) liée à la masse oscillante. La surface d'une armature est noté *S*. *Epsilon* est la permittivité du diélectrique. On a alors la valeur de la capacité grâce l'équation suivante :

$$C = \frac{\varepsilon * S}{e} \qquad \text{(III.1)}$$

Fonctionnement du capteur: Le mouvement de l'armature mobile va faire évoluer le champ électrique ce qui modifie la valeur de la capacité, ce que l'on mesure. Nous serons donc après analyse des données en moyen de retrouver la valeur de déplacement du pendule. Dans le but d'augmenter la précision du capteur il est possible de mettre les armatures sous forme de peignes imbriqués dans le but d'augmenter la surface capacitive.

On peut voir en Annexe B la photo d'un capteur capacitif présent sur un sismomètre industriel le SAS-2.

Capteur inductif.

Le principe de cette variété de capteur est d'entrainer une variation de flux magnétique grâce au déplacement. Il existe plusieurs familles de capteurs fonctionnant sur ce principe, nous retiendrons seulement la famille de capteurs par variation de mutuelle inductance entre deux circuits (*Linear Variable Differential Transformer*).

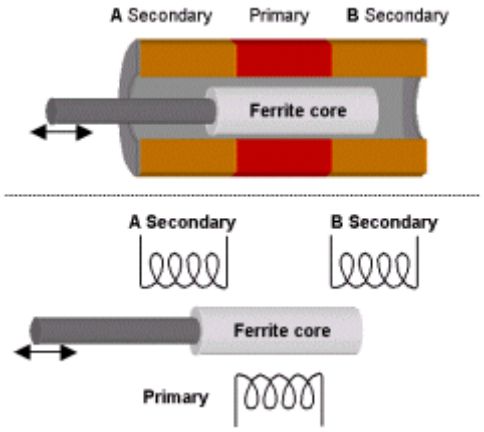


Figure III.3 *Schéma d'un capteur LVDT*

Fonctionnement du capteur : On envoie un signal sinusoïdal dans le primaire ce qui induit un signal sinusoïdal dans le secondaire. On étalonne le capteur de manière à avoir une tension de sortie nulle lorsque le système est au repos (ferrite immobile). Ainsi lorsque la ferrite entrera en mouvement nous pourrons mesurer un changement du signal de sortie qui sera proportionnel au déplacement.

Pour le développement de notre projet nous utiliserons un capteur type LVDT car il nous a paru que ce capteur était le plus simple à mettre en place (miniaturisation, utilisation et étalonnage). De plus l'idée du projet est de faire un sismomètre compréhensif, avec ce capteur on voit très bien le changement de la grandeur physique (déplacement) en une grandeur électrique. Le principe physique du couple aimant/bobine (ferrite/primaire ou ferrite/secondaire) utilisé par ce capteur, est explicité en partie II.3.2

III.2. Boucle de rétroaction.

On appelle boucle de rétroaction le fait de combiner en entrée une fraction du signal de sortie. Dans toute cette partie on note *p* la variable de Laplace.

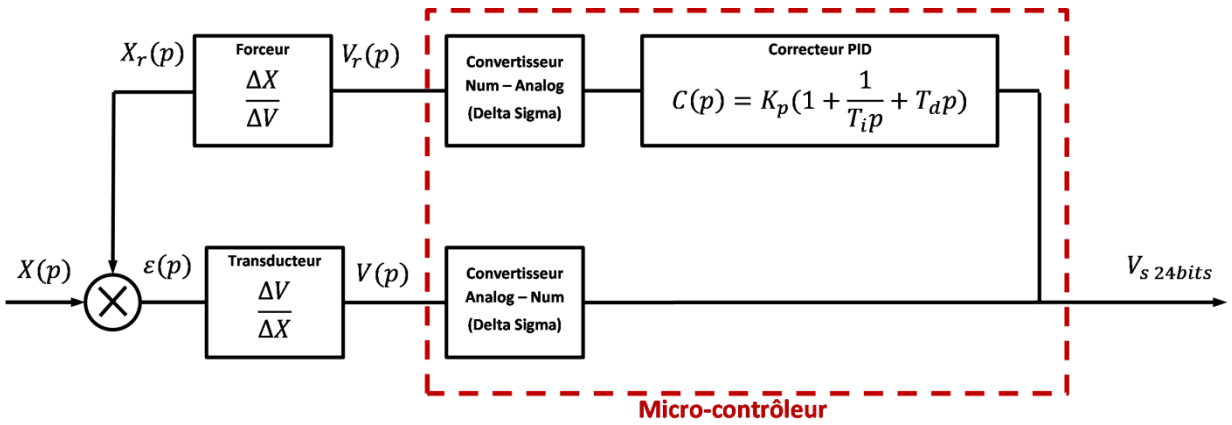


Figure III.4 *schéma de la boucle de rétroaction.*

On peut voir que l'entrée du transducteur est une combinaison du signal X(p) qui représente le mouvement de la masse dû au séisme et Xr(p) qui est la correction du déplacement (afin d'éviter les oscillations superflues). Le micro contrôleur fournit par le biais de la boucle de rétroaction une tension pour réduire les oscillations du système. Cette information est alors traduite par le forceur en un signal mécanique.

III.2.1. Nécessité de la boucle de rétroaction.

Comme on a pu le voir au cours de la partie précédente le sismomètre est soumis à des ondes qui mettent en mouvement la masse ainsi que le noyau de ferrite. On peut à partir de ce moment mesurer le déplacement (cf. III.1 Capteurs). Si l'on occultait la boucle de rétroaction, le système continuerait d'osciller du fait de la faible constante de raideur du ressort (cf. dimensionnement sismomètre) malgré le fait que les ondes du séisme n'agissent plus. Cependant grâce à la boucle de rétroaction on est capable d'intervenir sur le signal d'entrée pour le modifier. Ici le but de la boucle de rétroaction est de réduire le phénomène d'oscillation : on parle dans ce cas de rétroaction négative. En effet, la masse du sismomètre doit être constamment stabilisée dans son mouvement afin de ne pas obtenir sur le sismogramme les différents effets d'oscillation de celle-ci.

Pour notre étude nous avons pris la décision d'effectuer une rétroaction numérique que l'on implémentera sur un micro contrôleur. Nous avons fait ce choix du fait de la capacité d'adaptation que nous offre l'utilisation du numérique. En analogique une fois les composants mis en place il est plus difficile de les modifier. De plus l'utilisation du numérique peut être moins coûteux.

La boucle de rétroaction est composée d'un correcteur, dans notre cas un correcteur PID, puis d'un convertisseur Numérique-Analogique (cf. III.3.1 Convertisseurs). Ces deux éléments seront intégrés dans le micro contrôleur. Nous obtenons alors en sortie de celui-ci un signal électrique qu'il va falloir transformer en déplacement pour pouvoir réagir contre les oscillations qu'effectue la masse. Pour cela on utilise le transducteur dans son fonctionnement inverse, c'est-à-dire en forceur, afin de corriger la position du noyau de ferrite.

III.2.2. Correcteur PID.

L'acronyme PID signifie Proportionnel Intégral Dérivée. Ce correcteur est donc composé de trois parties distinctes qui ont toutes des actions différentes. La fonction de transfert du PID est :

$$C(p) = K(1 + \frac{1}{(Ti.p)} + Td.p). \qquad \text{(III.2)}$$

Correcteur Proportionnel.

La fonction de transfert du correcteur proportionnel est

$$C(p) = K \qquad \text{(III.3)}$$

Ce correcteur ne modifie pas la phase, et ne sert qu'à augmenter le gain. Ce dernier tend à augmenter la rapidité du système car à toutes les fréquences une augmentation du gain conduit à fournir plus d'énergie au système en commande. Si toutefois la valeur du gain est trop grande il est possible que le système devienne plus lent à converger.

De plus une augmentation du gain pourrait permettre une augmentation de la bande passante car cela augmente la pulsation de coupure.

Correcteur Intégral.

La fonction de transfert du correcteur proportionnel est :

$$C(p) = \frac{K}{p} \qquad \text{(III.4)}$$

Ce correcteur permet l'amplification des basses fréquences et diminue les hautes fréquences. L'amplification des basses fréquences nous assure la précision du système. En effet lorsque l'on ajoute ce type de correcteur au système on remarque que l'erreur statique est fortement diminuée, au vu du système l'erreur statique est de la forme $\frac{1}{1+K}$ est donc plus on choisit un K grand, plus l'erreur statique tend vers 0.

Ce correcteur ajoute aussi un retard de phase de $\frac{\pi}{2}$. Cette action a tendance à diminuer la stabilité du système. C'est pourquoi on associe souvent à ce correcteur une action proportionnelle, ce qui permet de trouver un juste compromis entre précision et stabilité. On parle alors de correcteur proportionnel intégral.

Ce type de correcteur est à la base des correcteurs à retard de phase, de la forme :

$$C(p) = K.\frac{1+\tau.p}{1+\alpha.\tau.p} \qquad \text{(III.5)}$$

Ils permettent de situer la position de la plage de retard de phase. En choisissant bien la valeur des paramètres K, α et τ on amplifie les basses fréquences (précision) sans modifier le comportement dynamique du système (rapidité, stabilité).

Correcteur dérivé.

Le correcteur dérivé pur est de la forme :

$$C(p) = K.p \qquad \text{(III.6)}$$

Il existe essentiellement dans la théorie car il n'est pas réalisable électroniquement du fait que le numérateur est de degré supérieur au dénominateur. De plus il ne respecte pas le principe de causalité.

Cependant il permet de se rendre compte de l'impact de l'aspect dérivé :

- Augmente la phase de $\frac{\pi}{2}$.
- Augmente les hautes fréquences, ce qui augmente l'effet du bruit.
- Réduit la précision et l'insensibilité aux perturbations du système.

Ce type de correcteur est à la base des correcteurs à avance de phase, de la forme :

$$C(p) = \frac{1+\alpha.\tau.p}{1+\tau.p} \qquad \text{(III.7)}$$

Ils permettent de situer où l'on veut que l'avance de phase ait lieu en général dans la gamme de fréquences où l'on mesure les marges de stabilité.

