الجمهورية الجزائرية الديمقراطية الشعبية
REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE

وزارة التعليم العالي والبحث العلمي
MINISTERE DE L'ENSEIGNEMENT SUPERIEUR ET DE LA RECHERCHE SCIENTIFIQUE

المدرسة الوطنية العليا للفلاحةالحراش -الجزائر-
ECOLE NATIONALE SUPERIEURE AGRONOMIQUE EL-HARRACH -ALGER-

Thèse

En vue de l'obtention du diplôme de Doctorat en sciences agronomiques

Thème

Conception d'un système de capteur pour l'analyse des propriétés physico-mécanique du sol en temps réel

Présenté par : BOUDHAR Lies

Devant le jury :

Président :	**M. AIDAOUI Abdellah**	(Professeur - ENSA-El-Harrach)
Directeur de thèse :	**M. AMARA Mahfoud**	(Professeur - ENSA-El Harrach)
Examinateurs :	**M. MOULAI Hocine**	(Professeur - USTHB-Bab-Azzouar)
	M. YAZID Krim	(Professeur - USTHB-Bab-Azzouar)

Année universitaire : 2017 /2018

Remerciements :

من شُكر الله شُكر عباد الله الذين جعلهم الله سببًا في مساعدتي، فمن عَجِز عن شُكر النَّاس، فهو عن شُكر الله أعجز.

Je remercie en premier lieu الله tout puissant de m'avoir accordé la puissance et la volonté pour achever ce travail.

Je voudrais aussi remercier grandement mon directeur de thèse, Monsieur Mahfoud Amara Professeur à l'école nationale supérieure agronomique, pour ses orientations, sa confiance, et sa patience qui ont constitué un apport considérable sans lequel ce travail n'aurait pas pu être mené à bon port.

Je suis très reconnaissant envers Monsieur AIDAOUI Abdellah, Professeur à l'école nationale supérieure agronomique, d'avoir accepté de présider le jury de cette thèse.

Mes remerciements et ma gratitude s'adressent à Monsieur MOULAI Hocine Professeur à l'USTHB bab azzouar qui a aimablement accepté de juger ce modeste travail et également à Monsieur YAZID Krim Professeur à l'USTHB bab azzouar que je le remercié chaleureusement.

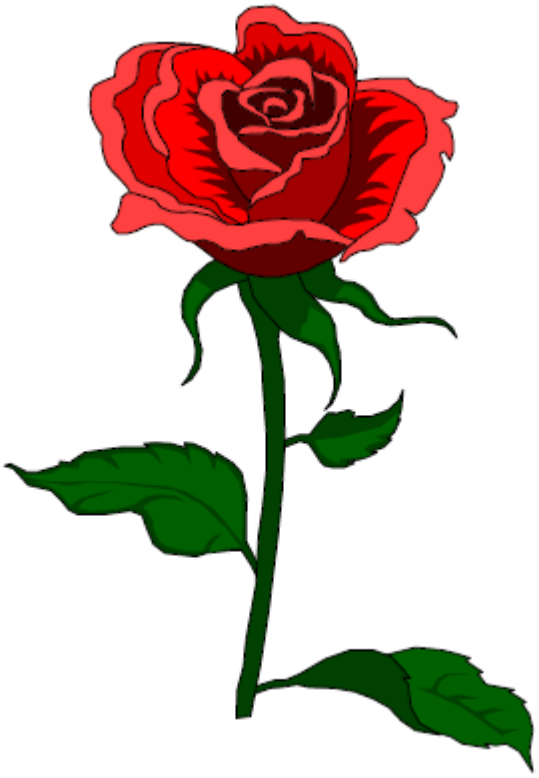
Mes remerciements s'adressent également à mes collègues et amis de l'école nationale supérieure agronomique et plus précisément amine et zaki, sans oublie les amis en dehors de l'ENSA.

Je remercie chaleureusement, mes parents, ma famille, mes frères et sœur. Je remercie énormément ma femme pour sa patience et son soutien durant la période de la fin de thèse.

Je remercie tout le monde qui m'a posé cette question récurrente, « quand est-ce que tu la soutiens cette thèse ? », bien qu'angoissante en période fréquente de doutes, m'ont permis de ne jamais dévier de mon objectif final.

Dédicace :

Je dédie cette thèse… ?



A ma très chère mère,

A mon très cher père,

A ma très chère épouse,

A mes très chers (es) enfants,

A mes frères et sœur,

A mes beaux-parents,

A mes beaux-frères et sœurs,

A mes amis.

Table des matières

LISTE DES FIGURES

LISTE DES TABLEAUX

LISTE DES ACRONYMES ET DES ABRÉVIATIONS

AC-DC :	alternating current - direct current
ADC :	analog-to-digital converter
AREF :	analog Reference
ARM :	Advanced RISC Machine
AVR :	Advanced Virtual RISC
CAN :	convertisseur analogique-numérique
CC :	courant continu
CEC :	Cation-exchange capacity
CI :	Cone index
CPU :	central processing unit
DAC :	digital-to-analog converter
DSPIC :	digital signal PIC
EEPROM :	Electrically-Erasable Programmable Read-Only Memory
FTP :	File Transfer Protocol
GNU :	Gnu's Not Unix
I2C :	Inter-Integrated Circuit
ICSP :	In Circuit Serial Programming
IDE :	integrated development environment
kB :	kilo byte (est une unité d'information)
LED :	light-emitting diode
LR :	lime requirement
MCU :	microcontroller unit
MEMS :	Microelectromechanical systems
MIPS :	Millions of instructions per second
MISO :	Master Out-Slave In
MOSI :	Master In-Slave Out
NIR :	near infrared
OLED :	Organic Light-Emitting Diode
PCB :	printed circuit board
PH :	potentiel hydrogène (Unité de mesure d'acidité)
PIC :	Peripheral Interface Controller or Programmable Integrated Circuit
PLL :	Phase-locked loop
PWM :	Pulse Width Modulation
RAM :	random-access memory
RC :	resistor–capacitor circuit
RISC :	Reduced instruction set computer
ROM :	read-only memory
SCK :	Serial Clock
SCL :	Serial Clock
SDK :	Software Development Kit
SDA :	Serial Data OUT
SPI :	Serial Peripheral Interface
SS :	Slave Select