Le correcteur PID est donc un mélange de ces trois correcteurs. S'il est bien réglé le PID cumule les avantages de chacune des actions proportionnelles, intégrales et dérivées. Toutefois s'il est mal réglé il peut fortement dégrader le comportement du système. L'action intégrale doit être placée à basse fréquence, bien avant la fréquence de coupure, son but est d'augmenter le gain en basse fréquence. L'action dérivée est placée où on mesure les marges de stabilité, le but est d'apporter de la phase pour augmenter la marge de phase. L'action proportionnelle permet de finaliser le correcteur et de fixer la rapidité du système.

III.3. Numérisation du signal.

Le but de la numérisation du signal est de pouvoir utiliser des éléments d'électronique numérique, tel qu'un microprocesseur ou la carte Raspberry Pi pour stocker, pour pouvoir plus tard restituer les données sur une interface, et traiter les données. La numérisation permet une très bonne reproductibilité des traitements, ainsi que la possibilité de développer des fonctionnalités complexes de manière plus aisée (comme par exemple le correcteur PID). Cela entraîne aussi une réduction des coûts de production.

Cependant pour utiliser ce type de matériel il faut que les signaux transmis soient adaptés au langage (codage) dans lequel travail le matériel. Ainsi il faut donc passer par un convertisseur.

III.3.1. Convertisseurs

Le convertisseur est un dispositif dont le but est de convertir une grandeur en un autre type de grandeur. Dans notre cas on a besoin de deux types de convertisseurs. Un premier dont le but sera de convertir une grandeur analogique (type tension ou courant) en une grandeur numérique, pour pouvoir stocker l'information éventuellement la traiter (par exemple boucle de rétroaction) et de la transmettre. Puis d'un second convertisseur cette fois-ci numérique vers analogique qui sera présent dans la boucle de rétroaction pour que l'on puisse agir sur le mouvement d'oscillation de la masse.

Convertisseur Analogique/Numérique(CAN)

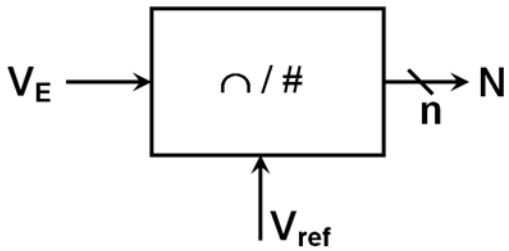


Figure III.5 *symbole d'un CAN*

Comme énoncé précédemment son but est de convertir un signal analogique (continu, une infinité de valeurs) en un signal numérique (discret, un nombre fini de valeurs), dans les documentations il est possible de trouver cet élément sous son acronyme anglais *Analog to Digit Converter*.

Pour effectuer ce type de conversion il faut suivre trois phases :

- Echantillonnage temporel.
- Quantification.
- Codage.

Les principales caractéristiques du convertisseur sont :

- Le quantum (LSB en anglais) : plus petite variation du signal d'entrée pour que l'on ait une modification du signal numérique de sortie. On le détermine par q=U_pleine_échelle/2^(N) où N est le nombre de bit du signal numérique.
- Le codage : code dans lequel est converti le signal numérique de sortie. Par exemple : Binaire naturel, DCB, binaire complément à 2, binaire avec bit de signe.
- Le temps de conversion : temps minimum que met le convertisseur pour délivrer un signal binaire stable, pour un signal d'entrée stable.
- Précision : écart maximal entre la sortie théorique et la sortie réelle. En effet le convertisseur est soumis à des erreurs de quantification, de linéarité, de gain et d'offset.
- Le rapport signal sur bruit, SNR (*Signal to Noise Ratio*) : correspond au rapport de la valeur efficace d'entrée sur celle du bruit. SNR = V_eff_entrée/V_eff_bruit. Dans le cas d'une entrée sinusoïdale pleine échelle SNR_dB=6.2*N+1.76 (N : nombre de bit)

La phase d'échantillonnage consiste à multiplier le signal analogique (tension) par un peigne de diracs :

$$x\_ech(t) = x(t) * \sum(\Delta(t - kT\_éch)) \quad \text{(III.8)}$$

D'un point de vu fréquentiel cette opération revient à périodiser le spectre du signal analogique X(f), compris entre fmin et fmax, toutes les périodes d'échantillonnage T_éch. Ce qui correspond à une opération de convolution entre X(f) et le spectre du peigne de diracs.

La phase de quantification consiste à déterminer la plage de Pleine Echelle qui correspond à la plage de variation acceptable du signal analogique d'entrée. Dès lors on divise cette plage de fonctionnement en N niveaux d'égale dimension (si on travaille en quantification uniforme) ou non. Cet écart entre deux niveaux correspond alors au quantum.

La phase de codage consiste simplement à choisir sous quelle forme sera codé le signal numérique de sortie.

Convertisseur Numérique/Analogique.

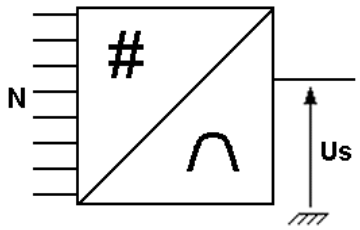


Figure III.6 *symbole d'un CNA*

L'objectif de cet élément est de convertir un signal numérique (discret, un nombre fini de valeurs) en un signal analogique (continu, une infinité de valeurs), dans les documentations il est possible de trouver cet élément sous son acronyme anglais *Digit to Analog Converter*.
Il est souvent suivi d'un filtre de lissage (filtre passe bas).
Pour effectuer ce type de conversion il faut suivre trois phases :

- Représentation temporelle.
- Quantification.
- Lissage. (cette étape n'est pas obligatoire cependant suivant l'utilisation analogique qu'il y a derrière elle est recommandée)

Les principales caractéristiques du convertisseur sont :

- Le quantum (LSB en anglais) : plus petite variation du signal de sortie pour que l'on ait une modification du signal numérique d'entrée. On le détermine par q=U_pleine_échelle/(2^(N) - 1) = V_ref/2^(N) où N est le nombre de bit du signal numérique.
- Le codage : code dans lequel est convertit le signal numérique de sortie. Par exemple : Binaire naturel, DCB, binaire complément à 2, binaire avec bit de signe.
- Le temps de conversion : temps minimum que met le convertisseur pour délivrer un signal binaire stable, pour un signal d'entrée stable.
- Erreur d'offset : décalage vertical de la caractéristique de transfert.
- Non linéarité intégrale : écart entre sortie analogique réelle et la valeur idéale associée au code.
- Le rapport signal sur bruit, SNR (*Signal to Noise Ratio*) : correspond au rapport de la valeur efficace d'entrée sur celle du bruit. SNR_dB=10*log(Psignal/Pnoise).

La phase de représentation temporelle consiste à représenter le message numérique en fonction du temps, on aligne sur axe temporel les messages successifs que l'on reçoit.

La phase de quantification consiste tout d'abord à déterminer la plage de Pleine Echelle sur laquelle on veut que le signal analogique de sortie corresponde. Dès lors on divise cette plage de fonctionnement en N niveaux d'égale dimension (si on travaille en quantification uniforme) ou non. Cet écart entre deux niveaux correspond alors au quantum. Ensuite on traduit le code en niveau de quantification. Ainsi on obtient une première image du signal analogique de sortie désirée que l'on appelle signal quantifié.

La phase de lissage consiste à lisser la courbe obtenue à la phase précédente pour éviter d'avoir des changements de valeur brutaux. Enfin on rappelle qu'un signal analogique est un signal continu donc ses variations ne peuvent pas être discontinues, d'où l'utilité du filtre de lissage.

III.3.2. Convertisseur Sigma Delta.

Avec une structure classique de convertisseur analogique numérique il est compliqué d'avoir un résultat rapidement et avec une grande précision. La rapidité d'un convertisseur est essentiellement due au nombre de comparateur qu'il utilise. Mais plus il y a de comparateurs plus la qualité de la linéarité différentielle est réduite.

Le principe du convertisseur sigma delta est de combiner qualité de linéarité différentielle (on n'utilise qu'un seul comparateur) et rapidité (on travaille sur la fréquence d'échantillonnage). Pour que le convertisseur fonctionne correctement il faut que le signal soit lent devant la fréquence d'échantillonnage. Le but sera alors de coder l'écart avec la valeur précédente ce qui revient à regarder le signe de la dérivée (le delta). On a donc besoin d'un seul comparateur qui nous indique si le signal augmente ou diminue (un seul comparateur). De plus si la fréquence d'échantillonnage est très grande devant l'évolution du signal, de telle manière que l'on puisse considérer le signal comme constant, alors on retrouve la valeur du signal à l'aide d'un simple intégrateur (le sigma).

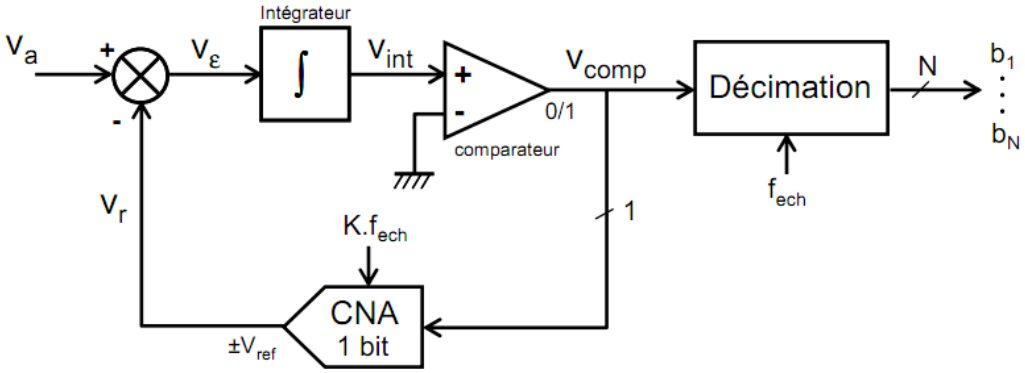


Figure III.7. *Schéma fonctionnel d'un convertisseur sigma-delta*

Un autre aspect du convertisseur sigma delta et le sur-échantillonnage. En effet, pour que ce convertisseur fonctionne correctement il faut sur-échantillonner le signal, c'est-à-dire prendre une fréquence d'échantillonnage supérieure de plusieurs ordres de grandeurs à deux fois la fréquence maximum du signal. Cette action permet de réduire le SNR rapport de signal sur bruit, c'est-à-dire que notre conversion sera moins sensible au bruit.

Le signal Vcomp en sortie du comparateur est sur 1 bit. Le bloc décimation réalise un filtrage passe bas numérique sur le flot de bits envoyé par le comparateur. Il en ressort en nombre binaire sur N bits à la fréquence d'échantillonnage.

Nous avons pris parti d'utiliser ce type de convertisseur car il offre une grande précision de mesure et un faible bruit, et nous a par ailleurs été proposé par M. Pillet lors de notre contact à Géo-Azur.

De plus pour notre étude nous avons besoin d'avoir une dynamique de 10^7 (140 dB), et donc pour obtenir une telle résolution il faut un convertisseur sur N bit avec $2^N \approx 10^7$ d'où N=24. On a donc besoin d'un convertisseur 24 bits. Avec un autre type de convertisseur que le sigma delta la qualité de la linéarité différentielle serait extrêmement dégradée.

Chapitre IV : Traitement des données et interaction avec l'utilisateur

En poursuivant le long de la chaîne d'acquisition des données, nous arrivons en dernier lieu là où l'information est centralisée, organisée, stockée et présentée à l'utilisateur final : l'unité de contrôle. Différents types d'unités sont disponibles sur le marché, et, bien que la carte Raspberry Pi que nous allons présenter ci-dessous nous ait été proposée comme outil de travail, il nous a également été possible de comparer et apporter notre propre solution.

Ainsi, nous aborderons dans un premier temps les différents moyens de traiter des données, avant de nous pencher davantage sur la Raspberry Pi et enfin d'apporter les pistes les plus prometteuses pour la programmation de notre système.

IV.1. Différents outils de traitement numériques

De nombreux articles sur les différents montages à base de micro-contrôleurs et micro-ordinateurs sont disponibles sur internet. Il existe même de bon ouvrages et magasines développant les potentialités des uns et des autres, mais avant cela, présentons les caractéristiques de notre projet.

IV.1.1. Interprétation des signaux numériques du sismomètre

Les signaux numériques transmis par le convertisseur analogique/numérique (ou plus souvent appelé « numériseur ») méritent quelques précisions avant de commencer l'implémentation : comment reconnaitre l'information, comment a-t-elle été codée et comment l'interpréter ?

Flux de données binaires

Les données transmises par le numériseur le sont de manière synchrones, c'est-à-dire qu'elles sont rythmées par une horloge dont la cadence est connue. Ces données sont codées en binaire par « mots », ou groupement de plusieurs bits. Selon le choix du convertisseur, les mots peuvent être de 8 bits, 12, 16, 24 bits etc. et séparés les uns des autres par un acquittement (signe distinctif synchronisé avec l'horloge).

Ces flux respectent des normes, écrites ou plus ou moins admises, qui permettent d'être interprétés par de nombreux systèmes (la majorité d'entre eux). Nous aborderons ces protocoles par la suite.

Tensions et courants

Selon les modules à l'origine du flux de données, les signaux n'ont pas nécessairement la même amplitude : nous trouvons communément des signaux de 0-3,3V mais également de 0-5V. Or un module fonctionnant sous 3,3V ne tolèrera pas de tension supérieure à ses broches, car cela endommagerait les composants. Il peut donc parfois apparaitre indispensable d'utiliser ou de fabriquer un convertisseur 3,3V – 5V afin de les rendre compatibles.

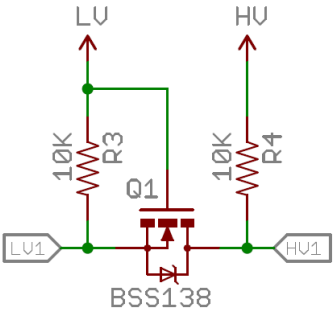


Figure IV.1 *Exemple du convertisseur de tension BSS138*

On observe ci-dessus les tensions LV (*Low Voltage*) et HV (*High Voltage*) correspondant respectivement à l'alimentation 3,3V et 5V. Le transistor NMOS établit un lien entre les deux en reliant son drain à HV et sa grille à LV, via des résistances de tirage. LV1 et HV1 sont les connecteurs mis à disposition pour la conversion.

En fonctionnement normal, LV1 est ramené à LV via la résistance de tirage R3. Ainsi, la grille est la source du transistor sont au même potentiel, soit $V_{GS}=0$ ce qui est inférieur à sa tension de seuil, il est donc bloqué. En conséquence, HV1 est également amené au potentiel HV.

En revanche, si LV1 est à la masse, la tension V_{GS} du transistor lui permet de conduire et d'amener HV1 à la masse.

Enfin, si HV1 est mis à la masse, le drain du transistor l'est donc aussi et la tension V_{GS} dépasse alors à nouveau la tension de seuil. LV1 est donc à la masse par l'effet de conduction du transistor.

Ce circuit, que nous aurons l'occasion de mettre en place au court de notre projet, permet donc une conversion efficace des tensions, tout en étant protégé par l'action du transistor.

Concernant les courants véhiculés dans un flux de données, comme le but n'est pas de commander un moteur ou d'actionner un mécanisme nécessitant une certaine puissance, ces courants sont assez faibles (<20mA). Ils le doivent d'autant plus que des courants trop supérieurs à la centaine de milliampères risquent d'endommager les systèmes d'acquisition en arrivant directement « percuter » le processeur. Les flux sont cependant de plus en plus normalisés et les risques potentiels sont davantage axés sur les problèmes de tension.

IV.1.2. Outils de traitement, que choisir ?

Il existe bon nombre de cartes permettant l'acquisition et le traitement des données, et sans en faire un comparatif exhaustif, nous allons présenter ici les différents types de systèmes.

Les plus connus : Arduino et Raspberry

Lors de nos recherches nous avons pu constater une très riche documentation à propos des cartes Arduino et Raspberry Pi. Ils sont tous les deux plutôt bon marché, d'une taille similaire, et à première vue se ressemble beaucoup. Cependant ils sont très différents : le Raspberry Pi est un micro-ordinateur, fonctionnant avec un système d'exploitation (Linux), et Arduino est un micro-contrôleur sans système d'exploitation. Chacun a donc une fonctionnalité assez particulière.

Figure IV.2 *Le micro-contrôleur Arduino (à gauche) et le micro-ordinateur Raspberry Pi (à droite)*

La gamme Arduino est constituée de micro-contrôleurs programmables. Leur unique fonction consiste à effectuer le programme écrit et inséré par l'utilisateur pour effectuer une tâche qu'il choisit. Il peut lire des informations provenant de capteur, les traiter, et éventuellement les envoyer à un ordinateur ou à une autre sortie (LED, relais, moteurs, écrans, etc.)

Les différents modèles de Raspberry Pi disposent, quant à eux, d'un système d'exploitation complet avec des ports USB, audio, HDMI, Ethernet, comme un véritable ordinateur. En réalité, C'EST un ordinateur. Cependant sa taille miniaturisé, son prix réduit, ses ports GPIO accessibles et son interface Linux font de lui un « cerveau » idéal pour des projets électroniques davantage software que hardware.

En somme, les cartes Arduino seront préférées pour des projets de traitement ou d'automatisation plutôt de type hardware, alors que les cartes Raspberry Pi peuvent avoir des applications plus larges, étant donné la puissance et la richesse de leur système. Pour notre projet, ce type de produit nous permettrait de centraliser l'information provenant du numériseur, de les traiter, les stocker, et les proposer à l'écran à l'utilisateur, chose qui serait plus difficile avec une carte Arduino.

Raspberry Pi et micro-ordinateurs

La première Raspberry Pi, conçue par David Braben, fut commercialisée en 2012 et connu un succès retentissant. Si bien que très peu de temps après, de nombreux concurrents ont investi le marché et nous trouvons maintenant une multitude de micro-ordinateurs de la taille d'une carte de crédit. Parmi eux nous pouvons citer la ClubieBoard, Gooseberry, APC Rock, OLinuxXino, Hackberry, etc.

Un tableau comparatif de ces produits est disponible en Annexe C.

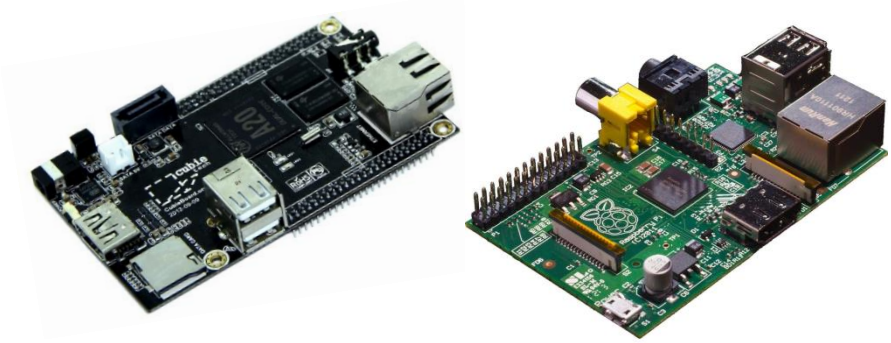


Figure IV.3 *Le ClubieBoard (à gauche) vient concurrencer la Raspberry Pi (à droite) sur le marché des micro-ordinateurs*

Notre regard sur les produits concurrents du marché des micro-ordinateurs :

Le Raspberry Pi est à l'origine d'une dynamique qui est vraisemblablement profitable à l'ensemble du marché. De plus, ces produits étant dans le monde Open Source (libre accès et modification des ressources), rien n'empêche les concurrents d'améliorer les produits des autres, de surenchérir dans les fonctionnalités et les performances, et tout cela au bénéfice de l'utilisateur final.

Pour en revenir à notre projet, où il n'y a pas de commande de moteurs (gourmand en énergie), ni une grande multitude de capteurs à implémenter, il n'est pas nécessaire d'opter pour un produit du type Hackberry (cadencé à 1,2GHz et comprenant une RAM de 1GB). Nous pouvons en effet amplement nous satisfaire des potentialités d'une Raspberry Pi qui reste par ailleurs moins chère.

Enfin, la communauté « Raspinaute » étant particulièrement étendue, il sera sans aucun doute plus aisé de se documenter et de trouver des composants compatibles.

IV.1.3. Potentialités de la carte Raspberry Pi B+ et transposition pour notre projet

En quoi la Raspberry Pi, lors de sa sortie, a-t-elle révolutionné le monde de l'électronique ? Les raisons sont nombreuses mais le plus notable est sans-doute l'innovation et les performances qu'elle apporte dans les montages dit « DIY » ou *Do It Yourself*. Bricoleurs, techniciens, ingénieurs, tous peuvent trouver une utilité de ce produit pour résoudre des problèmes parfois complexes. Si l'on s'approche plus près pour détailler ce qui la compose, on comprend en effet la multitude de potentialités qui la rendent particulière.