SSP : Synchronous Serial Port
TDR : Time Domain Reflectometer
TFT : Thin-film transistor
TTL : Transistor-Transistor Logic
UART : universal asynchronous receiver-transmitter
USART : universal synchronous and asynchronous receiver-transmitter
USB : Universal Serial Bus
USDA : United States Department of Agriculture
VIS : visible

INTRODUCTION GENERALE

Le tracteur est lié directement à l'histoire des machines agricoles, et est intéressant, car il met l'accent sur la forme de la mécanisation la plus ancienne et la plus décisive pour l'évolution de l'humanité. Le tracteur reste le symbole de la mécanisation agricole moderne ; il reste parmi les phénomènes technologiques les plus importants. Les développements technologiques de ces dernières années, ne sont pas un changement technique mais une mutation technologique. La technologie du tracteur est à la pointe, les outils de travail du sol ont connu le même processus.

Le tracteur est utilisé dans la plupart des travaux au niveau des exploitations agricoles. Il prend en charge tous les travaux de l'itinéraire technique pour la préparation du lit de semence. Il est le plus grand consommateur d'énergie surtout pendant l'opération de labour, la diminution de cette consommation d'énergie nécessite des connaissances en temps réel des caractéristiques mécaniques et physiques du sol pour agir instantanément sur les opérations agricoles au niveau du tracteur.

Le travail du sol qui a pour but, l'ameublissement du sol, la préparation du lit de semence, le contrôle des mauvaises herbes, modifie la résistance du sol en la réduisant et la porosité en l'augmentant, il modifie aussi l'humidité ; donc en général, il modifie les propriétés mécaniques et physiques du sol. Le compactage du sol peut limiter fortement la croissance des racines et l'infiltration de l'eau, ce qui entraîne une baisse en rendement quantitatif et qualitatif. La résistance du sol à la pénétration verticale peut aider à identifier les zones où les caractéristiques physiques du sol ont un impact négatif sur le rendement d'une culture. Toutefois, selon (Ayers and Perumpral, 1982; Morrison and Bartek, 1987; Hummel, Ahmad and Newman, 2004.). La résistance du sol à la pénétration verticale est fonction de l'humidité du sol, de la porosité et le type de sol ainsi que le compactage et la vitesse de pénétration du cône. L'instrument classique et standard pour mesurer la résistance du sol à la pénétration verticale est le pénétromètre à cône. La mesure de la résistance du sol à la pénétration verticale par un pénétromètre à cône est nommée indice de cône (CI) et est ponctuelle. L'indice de cône est utilisé comme un indicateur important pour le compactage du sol et le développement

racinaire des cultures, ni au moins il faut réaliser plusieurs mesures qui demandent beaucoup de travail pénible et fatigant.

Dans le cadre de l'agriculture numérique. Le développement d'un système de capteur pour la mesure en temps réel, et en continu, des propriétés mécaniques et physiques du sol, semble très intéressant, car il nous fournit un outil valable pour obtenir des informations peu coûteuses et rapides, afin de prévoir dans le contexte de l'agriculture de précision, une bonne gestion de l'état actuel et futur du sol, un schéma de développement racinaire des cultures, et un gain en énergie pendant les déférentes étapes du travail du sol. Mais l'énorme difficulté à obtenir les caractéristiques d'un sol agricole rapidement à moindre coût et in situ reste une des plus grandes limitations de l'agriculture de précision. Nombreux chercheurs et fabricants ont tenté de développer des capteurs pour les mesures in situ et en temps réel pour mesurer les propriétés mécaniques et physiques d'un sol agricole. Ces capteurs sont fondés sur l'électrique l'électromagnétique, l'optique, la radiométrique, la mécanique, l'acoustique et le pneumatique. Alors que seuls les capteurs électriques, électromagnétiques et optiques sont largement utilisés jusqu'à présent. Plusieurs travaux ont été faits dans ce domaine, soit pour automatiser les opérations de travail du sol selon l'état du sol afin de minimiser les dépenses d'énergie, ou bien pour avoir une idée future à la destination de ce sol.

L'objet de la thèse est le développement d'un système de capteur pour la mesure en temps réel, et en continu, de la résistance mécanique du sol agricole.

Par évidence, selon (G. Jahns, et H. Speckmann, 1999.) une agriculture saine et durable sans électronique est inconcevable aujourd'hui, les systèmes électroniques sont utilisés pour réduire les intrants agricoles, protéger l'environnement, augmenter le revenu agricole et produire des produits de haute qualité. Alors ce système de capteur se base sur plusieurs études faites sur le sol et le travail du sol et utilise plusieurs nouvelles technologies à savoir les capteurs les microcontrôleurs, l'électronique et la programmation en langage C et il est embarqué sur un tracteur.

Trois contributions ont été traitées : sélection et utilisation des modèles traitant les propriétés physico-mécaniques du sol, conception et réalisation d'un système électronique de mesure et enfin écriture des programmes avec une application Android qui gère tout le système de capteurs. Le manuscrit est divisé en deux parties, chaque partie est divisée en chapitres.

La première partie est divisée en 7 chapitres, c'est un état de l'art écrit sous forme d'une étude bibliographique. Après une petite introduction elle commence par des définitions, liste des travaux dans le domaine de la mesure de la résistance mécanique du sol agricole, puis une description des outils et des moyens existant pour le développement du système et se termine par une conclusion bibliographique.

La deuxième partie est divisée en 5 chapitres. Après une petite introduction elle présente les modèles de Brixius, et comment on les utilise pour déterminer la résistance pénétrométrique du sol, suivi par le chapitre développement matériel du système, dans lequel on étudie comment la partie matérielle du système est conçue et fonctionne, et décriait l'interconnexion entre les différentes cartes électroniques du système. Dans les trois autres chapitres on étudie comment la partie logicielle du système est conçue et fonctionne. Les programmes sont en open source.

Et on termine par une conclusion générale et des perspectives.

Première partie :

ETUDE BIBLIOGRAPHIQUE

L'objectif de cette partie est d'introduire au cours de quelques paragraphes introductifs, les principes scientifiques et technologiques des notions théoriques et expérimentales sur le sujet de recherche, afin de familiariser le lecteur avec le sol et ses paramètres mécaniques et physiques, comment rendre ces paramètres mesurables dans le cadre de l'agriculture de précision et de l'agriculture numérique à travers quelques travaux de recherche. On termine avec les moyens technologiques et techniques qui nous permettent de développer un système de mesures pour cartographier simultanément la teneur en eau du sol et la résistance mécanique du sol pour les objectifs de travail du sol et la conservation d'énergie.

I. LE SOL AGRICOLE

I.1 Définition générale du sol

Le sol fait partie intégrante des écosystèmes terrestres et constitue l'interface entre la surface de la terre et le socle rocheux. Il se subdivise en couches horizontales successives aux caractéristiques physiques, chimiques et biologiques spécifiques. Il a également différentes fonctions. Du point de vue de l'histoire et de l'utilisation des sols ainsi que d'une perspective écologique et environnementale, le concept de sols embrasse également les roches poreuses sédimentaires, les autres matériaux perméables, en plus de l'eau qu'ils contiennent et des réserves d'eau souterraine. (Winfried E.H. Blum, 2001 in Camuzard J.-P.).

I.2 Définition pédologique du sol

Le sol est un milieu vivant, complexe, sensible aux différentes contraintes à savoir l'action de l'homme et des sociétés, en tant que contrainte majeure imposée aux écosystèmes. Le sol est une zone d'échange entre la biosphère (milieu biologique) et la lithosphère (milieu physique). Il est le résultat d'une pédogenèse, un processus de formation et de différenciation dépendant étroitement des processus physico-chimiques qui contrôlent l'altération de la roche mère, dont le moteur principal serait l'action des agents climatiques. Selon son utilisation il possède différentes fonctions.

I.3 Définition agrologique du sol

Le sol agricole est la partie de la couche superficielle de l'écorce terrestre qui, grâce à sa structure meuble et sa composition physico-chimique, est en mesure d'assurer un développement normal des végétaux cultivés.

I.4 Le sol est une interface fragile

L'interface sol reste encore et restera longtemps indispensable à la satisfaction des besoins nutritionnels croissants de l'Humanité (malgré l'émergence des cultures hors-sol dont la consommation énergétique est immense). Mais, plus important encore, le sol est indispensable à la survie de la plupart des écosystèmes terrestres donc au maintien de la biodiversité. Or l'interface sol est sans cesse l'objet de contraintes nouvelles au niveau de l'anthroposphère : certes ces contraintes entretiennent une certaine dynamique des processus nécessaires au maintien des potentiels (ne serait-ce que celui de fertilité).

Cependant il existe des seuils qu'il convient de ne pas franchir. En effet certaines pratiques anthropiques entraînent des modifications irréversibles et à terme l'altération des fonctions, la disparition des sols ou leur stérilisation : l'érosion, l'altération quantitative et qualitative du niveau et du fonctionnement du réservoir en éléments nutritifs, l'acidification, la salinisation sont autant de processus engendrés par les méthodes modernes d'exploitation de la ressource sol. (Stengel et Gelin, 1998)

I.5 La formation des sols

Un sol est le résultat d'une altération superficielle (figure 1) d'une roche mère qui est le résultat de processus physiques (gel, pénétration des racines...) qui fragmentent la roche, et de processus chimiques (action des eaux chargées d'acides) qui dissolvent les calcaires et hydrolysent les minéraux silicatés pour engendrer des complexes d'altération (argile, oxydes de fer, sels...) cimentant les grains résultant de la précédente fragmentation. Il est aussi le résultat d'un enrichissement en matières organiques issues des êtres-vivants, du fait de la décomposition de la litière par des organismes décomposeurs. Trois facteurs entrant en jeu dans la formation d'un sol :

- **La roche mère :** par ses propriétés physiques et sa composition chimique a une influence directe sur la nature et la rapidité de l'évolution d'un sol

- **Les végétaux :** fournissent l'essentiel en matière organique présente dans le sol, et influencent aussi son évolution
- **Le climat**, affecte les deux facteurs précédents, par la température en ce qui concerne l'altération de la roche mère, et les précipitations pour les phénomènes de migration se déroulant au niveau du sol.