Plusieurs modèles de cartes

Il existe à l'heure actuelle 2 types de cartes, ayant chacune évolué d'une génération depuis leur sortie. Nous avons donc la carte A, A+, B et B+. Les versions « + » n'apportent que très peu d'ajouts de fonctionnalités, elles améliorent surtout l'ergonomie, la puissance, et la consommation. Les cartes A et A+, en comparaisons des B et B+, ne disposent pas d'interface réseau, n'ont qu'un seul port USB, et une mémoire RAM moins volumineuse. Par la suite, étant donné qu'il nous a été proposé de travailler sur une Raspberry B+, nous allons voir les différentes caractéristiques intéressantes de celle-ci.

Caractéristiques

Le centre de la carte est occupé par un processeur Broadcom BCM 2835 ARM11 cadencé à 700MHz et sa carte graphique VideoCore IV, sur lesquels est empilée la mémoire RAM de 512Mo.

La seconde puce très visible est le contrôleur réseau (absente de la carte A+ comme précisé ci-dessus).

Le reste de la Raspberry Pi est ensuite largement occupé par les connecteurs, à savoir :

- 4 ports USB, permettant de connecter clavier, souris et bien d'autres périphériques
- Une connectique RJ45 acceptant tout réseau Ethernet de 10 ou 100Mb/s (par de Gigabit Ethernet)
- Un connecteur audio de type jack 3,5mm femelle permettant de brancher un casque ou une paire de haut-parleurs
- Un port HDMI, compatible full HD de 1080p

- Un connecteur GPIO (*General Purpose Input Output*) mâle, de deux rangées de 20 points, permettant d'y connecter des interfaces externes (capteurs, afficheurs, etc.)
- Un connecteur Micro-USB pour l'alimentation en 5V de la carte
- Un connecteur destiné à recevoir une carte Micro-SD

Cette connectique est adaptée à un grand nombre d'utilisations qui peuvent aller de la constitution d'un media-center, d'un serveur web, d'un automate, au traitement de données reçue par un capteur, comme nous allons le voir, notamment via l'utilisation du GPIO qui constitue une véritable « ouverture sur le monde ».

Quant à l'ergonomie, l'utilisateur ne sera pas dépaysé et pourra utiliser sa carte Raspberry Pi avec la plupart des écrans, claviers et souris disponible sur le marché, en les connectant dans leur port respectif.

IV.2. Implémentation sur une carte Raspberry Pi

Comme indiqué précédemment, la Raspberry Pi appartient au monde du logiciel libre, tant sur le plan logiciel que matériel. C'est-à-dire que l'ensemble des programmes, schémas techniques, etc. sont mis à disposition des utilisateurs qui peuvent à leur tour proposer et mettre en ligne des améliorations au service de la communauté. Seulement cela signifie également qu'il n'y a pas de plateforme commerciale ni de support client qui centralise toutes les ressources et il faut alors faire l'effort de documentation nécessaire pour faire fonctionner le micro-ordinateur de la manière la plus judicieuse pour notre propre projet. Sans faire une présentation détaillée des multiples potentialités de la carte, nous allons aborder ci-dessous les éléments essentiels pour notre projet de réalisation d'un sismomètre.

IV.2.1. Système d'exploitation et préparation des données

La carte est livrée « nue » et inutilisable, car elle ne contient pas directement en sortie d'usine de mémoire de masse, ni de système d'exploitation. Nous allons donc devoir nous charger d'installer sur carte micro-SD le système d'exploitation dédié, puis d'insérer le tout dans notre Raspberry Pi.

Préparation de la carte micro-SD

Notre carte mémoire jouera le rôle de disque dur, elle va donc contenir l'ensemble du système d'exploitation (permettant à la Raspberry Pi de démarrer et fonctionner comme un ordinateur normal), mais également l'ensemble des données de l'utilisateur (en fonctionnant comme un système de stockage de masse). Il faut donc prévoir suffisamment d'espace pour que le système ne soit pas saturé, et nous préfèrerons sans doute une micro-SD de 8 Go, bon compromis entre volume de stockage et prix.

La Raspberry Pi utilise un processeur ARM 11, dont le jeu d'instructions n'est pas compatible avec la famille x86 d'Intel, il est donc impossible d'installer Windows ou encore des versions de Linux destinées à un PC. Une distribution libre a donc été proposée par les concepteurs, adaptant Linux à notre carte, et nommée Rasbian Wheezy sous sa dernière mise à jour. Elle est disponible sur le site de la fondation Raspberry Pi à l'adresse www.raspberrypi.org/downloads.

La préparation de la carte consiste alors à extraire l'archive téléchargée sur la carte micro-SD de manière à ce que la Raspberry puisse booter correctement lors de sa mise sous tension. Nous passerons le détail des étapes, bien expliquées et illustrées dans notre livre-support, et noterons simplement qu'il faut prévoir environ 2Go de mémoire sur la carte micro-SD pour le système d'exploitation.

Le système d'exploitation Rasbian Wheezy

Une fois la Raspberry branchée à un écran, un clavier et une souris, et le système d'exploitation installé, on retrouve le fonctionnement bien connu de Linux avec interface graphique.

Un détail supplémentaire, cependant, est susceptible d'attirer notre attention dans le menu *Programmation*. En effet, d'après la fondation Raspberry Pi, la carte a été conçue à l'origine dans un but éducatif, pour permettre d'enseigner la programmation aux enfants. Nous remarquons donc le programme Scratch, interface de programmation ludique très visuelle. Ce ne sera cependant pas notre seul moyen de communication avec notre capteur sismométrique, fort heureusement. Nous disposons également de l'environnement IDLE et IDLE3 permettant de programmer en python, sujet que nous aborderons par la suite dans notre rapport.

Enfin, comme il est fréquent lors de l'utilisation de Linux et notamment pour l'exécution de programmes, nous serons amenés à accéder au mode ligne de commande, ou console, nommée ici *LXTerminal*.

IV.2.2. Gestion des signaux d'entrée et sortie : le port GPIO

Nous l'évoquions précédemment, ce qui différencie la carte Raspberry Pi d'un ordinateur de bureau, outre sa miniaturisation et son prix, c'est la présence de lignes d'entrées et de sorties directement reliées au processeur et rendues particulièrement accessibles. Ces lignes sont regroupées sur le GPIO (*General Purpose Input Output*, ou *entrées et sorties à usage général*), et certaines d'entre elles ont des fonctions bien particulières, comme le montre le schéma ci-dessous proposé par DesignSpark.

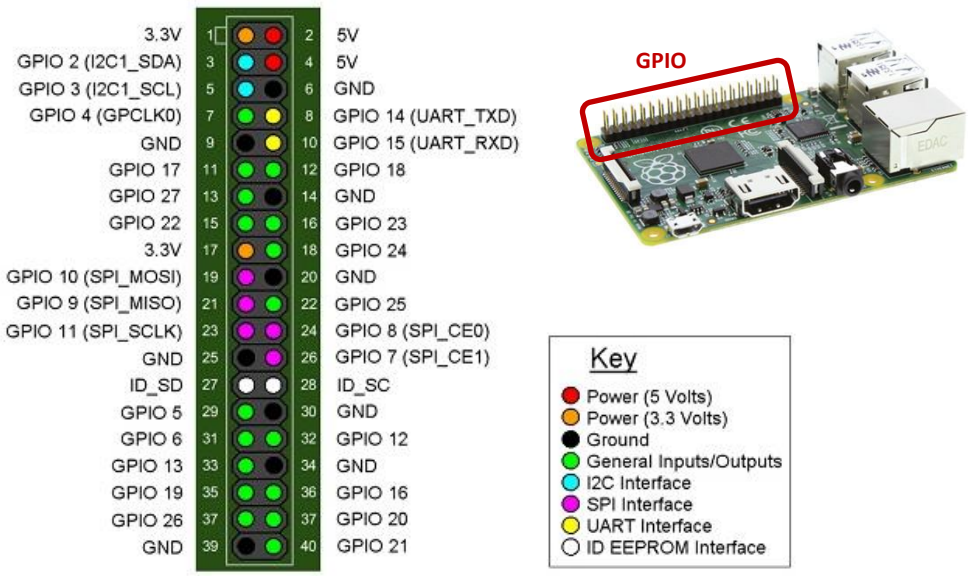


Figure IV.4 *Détails des lignes du port GPIO de la carte Rasperry Pi B+*

On trouve notamment sur le port GPIO :

- De nombreuses lignes pouvant fonctionner en entrée ou en sortie parallèles sans être partagées avec d'autres ressources (en vert clair sur le schéma)

- Un UART, ou port série asynchrone (en jaune), comprenant une ligne d'émission de données (TXT) et une ligne de réception (RXD)
- Un bus I2C, bus série bifilaire utilisant une ligne de données (SDA) et une ligne d'horloge (SCL).
- Un bus SPI, bus série synchrone comprenant une ligne de sortie des données (MOSI), une ligne d'entrée des données (MISO), une horloge (SCLK), et deux lignes de sélection (CE0 et CE1)
- Plusieurs alimentations 5V et 3,3V continues (respectivement rouges et oranges) et des lignes de masse.

Manipuler les tensions analogiques

Nous l'observons ci-dessus, aucune ligne analogique n'est indiquée, il faut donc implémenter soi-même des circuits de conversion.

Par exemple, pour générer un signal analogique à partir d'un signal numérique, il est possible de profiter de la fonction de PWM disponible grâce au GPIO, et d'y implémenter une cellule RC. En effet ce filtre passe-bas permet de disposer de la valeur moyenne du signal PWM : la méthode est simple mais le résultat peut manquer de précision.

Voici ci-dessous un exemple de montage transformant le signal PWM de 50Hz de la Raspberry Pi en tension analogique correspondante :

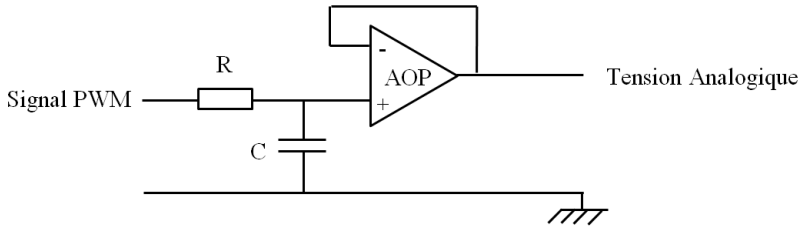


Figure IV.5 *Conversion d'un signal PWM en tension Analogique*

L'amplificateur opérationnel monté en mode « suiveur de tension » permet de rendre la cellule RC indépendantes de la charge en sortie.