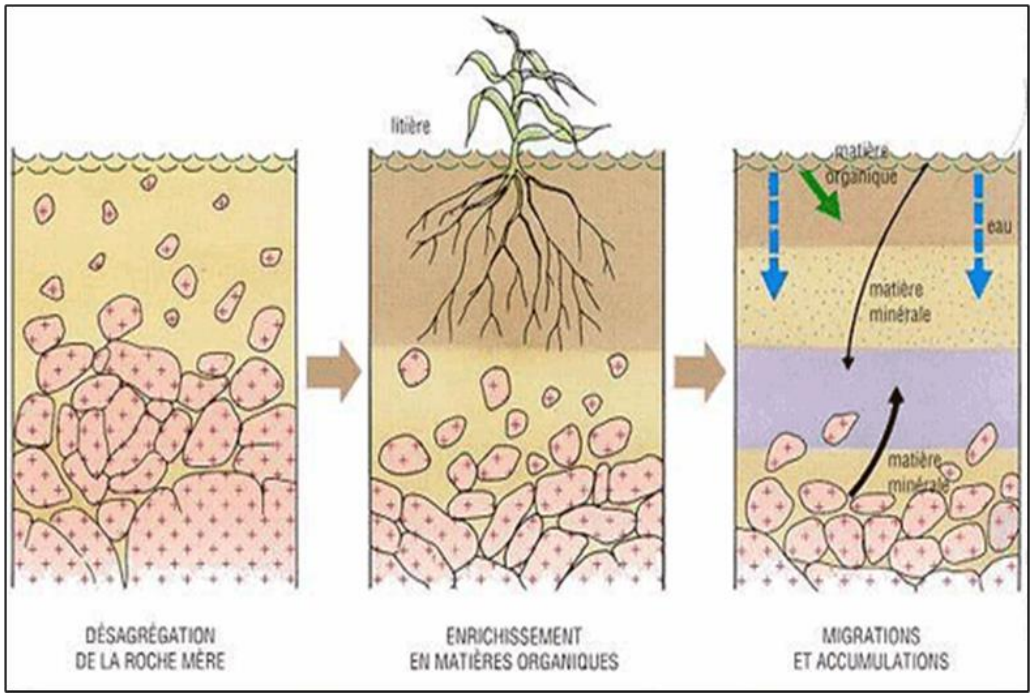


Figure 1 : Les différentes étapes de la genèse d'un sol (Rihani, 2012)

I.6 Caractéristiques et propriétés physiques du sol

Le sol est un matériau non homogène (figure 2), il est un mélange de particule solide, liquide, gazeuse, d'humus et d'êtres vivants ou morts. La phase solide du sol comporte les minéraux et les substances organiques inertes, mais aussi les êtres vivants. La phase liquide, quasi exclusivement aqueuse qui en plus de l'eau, contient l'ensemble des substances et gaz dissous et jouent un grand rôle dans les fonctions du sol (nutrition, réservoir et filtre de certains éléments...). Enfin la phase gazeuse en équilibre avec la phase liquide constitue « l'air » du sol, dont la composition est assez différente de celle de l'air atmosphérique, avec lequel il existe de nombreux échanges. La teneur en CO_2 est manifestement plus élevée (0,5 à 5%) contre 0,035% dans l'atmosphère, mais la teneur en oxygène nécessaire à la respiration des organismes vivants dans le sol (racines, champignons, vers de terre, etc....) est parfois plus basse.

Selon Monnier et Stengel (1982) in Amara (2007), au plan physique, le sol est un

système poreux, caractère résultant de l'organisation de ses constituants solides à différents niveaux. Ces deux auteurs distinguent deux niveaux : le premier est qualifié de textural : dans ce cas, si l'on considère une masse de terre à une teneur en eau donnée, le volume poral résultant de l'arrangement des constituants élémentaires, est caractéristique du matériau. C'est à dire qu'il est lié à la nature de ces constituants et aux conditions énergétiques de leur mise en place au cours de la pédogenèse, on parle alors de porosité texturale (Fies, 1971) in Amara (2007). La mesure de la densité texturale se pratique sur les échantillons soumis à une histoire hydrique ou, en établissant la courbe de retrait de petits agrégats (Fies et Stengel, 1981) in Amara (2007).

Le deuxième niveau qualifié de structural, est constitué par les vides ménagés entre les éléments structuraux résultant de l'agrégation des assemblages élémentaires. Ce niveau comprend : les fissures créées par le retrait du matériau lors de sa dessiccation, les galeries d'origine biologique et les défauts d'ajustement entre les éléments structuraux en liaison avec leur forme. On parle alors de porosité structurale.

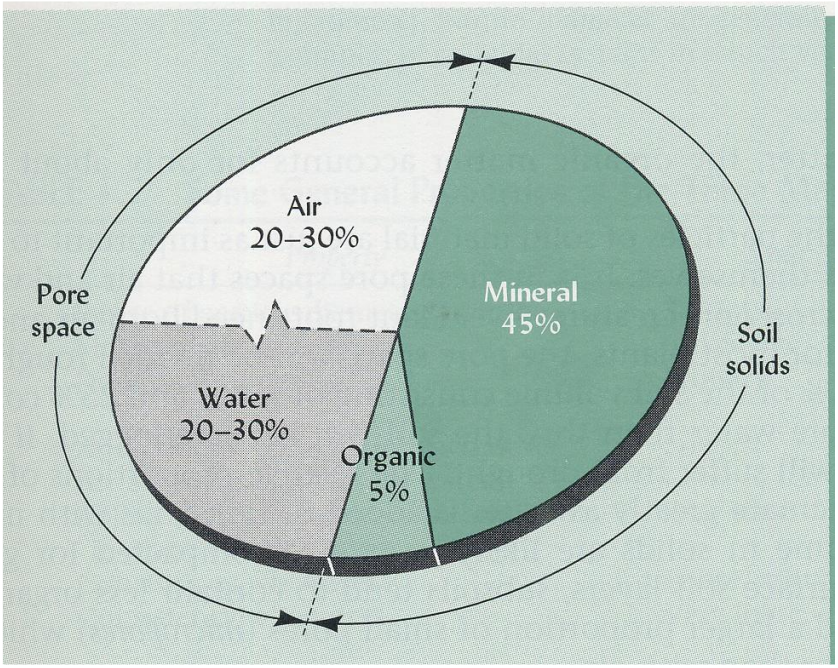


Figure 2 : Répartition moyenne en % des différentes phases du sol (Weil et Brady, 2008)

I.6.1 Texture du sol

On appelle texture (tableau 1) du sol la résultante de la répartition granulométrique du mélange de ses constituants en terres fines et grossières dont les pourcentages varient d'un sol à l'autre. C'est donc la proportion entre les petites particules, les argiles, les particules de taille moyenne, les limons, et particules de grande taille, les sables (dont le diamètre reste tout de même inférieur à 2 mm).

Note : les particules les plus intéressantes en agriculture sont les terres fines.