Pour le traitement de données analogiques arrivant en entrées du GPIO, il est, de même, assez difficile d'estimer précisément la valeur numérique correspondante. Cependant, pour certains montages, comme avec l'utilisation de capteur de lumière, il peut être suffisant de détecter (1) ou non (0) la présence de lumière. Ainsi, lorsque le capteur analogique envoie un signal dont la tension est supérieur à 2V (seuil de détection d'un niveau logique haut), il sera admis qu'il fait jour, et si le signal est inférieur à 2V, il fait nuit. D'autre technique existent en montage RC et en comparant la constante de chargement de la capacité avec l'instant de détection d'un niveau logique haut, mais ces méthodes restent approximatives.

Nous avons ainsi expliqué l'importance de l'utilisation d'un convertisseur analogique numérique pour « traduire » les données du sismomètre vers notre système d'acquisition

IV.2.3. Les interfaces I2C et SPI

Le dialogue avec le monde numérique est de plus en plus normalisé et utilise essentiellement deux types de bus pour transférer des données synchrones : les bus I2C et SPI. Le choix de l'un d'entre eux dépendra de la compatibilité des systèmes avec lesquels on souhaite dialoguer.

Interface I2C

Le protocole I2C permet le transfert jusqu'à 400 kilobits de données par seconde via une seule ligne, l'autre étant réservée pour l'horloge. De ce fait, pour que le dialogue soit possible entre 2 systèmes ou plus, le codage doit respecter certaines normes :

- Tout d'abord, la transmission d'un message nécessite une indication de début et de fin (ou condition de départ et condition d'arrêt).

- Le message est découpé en mots de 8 bits (un octet), séparés les uns des autres par un « acquittement ».

- Le premier mot contient 7 bits d'adressage (pour sélectionner le destinataire parmi 128 combinaisons possibles) et un bit de lecture/écriture. Si le bit est à 1, l'émetteur (ou maître) va lire le contenu du récepteur (ou esclave), alors qu'il agira en écriture si le bit est à 0.

- A la fin du message, le maître et les esclaves peuvent alors changer de statut et envoyer ou recevoir de nouvelles données.

Ce protocole dispose de librairies python et d'outils de contrôles le rendant facilement utilisable via notre Raspberry Pi.

Interface SPI

La liaison SPI ne fait l'objet d'aucune norme officielle, contrairement au protocole I2C, mais de nombreux circuits y sont compatibles, nous souhaitons donc également en faire une courte présentation dans ce rapport.

La principale différence de la liaison SPI avec la liaison I2C est que la transmission se fait entre un unique maître et un ou plusieurs esclaves. Le transfert des données ne s'effectue plus non-plus par une ligne unique, mais par deux lignes de façon simultanées : l'une ascendante (MISO : *Master In Slave Out*), l'autre descendante (MOSI : *Master Out Slave In*). Enfin, et comme précédemment, une ligne (SCK) permet la synchronisation entre le maître et les esclaves en délivrant un signal d'horloge.

Par défaut le Raspberry Pi est maître, et il n'est pour le moment pas possible de le configurer esclave. Il peut donc transmettre et recevoir des données d'un périphérique esclave sélectionné au moyen des deux sorties CE0 et CE1 (voir ***Figure IV.4*** Détails des lignes du port GPIO de la carte Rasperry Pi B+). Lorsque la transmission est terminée, le maître arrête l'horloge ce qui désactive alors l'esclave.

Après installation des modules nécessaires à la bonne gestion du protocole SPI, puis redémarrage de la Raspberry Pi, il est possible d'utiliser les librairies associées pour programmer le fonctionnement en python.

IV.2.4. Méthodes d'interaction

Maintenant que nous avons abordé les entrées / sorties de la Raspberry Pi ainsi que les protocoles de communication avec certains périphériques, nous allons pouvoir interagir de manière plus libre avec le monde extérieur. Nous allons donc dans cette partie expliquer les outils dont nous disposons pour engager des actions plus ou moins complexes. L'objectif étant de permettre l'interaction entre l'utilisateur et la machine, mais également entre le sismomètre et la machine.

Interaction de base : les boutons poussoirs

Si l'on souhaite implémenter notre carte, nous supposons que la meilleure méthode est de le faire graduellement. Les éléments de communication de base sont les boutons poussoir et interrupteurs, et nous abordons donc ce moyen d'interaction en premier.

Lorsque l'utilisateur agit et que la machine doit interpréter, alors l'action, ou plutôt le signal électrique portant l'action, arrive en entrée de la carte. Nous devons donc prévoir un montage fonctionnant sur une broche parallèle du GPIO configurée comme une entrée (ce paramétrage se fait par la programmation python que nous aborderons dans la prochaine partie du rapport). Qu'il s'agisse d'un bouton, d'un commutateur ou d'une roue codeuse, le schéma de montage reste le même et comme on peut le voir ci-dessous, deux méthodes sont possibles.

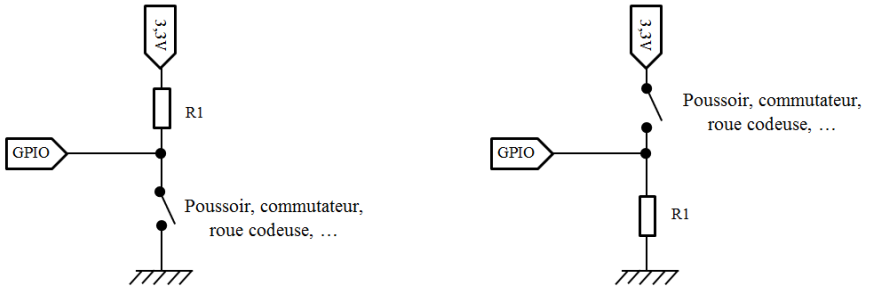


Figure IV.6 *Schéma du montage d'un interrupteur sur une entrée du GPIO*

Le montage d'un interrupteur peut se faire soit entre l'entrée parallèle et la masse, soit entre l'alimentation et l'entrée parallèle. Dans le premier cas, lorsque celui-ci est ouvert, l'entrée du GPIO est ramenée à 3,3V par la résistance de tirage R1, on obtient donc un niveau logique haut. En revanche, lorsqu'il est fermé, il établit un contact entre l'entrée du GPIO est la masse et on obtient un niveau logique bas. Le cas inverse est présenté en partie droite et nous obtenons alors en entrée du GPIO un niveau bas lorsque l'interrupteur est ouvert, et un niveau haut lorsque l'interrupteur est fermé. La résistance utilisée devra dans notre cas être de l'ordre de 50-100kΩ afin de réduire au maximum la consommation.

Ce montage n'évite cependant pas les effets de rebondissement, généré lors de l'action humaine sur l'interrupteur. En effet, au lieu de générer un créneau idéal, l'interrupteur peut lancer un « train d'impulsion » avant d'être stable sur le niveau voulu, ce qui peut parasiter l'exécution logicielle qui découle de cette entrée.

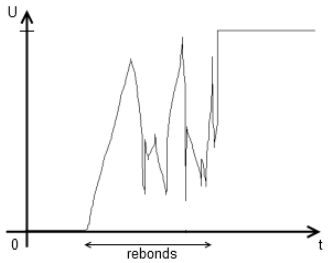


Figure IV.7 *Phénomène de rebondissement d'un interrupteur*

Nous pouvons donc améliorer notre montage en ajoutant un condensateur en parallèle de l'interrupteur, ou choisir de traiter le problème de manière logicielle en adaptant le code d'interprétation de l'action.

Nous aurons l'occasion d'utiliser ce type de montage à des fins de réinitialisation des mesures ou encore d'extinction du système (bouton off).

Affichage des données et ergonomie

L'affichage des données est possible de diverses manières, plus ou moins ergonomique.

La méthode communément utilisée pour l'affichage de données alphanumériques, et de loin la moins onéreuse, est d'avoir recours à un afficheur à cristaux liquides 8bits, connecté à 11 broches disponibles du port GPIO. Cela permet d'apporter des informations textuelles simples à l'utilisateur via un écran pouvant disposer jusqu'à quatre lignes de quarante caractères chacune. Beaucoup plus parlant pour décrire l'état d'un système qu'une simple LED.

L'ergonomie peut cependant être améliorée par l'utilisation d'un écran LCD TFT graphite par exemple (*Thin Film Transistor Liquid Crystal Display*). Il se compose d'une dalle couleur rétroéclairée, parfois tactile, et de manière générale assez répandu pour des tailles de moins de 5 pouces de diagonales. Ces circuits restent assez abordables (moins de 30€) et apportent un gain en confort visuel. Ils se connectent sur les ports SPI du GPIO que nous avons présenté précédemment, et, une fois les librairies python associées installées, peuvent servir de véritable écran intégré au Raspberry Pi. Ce type de produit serait extrêmement intéressant pour notre projet en apportant une portabilité du système qui deviendrait « tout en un ». Cependant, les coûts sont à prendre en compte et nous serons amenés à choisir si une telle option est envisageable.

Enfin, la solution apportant le plus de confort est l'utilisation d'un écran déporté, en le banchant sur le port HDMI de la carte Raspberry Pi. Cela permettra à l'utilisateur de retrouver un environnement connu (celui de Linux) et d'y lancer le programme spécifique au traitement des données du sismomètre, comme sur n'importe quel ordinateur.

IV.2.5. Gestion du temps, synchroniser les données reçues avec l'heure internationale

Dans le cadre du projet, nous serons amenés à associer les signaux provenant du sismomètre avec l'heure internationale. Il deviendra alors possible de comparer les chronogrammes avec d'autres stations pour pouvoir améliorer la localisation d'un foyer sismique, ou encore étudier la nature géologique des sols (une onde se propage plus rapidement si le support est dense).

Mise à jour de l'heure

Dans les ordinateurs actuels, il existe un petit module autonome à très faible consommation permettant de conserver la date et l'heure en temps réel, c'est un module RTC (*Real Time Clock*). Son alimentation est indépendante du reste du système si bien que si l'on retire la batterie de notre PC portable, il sera toujours à l'heure en le remettant en marche quelques jours plus tard.

La Raspberry Pi a été conçue pour réduire au maximum la taille et le coût, il a donc été choisi de ne pas implémenter de RTC et d'utiliser la connexion internet à chaque démarrage pour mettre à jour la date et l'heure. Pour ce faire, elle doit se connecter aux serveurs NTP (*Network Time Protocole*) qui disposent d'une grande précision grâce à une synchronisation avec l'horloge atomique. En l'absence de connexion, l'utilisateur doit manuellement mettre à jour le système.

En somme, pour notre projet, si nous souhaitons un système dont le « timing » des mesures est fiable, il faudra avoir recours à un périphérique externe. Nous avons d'ores et déjà deux solutions qui se présentent à nous : monter un module RTC sur la carte, ou faire appel à un dispositif GPS complet.

Le module RTC externe

Ce type de composant est très répandu sur internet, on le trouve pour moins de 3€ et son encombrement est très faible. En fait, l'élément le plus volumineux est la pile bouton de 3V mais celle-ci est indispensable puisqu'elle lui confère une autonomie quasi-illimitée. Le câblage se fait par l'intermédiaire du GPIO, en utilisant les broches du protocole I2C. A propos de la précision, on trouve dans les caractéristiques techniques (celle du DS1302 par exemple) une déviation possible de 50ppm, soit une erreur de 0,005% pour un quartz de fréquence 32,768 kHz. Cela peut paraitre peu, mais si l'on ne dispose d'aucune connexion internet, cette erreur peut atteindre 2 minutes par mois, ce qui nous parait assez gênant dans notre cas présent.

Le dispositif GPS

Le dispositif GPS est d'une autre dimension que le composant précédent, tant par sa taille que par ces fonctionnalités. Malheureusement, il est également plus cher (une quarantaine d'euros en moyenne), mais il est possible que cette dépense puisse valoir le coup.

En effet, le GPS peut être disponible en périphérique USB, ce qui lui donne une certaine simplicité d'utilisation et de câblage, *plug and play*. Il est également très précis, et ne nécessite aucune connexion internet, la puce GPS intégrée mettant les informations à jour en continu. Enfin, outre sa capacité à donner l'heure, il peut également donner la position à moins de 10m. Cet avantage peut être considérable s'il s'agit de proposer un sismomètre totalement autonome. Connaissant l'heure exacte et la position exacte, il est possible d'obtenir des sismogrammes de grande précision, et de les croiser par la suite avec d'autres stations sismométriques pour obtenir des informations exploitables.

Nous aurons l'opportunité de choisir entre les différentes options présentées précédemment, mais ce choix doit être pris après mûre réflexion car le temps est une donnée essentielle en sismologie.

IV.3. Python est ses potentialités

Nous aurions pu aborder les différents langages de programmation, les comparer afin de choisir celui qui correspond le mieux aux exigences du projet. Cependant, un argument de poids nous conduit vers le choix unanime de Python : la mise à disposition sur internet de nombreuses bibliothèques Python traitant de sismologie, et plus particulièrement de sismométrie. Comme il ne serait pas judicieux de refaire ce qui existe déjà, nous allons travailler à partir de ces librairies et les manipuler au mieux pour mener à bien notre projet.

IV.3.1. Programmation Python sous Linux avec une carte Raspberry Pi

Python est l'un des langages de programmation les plus intuitifs, puisqu'il dispose de règles de syntaxe simples et peu nombreuses. Adieu, donc, les points-virgules en bout de ligne, il faudra en contrepartie être vigilant sur l'indentation qui a ici toute son importance. Il est constamment enrichi de fonctionnalités grâce à des bibliothèques libres (*open source*) dont voici quelques-unes de ses fonctionnalités :

- Créer des interfaces graphiques
- Faire circuler des informations au travers d'un réseau
- Dialoguer d'une façon avancée avec le système d'exploitation

C'est un langage interprété, contrairement à une grande partie d'autres langages, comme le C, qui sont des langages compilés. Dans ce deuxième cas, le programme doit en effet être « traduit » en langage machine pour être rapidement disponible lors de son exécution. Python, quant à lui, est traduit au fur et à

mesure de l'exécution du programme, ce qui, certes, le rend moins rapide, mais permet également de localiser les erreurs à l'endroit où s'arrête le programme.

Ecrire et exécuter un programme

L'environnement de travail pour programmer en Python est directement disponible au lancement de la Raspberry Pi, il a été préinstallé dans le système d'exploitation, et s'appelle IDLE ou IDLE 3, selon qu'il fait référence à la version 2 de Python, ou à la version 3.

Lorsque l'on ouvre l'environnement de développement, on trouve une fenêtre permettant d'exécuter des fonctions en mode ligne de commande. Aucun programme ne peut cependant y être écrit, car il serait perdu à la fermeture. Ainsi, pour rédiger un programme, il faut ouvrir un fichier (via ce même environnement de développement) et l'enregistrer sur la machine pour qu'il puisse être réutilisé à posteriori. Le fichier doit alors contenir un *header* ou ligne d'en-tête, lui permettant d'être reconnu comme programme en langage Python. Nous devons donc prévoir l'insertion de la ligne suivante de manière systématique :

```
# !/usr/bin/env python
```

Une fois le programme écrit, enregistré et testé, il est possible de le rendre exécutable depuis un terminal, en modifiant les permissions à l'aide de la fonction `chmod` :

```
$ chmod a+x programme.py
```

A ce moment, il est possible de lancer le programme avec la commande :

```
$ ./programme.py
```

Cependant si le fichier est stocké dans un sous-répertoire, il faudra retaper l'adresse complète à chaque appel. Pour qu'il puisse être libéré de cette contrainte, de même que de son extension (la ligne d'en-tête permettra au système de savoir de quel type de langage il s'agit) avec les commandes suivantes :

```
sudo cp programme.py /user/local/bin
sudo mv /usr/local/bin/programme.py /usr/local/bin/programme
```

Il est donc maintenant possible de faire un appel à *programme* depuis n'importe quel endroit dans le terminal.

IV.3.2. Des bibliothèques riches en fonctionnalités

Les bibliothèques regroupent des ensembles de fonctions dédiées à une utilisation bien particulière, et qui sont en grande partie mise à disposition par leurs auteurs sur internet.

Interaction avec le GPIO

Pour gérer les entrées et sorties de la carte Raspberry Pi, une bibliothèque est directement intégrée dans l'environnement Rasbian Wheezy, et se nomme RPi.GPIO. Tout programme souhaitant interagir avec le GPIO devra donc l'importer en début de programme avec l'instruction suivante, avant d'en utiliser les différentes fonctions.

```
import RPi.GPIO
```

Certaines fonctions deviennent alors indispensables, par exemple pour affecter une broche en sortie (GPIO.OUT) ou en entrée (GPIO.IN), on utilise la fonction `setup` :

```
RPi.GPIO.setup(16,GPIO.OUT)
```

Pour affecter un niveau logique haut (GPIO.HIGH) ou bas (GPIO.LOW) à une sortie, on utilisera la fonction `output`, et pour lire la valeur d'une entrée on utilisera la fonction `input` (que l'on attribuera le cas échéant à une variable) :

```
RPi.GPIO.output(16, GPIO.HIGH)
signal22 = RPi.GPIO.input(22)
```

Il est également possible de détecter automatiquement un changement d'état, permettant alors de déclencher certaines actions en conséquence. Les fonctions à utiliser sont `add_event_detect` et `event_detected` :

```
RPi.GPIO.add_event_detect(16, GPIO.RISING)
if RPi.GPIO.event_detected(16) :
   action1
   action2
```

Nous ne pouvons pas présenter ici toutes les fonctionnalités, mais celles-ci étant les bases de l'interaction avec les entrées et sorties, il nous apparaissait important d'expliquer très brièvement le mécanisme de programmation que nous allions utiliser.

Programmation du bus I2C

Le protocole I2C n'est pas activé à l'origine par la Raspberry Pi, il faut donc lui indiquer que nous allons utiliser ce module en éditant les fichiers `/etc/modules` et `/etc/modprobe.d/raspi-blacklist.conf`. Nous devons par la suite télécharger les outils nécessaires à la manipulation du bus I2C via la commande suivante dans le terminal :

```
$ sudo apt-get install i2c-tools
```

Pour la programmation de notre bus I2C, nous devrons enfin ajouter à notre programme python la bibliothèque `smbus` associée :

```
from smbus import *
```

Programmation du bus SPI

De la même façon que le bus I2C, lorsqu'on souhaite programmer le fonctionnement du bus SPI, il convient d'activer les modules au sein de la carte Raspberry Pi (même méthode que précédemment), puis télécharger les bibliothèques Python nécessaire au bon fonctionnement du programme. On trouve facilement les bibliothèques « standarts » sur le site de python.org. Enfin, on indique en en-tête de notre programme la dépendance aux fonctions contenues dans la bibliothèque que nous venons de télécharger.

```
from spi import *
```

Temps RTC et GPS

Bien souvent, lorsque les composants sont moins répandus, ou qu'ils n'utilisent pas le protocole habituel, les fabricants ou les fournisseurs proposent eux-mêmes les bibliothèques pythons permettant leur contrôle. Ainsi, pour le circuit RTC à base de DS1302 abordé précédemment, nous pouvons trouver sur le site de HobbyTronics un lien de téléchargement de la bibliothèque `rpi_time`, à utiliser de la même manière que précédemment.

Pour le GPS modèle BU-353S4 évoqué plus haut, les bibliothèques peuvent être téléchargées directement via la commande suivante dans le terminal :

```
apt-get install -y gpsd gpsd-clients python-gps
```

Il suffira ensuite d'appeler la bibliothèque dans notre programme python avec l'instruction :

```
from gps import *
```

Nous avons donc à travers ces paragraphes compris le fonctionnement des bibliothèques, leur richesse en fonctionnalités, et le gain de temps possible dans la programmation en python de systèmes internes ou externes à la Raspberry Pi.