Tableau 1: Echelle granulométrique de la texture du sol

Terre fine					Terre grossière	
Argiles	Limons fins	Limons grossiers	Sables fins	Sables grossiers	Graviers	Cailloux
< 2 µm	2 - 20 µm	20 - 50 µm	50 - 200 µm	0.2 - 2 mm	2 - 20 mm	> 20 mm

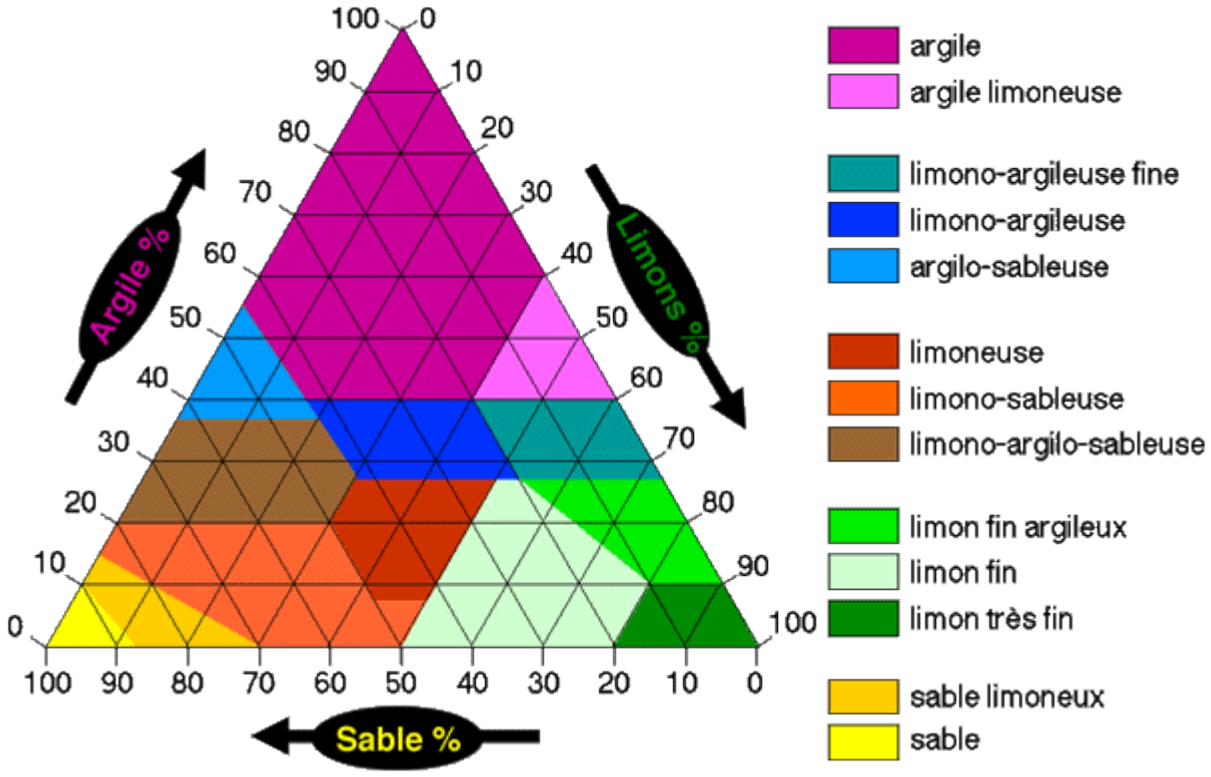


Figure 3 : Triangle textural USDA

La texture est regroupée en diverses classes, comme indiquée dans le triangle de texture standard (figure 3), de sorte que le total de toutes les fractions sera toujours égal à 100 %. Le sol le plus favorable n'est ni trop sableux (qui sera moins fertile et perméable) ni trop argileuse (lourd à travailler et a tendance à être imperméable), mais quelque part entre les deux.

I.6.2 Structure du sol

HENIN (1976) définit la structure du sol comme le mode d'arrangement des différents éléments qui constituent le sol et la conséquence de cet assemblage à un moment donné.

L'état structural du sol est un facteur limitant à toute pratique agricole.

Les éléments du squelette sont associés, par l'intermédiaire de la phase argileuse, de la matière organique et d'autres liants tels que les oxydes de fer et la calcite pour forme des agrégats qui est l'élément structural de base. La forme, la taille et la disposition des agrégats caractérisent la structure du sol, qui doit être à la fois stable et poreuse.

Les classifications et les descriptions morphologiques pour décrire la structure des sols sont tous basées sur la taille et la forme des éléments structuraux.

I.6.3 Les différents types de structure

I.6.3.1 Structure particulaire :

Les éléments sont entassés sans aucune liaison. C'est le cas des sols très sableux peu pourvus en argile. C'est une structure défavorable : risque de battance, sol filtrant.

I.6.3.2 Structure compact :

C'est le cas des sols argileux ou l'argile n'est pas maintenue constamment floculée. En condition humide, elle se disperse dans l'eau, n'assurant plus la liaison des sables et elle devient imperméable. Structure très défavorable (asphyxie, travail très difficile).

I.6.3.3 Structure grumeleuse :

Formation d'agrégats stables du fait que l'argile est maintenue constamment floculée. C'est le type de structure recherchée (sol facile à travailler, et très bonne circulation de l'eau et de l'air.

La structure des sols est fondamentale en raison du rôle qu'elle joue dans :

- Le stockage de l'eau ; faux
- Le mouvement de l'eau entre la surface du sol, la couche sous-jacente et le sous-sol (drainage et capillarité), en lien avec la connectivité des pores ;
- L'aération du sol ;
- Le développement et le fonctionnement des racines ;
- Le stockage et la libération de nutriments ;
- La température du sol et ses fluctuations ;
- Les lieux d'hébergement et d'activité de la vie du sol (bactéries, champignons, nématodes, vers de terre, insectes, ...) ;
- L'environnement (diminution de l'érosion, rétention et dégradation de polluants).

I.6.4 La porosité

La porosité d'un horizon, qui est en étroite relation avec les notions de réserve en eau, de circulation de l'eau et de l'air ou encore d'enracinement, peut être définie par l'ensemble des vides que comporte cet horizon.