IV.3.3. Le domaine spécifique de la sismologie

Présentation de la bibliothèque ObsPy

Nous ne pourrons mieux présenter ObsPy que Moritz Beyreuther dans *Seismological Research Letters* (2010) :

> *The wide variety of computer platforms, file formats, and methods to access seismological data often requires considerable effort in preprocessing such data. Although preprocessing work-flows are mostly very similar, few software standards exist to accomplish this task. The objective of ObsPy is to provide a Python toolbox that simplifies the usage of Python programming for seismologists. It is conceptually similar to SEATREE (Milner and Thorsten 2009) or the exploration seismic software project MADAGASCAR (www.reproducibility.org).*
>
> *In ObsPy the following essential seismological processing routines are implemented and ready to use: reading and writing data only SEED/MiniSEED and Dataless SEED, XML-SEED (Tsuboi* et al. *2004), GSE2 and SAC, as well as filtering, instrument simulation, triggering, and plotting. There is also support to retrieve data from ArcLink (a distributed data request protocol for accessing archived waveform data, see Hanka and Kind 1994) or a SeisHub database (Barsch 2009). Just recently, modules were added to read SEISAN data files (Havskov and Ottemöller 1999) and to retrieve data with the IRIS/FISSURES data handling interface (DHI) protocol (Malone 1997).*

Nous apprenons à travers ce paragraphe qu'une bibliothèque Python a été conçue pour regrouper et centraliser les différentes actions possibles pour traiter les données d'un sismomètre. Outre les fonctions de compatibilité entre les types de données, elle dispose également de fonction de filtrage du signal numérique, de méthode d'affichage des courbes et d'autres outils de calculs précisés dans la suite de l'article.

Nous retrouvons quelque peu la syntaxe des logiciels de type MatLab, mais à la différence près que les outils ObsPy ne sont pas propriétaires mais, comme la majorité des dispositifs cités dans ce rapport, libre d'accès ou *open source*.

Format des données : le standard SEED

L'IRIS (*Incorporated Research Institution for Seismology*), et plus particulièrement son Centre de Management des Données, collecte les données sismiques provenant d'une grande quantité de stations du

monde entier. Leur objectif est d'archiver et distribuer ces données de manière responsable, dans le but d'améliorer les échanges, croiser les informations et servir la recherche.

Le format SEED (*Standard for the Exchange of Earthquake Data*) a donc été établit dans le but de rendre ce partage plus uniforme, et est devenu un standard international de données sismiques. Il repose sur la compression des données qui pouvaient jusqu'alors poser problème. En effet, avec le développement de l'informatique et la numérisation des données, les stations enregistrant une activité 24 heures sur 24 pouvaient générer plusieurs dizaines de mégaoctets par jour, et être rapidement à cours de stockage, ou encore avoir des difficultés à transmettre les données à une autre station.

La compression existe sous deux formes, l'une engendre une perte d'information (compression d'une image, par exemple, qui peut paraître pixélisée en sortie), et l'autre conserve la même qualité que l'élément d'origine. Concernant les données sismiques, il est absolument indispensable que la conversion n'entraine pas de perte d'information, c'est pour cette raison que le format SEED repose sur un algorithme de compression qualifié de *lossless* (sans perte) et nommé Tunstall Coding. Celui-ci établit un schéma statistique en examinant les données les plus probables, et attribut davantage de caractères aux données les moins probables, ce qui le rend très rapide et précis.

Un autre point d'attention a été apporté pour que ce format soit compatible avec les outils de traitement déjà existants. Les utilisateurs n'ont donc pas besoin de changer leurs habitudes et peuvent dès lors disposer de fichier beaucoup plus souples, plus propices aux échanges.

Nous allons donc dans notre projet travailler à la compatibilité de nos données avec les formats internationaux, et utiliser les outils de traitement adaptés pour proposer à l'utilisateur une expérience de type professionnel.

Conclusion

Nous avons pu voir tout au long de cette bibliographie différents points d'intérêt de notre projet. Tout d'abord nous avons replacé le contexte du projet en rappelant les domaines de travail de la sismologie. Notamment les caractéristiques physiques importantes qui influent sur la sismométrie, comme par exemple le bruit de fond sismique, la mesure de l'amplitude, et les ordres de fréquences qu'il faut mesurer. Nous avons ensuite fait l'étude du sismomètre, les différentes parties dont il est composé mécanique, électrique, numérique. Nous avons pris le parti de choisir un sismomètre de type vertical. Le sismomètre fonctionne grâce au principe d'inertie et d'une masse flottante (le pendule). Une fois que la masse entre en mouvement (détection d'un séisme) le but va être de mesurer l'amplitude du séisme et de pouvoir observer le phénomène sur un ordinateur. De plus, au vu de l'ordre de grandeur des déplacements à mesurer il faut que le ressort associé au pendule ait un faible coefficient de raideur ce qui amène un problème d'oscillation.

Pour pouvoir observer les oscillations dues aux séismes il nous faut pouvoir transformer le déplacement (grandeur mécanique) en une tension ou un courant (grandeur électrique) pour que l'on puisse ensuite convertir en signaux numériques pour que les données soient traitées puis affichées sur un écran.

A ce point de l'étude il faut donc comprendre comment va fonctionner la mesure du déplacement. Pour cela on a expliqué les lois physiques qui régissent le capteur que l'on utilisera sur notre sismomètre. Puis on est revenu sur la définition d'un capteur, et plus précisément sur les capteurs qui nous intéressent les capteurs de déplacements. Dans le cadre de la sismométrie deux types de capteurs à déplacement sont utilisés : capteurs inductifs (celui que nous utiliserons) et capacitifs (présents sur certains sismomètres professionnels).

On a pu voir lors de l'étude mécanique que nous aurions un problème d'oscillation du pendule (qui mettrait trop longtemps avant de revenir au repos). Pour s'absoudre de ce problème, nous avons décidé d'utiliser une boucle de rétroaction incluant un correcteur Proportionnel Intégral Dérivé, dont le but est de réduire le temps des oscillations du pendule. On a donc étudié les effets d'un correcteur PID pour comprendre son fonctionnement.

Dans le but de réduire les coûts, le correcteur PID sera sous forme numérique, ce qui permet aussi des réglages plus faciles que si l'on utilisait un correcteur analogique. Pour convertir les grandeurs électriques en grandeurs numériques on utilise un convertisseur analogique numérique, dont le principe et les principales caractéristique ont été revues au cours de cette bibliographie (Partie III.3.1). Inversement, pour récupérer une grandeur électrique à partir d'une grandeur numérique on utilise un convertisseur numérique analogique. Dans le cadre de notre projet nous avons l'intention d'utiliser des convertisseurs types sigma-delta, dont le principe de fonctionnement a aussi été rappelé.

Pour le traitement de données les deux cartes les plus connus sont la carte Raspberry Pi et la carte Arduino. Cependant il nous a semblé après un comparatif des deux systèmes que la carte Raspberry Pi serait plus adaptée à l'interaction avec l'utilisateur. Nous avons alors développé nos connaissances sur cette carte (partie IV). Pour pouvoir utiliser le plus de fonctionnalités que ce type de carte peut nous offrir. La carte Raspberry est compatible avec le protocole IC2, permettant les flux de données. De plus le langage Python, qui est le langage de programmation de la Raspberry Pi, possède des librairies particulières pour la sismométrie/sismologie.

L'étude des coûts présentée en première partie souligne la viabilité du projet, que nous souhaitons terminer dans les délais que nous nous sommes fixés. Nous espérons ainsi répondre aux attentes de chacune des parties prenantes au projet, l'équipe pédagogique composée de professeurs du

CIV, d’un enseignant chercheur de Polytech, et d’un sismologue de GéoAzur et ainsi que le groupe d’élèves du CIV.

Bibliographie

[Bau'09] F.Baudoin et M.Lavabre, *Capteurs : principes et utilisations*, 2009

[Bel'13] B. Bellamy, *Microcomputeur comparison*, 2013

[Ber'10] A. Beretz, *Mesurer les séismes, la station sismologique de Strasbourg*, 2011

[Bey'10] M. Beyreuther, "ObsPy: A Python Toolbox for Seismology", *Seismological Research Letters*, Vol. 81 Num.3, 2010

[Bla'12] C. Blaess, *SPI sur Raspberry Pi*, Ver.1, 2012

[Bod'14] D. Bodor, "Compatibilité des systèmes", Hackable Magazine, 2014

[Bou'06] D. Bourbon, ''Asservissement et régulation", *Cours d'électrotechnique*, 2006

[Cha'80] P. Chappel, *Géophysique appliquée, Dictionnaire et plan d'étude*, 1980

[Cha'09] E. Charmois, *La Conversion A /N « Single Bit » (Delta-Sigma)*, 2009

[Cod'13] Codeduino, *Arduino vs Raspberry Pi Comparison*, 13

[Cou'73] J. Coulomb et G. Jobert, ''Sismologie et Pesanteur'', *Traité de Géophysique Interne*, 1973

[Esk'13] Eskimon, ''Montage d'un interrupteur sur Arduino'', *Arduino, premiers pas en informatique embarquée*, 2013

[Gee'01] E. De Geest, *Méthodes d'optimisation pour le réglage de contrôleurs PID*, 2001

[Iri'12] I.R.I.S., *Standard for the Exchange of Earthquake Data (SEED), Reference Manual*, 2012

[Lec'09] J. Lecoq, *Etude des différents systèmes de conversion numériques-analogiques et analogiques-numériques*, 09

[New'06] S.-F. Newman, *Seismographic Data Compression*, 2006

[NXP'14] NXP Semiconductors, *I2C-bus specification and user manual*, Rev.6, 2014

[Pil'12] R. Pillet, *De seosmometricum, Manuel de Sismométrie*, 2012

[Ras'14] Raspberry Pi Fundation, *About Raspberry Pi*, 2014

[Raz'94] B. Razavi, *Principles of data conversion system design*, 1994

[Seg'71] M.K. Seguin, ''La Géophysique et les propriétés physiques des roches'', *Les Presses de l'Université LAVAL*, 1971

[Tav'13] C. Tavernier, *Raspberry Pi, Prise en main et premières réalisation*, 2013

[Ver'10] M.Vermelle, ''LVDT Specification'', *Meggitt Sensing Systems*, 2010

Annexes

Annexe A :

La force de rappel proportionnelle à l'allongement du ressort s'écrit :

$$F_{rappel} = k_{rappel} z(t) = Sz(t) \qquad (2.1)$$

Le coefficient *S*, dureté du ressort s'exprime en N/m. La notion de ressort est utilisée dans son sens très général et il pourra être mécanique ou relatif à un champ de force, comme nous le verrons plus loin.