La porosité est la fraction de l'unité de volume du sol en place qui n'est pas occupée par la matière solide (HENIN et al., 1969). En d'autres termes, c'est le volume occupé par les constituants liquides et gazeux ; ce sont aussi les voies des transferts solides, liquides et gazeux, ainsi que de l'activité biologique.

Ainsi la porosité exprime le pourcentage de vide par rapport au pourcentage de sol en place et on distingue la microporosité de la macroporosité :

La microporosité est constituée des espaces lacunaires les plus fins qui sont occupés par l'eau ;

La macroporosité est constituée des espaces les plus grands, réservés à l'air dans un sol normalement ressuyé.

Généralement, la porosité s'exprime en pourcentage (%) par la formule suivante :

$$P(\%) = \frac{V_V}{V} x100 \qquad \text{où} \quad V = V_V + V_S$$

Avec :

P : Porosité du sol (%);

V : Volume total;

V_V : Volume non occupé par la matière solide;

Vs : Volume occupé par la matière solide.

La porosité subit des variations temporelles et spatiales rapides (saisonnières et en profondeur). Cette variation est liée naturellement aux modifications de la teneur en eau et la densité apparente du sol. Elle peut varier également en fonction de l'activité racinaire donc du type de couvert végétal (DERDOUR., 1993).

-Pour MONNIER et al. (1973), ce taux serait en fonction du type de sol, ainsi pour un sol équilibré, il serait de 50% dont 20% de macroporosité et 30% de microporosité, ceci afin de permettre :

- ✓ Une bonne aération pour le développement des racines et les micro-organismes.
- ✓ Un optimum de rétention en eau.
- ✓ Une grande capacité de remontée capillaire.

La circulation d'engins sur un sol va entraîner un déplacement des particules, qui va modifier leur agencement et donc bien souvent la porosité du sol (figure 4).

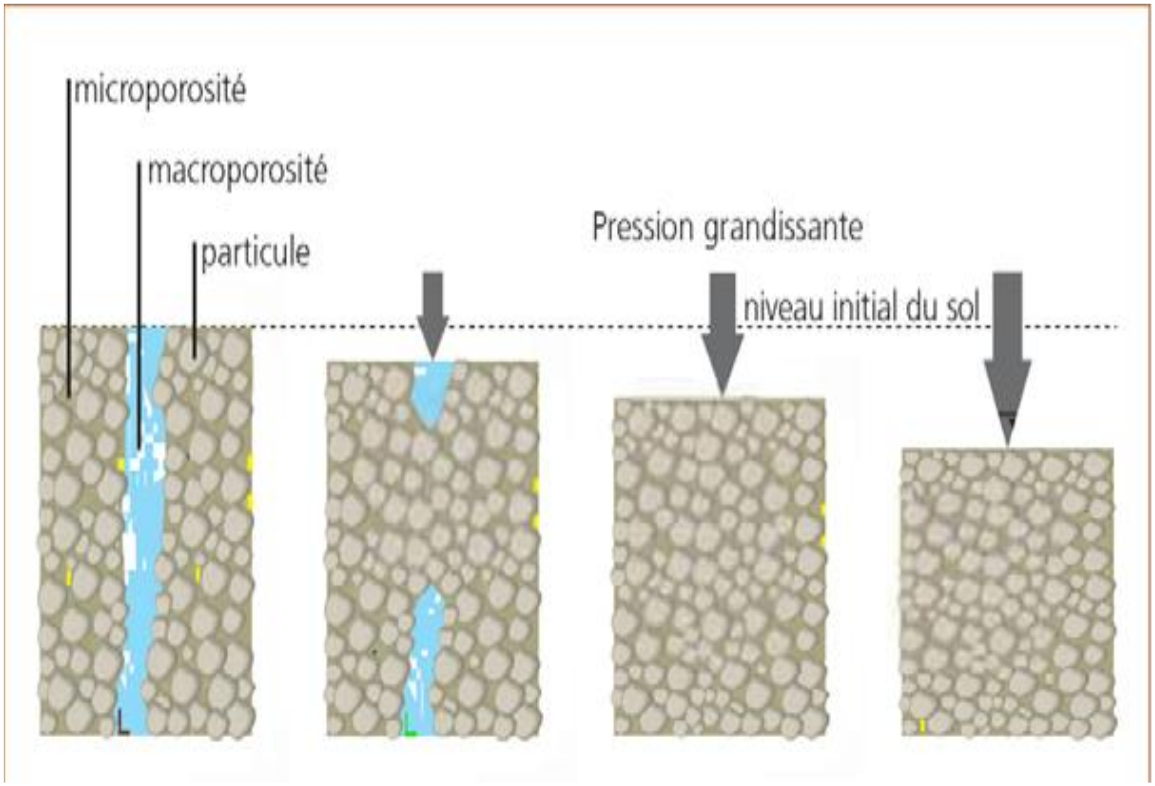


Figure 4 : Effet du tassement sur la macro et microporosité du sol.
(De Paul et Bailly, 2005)

Note : plus ou moins directement, toutes les propriétés physiques se rattachent à une caractéristique fondamentale qui est la *porosité*.

I.6.5 La densité apparente

La densité apparente est l'un des paramètres les plus importants dans les études portant sur la structure du sol. Elle est, en effet, liée à la nature et à l'organisation des constituants du sol (CHAUVEL, 1977). Elle permet, en outre, de calculer la porosité et d'apprécier ainsi indirectement la perméabilité, la résistance à la pénétration des racines (MAERTENS, 1964), la cohésion des horizons (YORO, 1983 ; YORO et ASSA, 1986) et la réserve en eau du sol (HENIN, MONNIER et GRAS, 1969). (in YORO, et GODO, 1990)

Exprimée en g/cm3, la densité apparente est le rapport du poids de sol sec sur son volume total. En d'autres termes, elle représente la masse volumique du sol sec. La densité apparente est directement fonction de la texture et de la structure, elle influence la capacité de rétention du sol en eau et en air ; sa valeur s'accroît corrélativement avec l'augmentation du compactage.