D'autre part, l'attache de cette masse d'inertie n'est pas exempte de frottement. La force de frottement est proportionnelle à la vitesse de déplacement de la masse et s'écrit :

$$F_{frottement} = k_{frottement} \frac{\partial z}{\partial t} = k_{frottement} \dot{z}(t) = R\dot{z}(t) \qquad (2.2)$$

Le frottement est schématisé sur la figure par un piston dans un cylindre, comme un amortisseur d'automobile. Le coefficient de frottement s'exprime en N/m/s. Pour les mouvements plutôt lents, comme en sismologie, les forces de frottement sont simplement prises proportionnelles à la vitesse de déplacement de la masse par rapport au châssis.

Une accélération de la masse d'inertie résulte de la somme des forces appliquée : force extérieure, force de frottement et force de rappel du ressort. On écrit :

$$M\frac{\partial^2 y}{\partial t^2} = F_{extérieure} - R\frac{\partial z}{\partial t} - Sz \qquad (2.3)$$

$$M\ddot{y}(t) = F_{extérieure} - R\dot{z}(t) - Sz(t) \qquad (2.4)$$

Nous recherchons le déplacement du sol noté $x(t)$. Ces mouvements sont reliés entre eux par l'équation suivante : $z(t) = y(t) - x(t)$. Le mouvement de la masse par rapport au châssis est égal à son mouvement absolu moins le mouvement du sol.

Nous obtenons : $M\ddot{z} + R\dot{z} + Sz = F_{extérieure} - M\ddot{x}$ (2.5)

La partie gauche de l'équation représente le comportement du pendule au travers d'une équation différentielle du deuxième ordre et la partie droite représente les stimulations extérieures qui vont agir sur le pendule : soit une force extérieure directement appliquée à la masse, soit une accélération du sol. En sismologie nous désirons mesurer les accélérations du sol mais cette équation montre que pour l'étude du pendule nous pourrons utiliser deux voies : une table vibrante qui va reproduire les accélérations du sol ou une force extérieure directement appliquée sur la masse.

Nous avons écrit la mesure de l'accélération du sol *x(t)* en fonction d'une équation différentielle du deuxième ordre de variable *z(t)* qui représente une grandeur facilement mesurable : c'est simplement le mouvement de la masse par rapport au châssis (lié au sol). La mesure de l'accélération du sol n'est pas aussi simple qu'une mesure de déplacement habituel entre un élément fixe et un élément mobile mais elle est parfaitement réalisable moyennant quelques concessions que nous devons faire aux équations différentielles.

Etude de l'équation précédente réduite au pendule seul : $\ddot{z} + {}^{R}\!/_{M}\,\dot{z} + {}^{S}\!/_{M}\,z = 0$

Nous allons étudier cette équation d'un point de vue purement mathématique.

2.4.1 considérons que ce pendule subit des frottements négligeables, l'équation se réduit à : $\ddot{z} + {}^{S}\!/_{M}\,z = 0$. Pour résoudre cette équation différentielle, nous recherchons une solution de la forme : $z(t) = A(\omega)\exp(j\omega t)$ .Nous écrivons :

$-A\omega^2\exp(j\omega t) + {}^{S}\!/_{M}\,A\exp(j\omega t) = 0$, ce qui impose : $\omega^2 = {}^{S}\!/_{M}$. Ce résultat montre que, soumis à toutes les fréquences, ce pendule ne restitue qu'une seule fréquence fixée par les caractéristiques mécaniques du pendule : la masse et la raideur du ressort. On pose : $\omega_0 = 2\pi f_0 = {}^{2\pi}\!/_{T_0} = \sqrt{{}^{S}\!/_{M}}$ qui est la pulsation propre du pendule.

Un pendule parfait, sans frottement, est un filtre mono-fréquentiel parfait qui ne restitue, après une excitation contenant toutes les fréquences, que sa fréquence propre. Elle est fixée par la raideur du ressort et la valeur de masse d'inertie.

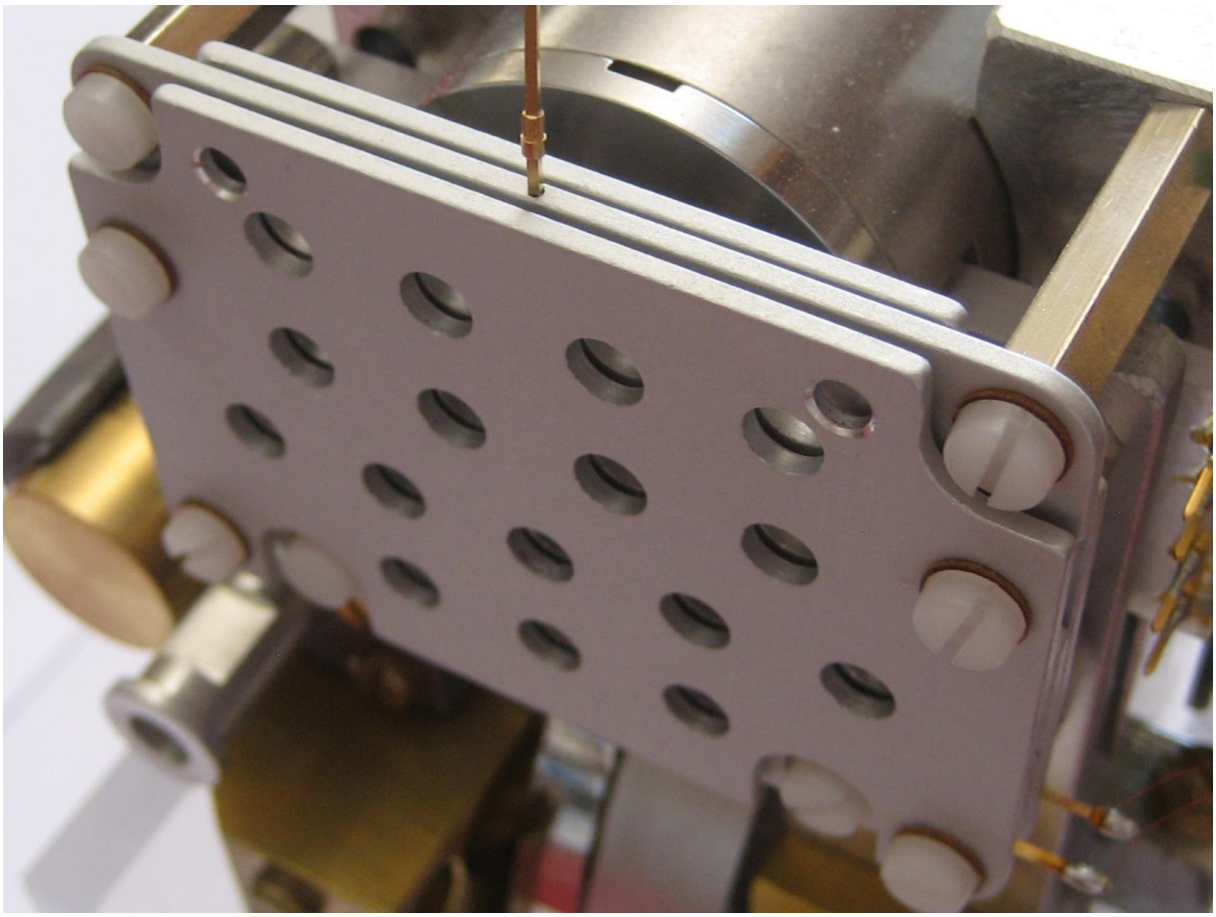
Ce pendule en oscillation va décrire une courbe sinusoïdale mono fréquentielle non amortie.

2.4.2 La prise en compte les forces de frottement montre que la réponse du pendule est une sinusoïde amortie. On démontre que l'enveloppe de cette sinusoïde amortie est de la forme $\exp(-\beta\omega_0 t)$ avec $2\beta\omega_0 = {}^{R}\!/_{M}$. L'équation finale s'écrit :

$$\ddot{z} + 2\beta\omega_0\dot{z} + \omega_0^2 z = 0 \qquad (2.6)$$

L'amortissement du pendule β est relié au facteur de qualité de la suspension par $Q = {}^{1}\!/_{2\beta}$.

Annexe B : Image d’un capteur capacitif réel :



Annexe C : Tableau comparatif de différents micro-ordinateurs et micro-contrôleurs du marché, proposé par Techwatch.

Board name	Price	Version	Licence	Community	Docs	Processor	Clock Speed	SoC	GPU	RAM	Memory	Max Memory	GPIO	Analog In	Analog Out	USB	USB host	Ethernet	Wifi	HDMI	VGA	Video out	SD	µSD	Audio out	Audio Line In	Mic. In	Sata	Infrared	Linux	Android	Android Play Store
Arduino Uno	$27 20€	Rev 3	(c) BY-SA			ATmega328	16Mhz	/	/	2KB	32KB	32KB	14	6	1	1	1															
Arduino Due	$53 39€	Rev 1	(c) BY-SA			AT91SAM3X8E	85MHz	/	/	96KB	512KB	512KB	54	12	2	1	1															
Raspberry Pi, model B	$35 26€	Rev 2				ARM11	700MHz	Broadcom BCM2835	VideoCore IV	512MB	None	32GB via SD	26	12	2	2	1															
CubieBoard	$49 36€	Rev 1				ARM Cortex-A8	1GHz	Allwinner A10	ARM Mali-400	1GB	4GB NAND flash	32GB via SD	96			2	1															
Gooseberry	$63 46€	Rev 1				ARM Cortex-A8	1GHz	Allwinner A10	ARM Mali-400	1GB	4GB NAND flash	32GB via SD	None			0	1															
APC Rock	$79 59€	Rev 1				ARM Cortex-A9	800MHz	Wondermedia Prizm WM8950	ARM Mali-400	512MB	4GB NAND flash	32GB via SD	24			2	1															
A13 OLinuXino Wifi	$74 55€	Rev. E	(c) BY-SA			ARM Cortex-A8	1GHz	Allwinner A13	ARM Mali-400	512MB	4GB NAND flash	32GB via SD	68/74			3	1															
A10 OLinuXino		TBA				ARM Cortex-A8	1GHz	Allwinner A10	ARM Mali-400	1GB	4GB NAND flash	32GB via SD	132			2	1															
Hackberry A10	$65 48€	Rev 1				ARM Cortex-A8	1.2GHz	Allwinner A10	ARM Mali-400	1GB	4GB NAND flash	32GB via SD	None			2	0															