Plusieurs méthodes existent pour mesurer la densité apparente, la plus utilisée étant celle du cylindre enfoncé précautionneusement dans le sol et remonté avec son contenu, tout tassement devant être évité. Pour la majorité des sols, la densité apparente varie de 1 à 2.

I.6.6 La portance :

C'est une caractéristique qui définit la capacité d'un sol à supporter les charges qui lui sont appliquées. Elle dépend de la nature du sol, du pourcentage d'eau et du degré de compactage. La portance du sol est déterminée par l'essai Proctor.

Il existe une relation entre la portance, déformation de la surface du sol et le tassement, mais cette relation varie suivant l'importance relative du fluage. On peut, suivant La nature du sol et sa teneur en eau, avoir d'importantes déformations de surface sans tassement ou la situation inverse. L'allure de la surface du sol après roulage n'est donc pas on bon indicateur des dégâts qu'il a subis.

En sens inverse. Il existe également une relation entre tassement et portance, qui constitue une conséquence favorable du celui-ci. Un sol tassé est moins déformable et sa portance est accrue. (Guerif, 1983)

I.6.7 L'humidité d'un sol

On appelle l'humidité d'un sol, la proportion d'eau contenue dans le sol HENIN (1977).

Le régime hydrique du sol dépend directement de trois propriétés :

- La texture détermine les forces de rétention de l'eau,
- La structure influence la circulation de l'eau,
- La porosité définit le volume du réservoir hydrique de sol.

Note : L'humidité et la porosité du sol déterminent la résistance du sol à la pénétration (Ayers and Perumpral, 1982).

La connaissance de la variabilité spatiale des eaux de surface et des eaux souterraines est essentielle dans les domaines de l'agriculture, de l'environnement, de la physique des sols et de l'hydrologie des eaux souterraines surtout si le sujet étudié est l'infiltration, le ruissellement et l'évapotranspiration (Weihermüller et al., 2007). L'information sur la variabilité spatiale de la teneur en eau du sol peut fournir des connaissances importantes dans la prédiction de l'infiltration et du ruissellement de surface (Merz et Bárdossy, 1998; Pauwels et al., 2001), le rendement des cultures et le contrôle des inondations (Schlesinger et al., 1990). Le travail des tracteurs en présence d'un taux élevé d'humidité du sol peut produire des effets indésirables tels que le compactage élevé et la dégradation de la structure du sol (Hamza et Anderson, 2005).

La teneur en eau du sol peut être mesurée en utilisant trois méthodes différentes :

1) analyse au laboratoire des carottes de sol provenant du champ ;

2) mesures ponctuelles in situ;

3) mesures en continu et en temps réel.

L'analyse de laboratoire permet la détermination du poids de l'échantillon de sol avant et après évaporation de l'eau à travers la méthode de séchage au four (Topp, 1993), qui consomme du temps et nécessite beaucoup de travail. La réflectométrie temporelle (TDR), les sondes à neutrons, les blocs de gypse et l'atténuateur des rayons gamma sont autant d'exemples de techniques de détection adoptées par les méthodes ponctuelles (Topp, 1993). Les mesures ponctuelles deviennent coûteuses lorsque la teneur en eau du sol d'un grand nombre d'emplacements doit être surveillée.

La méthode la plus prometteuse pour cartographier de grandes surfaces pour la

détermination de la teneur en eau du sol est une détection sur le terrain. Les données géo référencées sont recueillies lors du déplacement sur un paysage de l'instrument de mesure (Adamchuk et al., 2004). Plusieurs capteurs d'humidité de sol ont été développés en fonction, soient des principes radiométriques ou électriques. La spectroscopie d'absorption infrarouge proche est basée sur La capacité de l'eau à absorber de l'énergie lumineuse à certaines bandes du spectre (Norris, 1964). Un spectrophotomètre VIS / NIR a été testé par Mouazen et al. (2005) pour développer un système de capteur pour la mesure exacte de la teneur en eau du sol en mesurant la réflectance de la lumière.

La résonance magnétique nucléaire est une Technique qui dépend de l'interaction entre les moments magnétiques nucléaires de l'hydrogène et un champ magnétique. Il a été utilisé pour développer un instrument qui peut être monté sur un tracteur, et qui avait une puissance d'alimentation relativement élevée (Paetzold et al., 1985). Une technique prometteuse de capteurs à base de capacitance a été explorée par certains chercheurs. Dean et al. (1987) ont développé un capteur à capacité fonctionnant à 150 MHz. Un capteur similaire a été évalué dynamiquement par Whalley et al. (1992) en utilisant des vitesses typiques de plantation de semences, il a été jugé sensible aux fluctuations de la densité apparente du sol ainsi que la CEC. Liu et al. (1996) et plus tard Andrade-Sanchez et al. (2007) ont évalué un capteur d'humidité diélectrique dans des conditions dynamiques en l'incorporant dans un bloc en nylon qui était attaché à une dent d'un cultivateur, une série d'études ont démontré que la salinité, la texture et la température affectaient également les mesures des capteurs. Un capteur de type capacité similaire a été évalué plus récemment par Adamchuk et al. (2009), Qui ont conclu que leur capteur était capable de donner des valeurs de la teneur en eau du sol avec une haute résolution et une erreur standard relativement faible (0,027 g / g ou 0,039 ml / ml). La densité apparente du sol était également prédite dans cette étude en mesurant simultanément la résistance du sol à la pénétration et l'humidité du sol.

I.7 Propriétés mécaniques du sol

Le sol présente des propriétés variables en fonction de sa constitution physique et de son humidité. Ainsi, au cours des opérations de travail du sol, le résultat du passage d'un outil dépend du comportement mécanique du sol. Il en est de même des déformations du terrain dues aux roulages de véhicules ou d'appareils de traitement phytosanitaire ou de récolte, et de celles qui se produisent sous l'effet des pressions exercées par les racines lors de leur croissance.

La granulométrie et la perméabilité en sont les caractéristiques essentielles pour définir les propriétés mécaniques des sols. On distingue les sols fins et peu perméables (limons, argiles et sables fins) des sols grossiers (sables grossiers et graviers) qui sont, eux, très perméables.

Cette propriété mécanique du sol correspond à la capacité à supporter une pression sans se déformer, ni se fracturer. Elle est fonction d'une part des différents liens entre les particules solides (cohésion) et d'autre part des forces de friction existant au point de contact entre ces particules (HILLEL, 1980). La cohésion ainsi que les forces de frictions inter-particulaires dépendent des propriétés physiques, chimiques et biologiques du sol. Ainsi, la résistance à l'enfoncement varie selon la texture du sol, sa structure, sa teneur en eau et sa masse volumique apparente. D'autres facteurs comme la présence de fragments grossiers ou encore la teneur en matière organique.

L'augmentation de la résistance mécanique des sols liée au phénomène de compactage peut nuire à la croissance racinaire. En effet, pour qu'une racine puisse croître, il faut que la pression qu'elle est capable d'exercer soit supérieure à la résistance mécanique du sol.

Le premier auteur qui ait mesuré la pression maximale que peuvent exercer les racines est Pfeiffer en 1893. Il a trouvé des valeurs allant de 0,7 à 2,5 MPa, ceci pour différentes espèces de plantes (BENNIE, 1991). D'autres auteurs ont trouvé des valeurs allant de 0,24 à 1,45 MPa pour l'élongation et de 0,51 à 0,90 MPa pour la croissance radiale (BENNIE, 1991). La valeur de 3 MPa est habituellement reconnue comme le seuil limite au-delà duquel la croissance racinaire est arrêtée.

L'indice de cône du sol (CI) est une propriété mécanique du sol largement utilisée pour évaluer la résistance du sol à la recherche de travail du sol (tableau 2).

Tableau 2: Valeurs typiques d'index de cônes (Brixius, 1987)

Classe de sol	Indice de cône		Conditions typiques
	kN/m^2	psi	
	0	0	
Sol léger ou sableux	350	51	Récolte de riz.
(CI = 450)	480	70	Discage sur un terrain labouré ou Exploitation des terres basses.
	700	101	Labour de printemps ou Terrassement sur sol humide.
Sol moyen ou labouré	850	123	Plantation, culture de champ
(CI = 900)	1000	145	Récolte de maïs, labour d'automne
	1200	174	Récolte de blé
Sol Agricole	1750	254	Labour d'été
(CI = 1800)			Enregistrement en saison sèche
			Terrassement sur un sol sec et argileux

II. Mesure des propriétés du sol

Comme propriété mécanique du sol utilisée pour évaluer la résistance du sol est la résistance du sol à la pénétration, mesurée par un pénétromètre à cône est également appelé indice de cône du sol (CI) (tableau2).

L'indice de cône du sol (CI) est utilisé comme indicateur important pour le compactage du sol (Bédard et al., 1997; Tessier et al., 1997), le développement des racines (Chen et al., 2005), l'infiltration d'eau dans le sol (Botta et al., 2006), dans l'appréciation du travail des outils de travail du sol (Manuwa et Ademosun, 2007) et l'étude des performances des tracteurs (Mari Et al., 2006).

L'indice de cône du sol (CI) varie en profondeur. Les valeurs inférieures du CI sont associées à une couche cultivée près de la Surface du sol, tandis que les valeurs du CI les plus élevées sont associées à une couche de sol compact au-dessous de la couche cultivée (Doan et al., 2005). L'humidité du sol est un facteur important qui affecte le CI

des sols (Franzen et al., 1994). Typiquement le sol est plus sec à des valeurs de CI plus élevées (Francis et al., 1987; Tekeste et al., 2008). Busscher et al. (1997) ont trouvé une relation linéaire inverse entre CI et l'humidité du sol, tandis que Ohu et al. (1988) ont trouvé une relation exponentielle entre le CI et la teneur en humidité des sols limoneux et argileux. Le CI est également lié à la densité apparente du sol et les paramètres texturaux du sol. Ayers et Perumpral (1982) Ont signalé une relation directe entre l'IC et la densité apparente.

On peut dire donc que l'indice de cône du sol est lié aux propriétés physiques du sol tel que les paramètres texturaux du sol (le sable, le limon et la teneur en argile), teneur en humidité, densité apparente et système de culture, ainsi que les pratiques de travail du sol.

Dans la littérature scientifique, deux différentes méthodes sont utilisées pour mesurer la résistance à la pénétration du sol. Dans la première méthode, les échantillons de sol sont prélevés sur le terrain à une certaine profondeur de sol à l'aide d'un tube, ensuite les échantillons sont analysés en laboratoire pour déterminer la résistance à la pénétration. Avec l'autre méthode, on utilise un pénétromètre à cône. Afin de déterminer l'indice de cône du sol, on utilise généralement des pénétromètres statiques ou dynamiques. Le cône agit perpendiculairement sur la surface du sol pour les deux pénétromètres. La différence entre les deux pénétromètres réside dans la façon dont la force de pénétration du cône dans le sol est fournie.

Le pénétromètre dynamique de construction très simple, il est composé d'une tige verticale terminée par la pointe d'un cône. Une masselotte coulisse autour de la tige. On la remonte manuellement et on la laisse tomber : le choc sur une butée provoque l'enfoncement de la pointe dans le sol.

En reportant sur un graphique les enfoncements successifs et les nombres de coups correspondants, on obtient un profil pénétrométrique en énergie de pénétration.

En utilisant la formule empirique dite" **des Hollandais** », on obtient les valeurs correspondantes de la résistance à la pénétration. Ce type de pénétromètre est souvent utilisé lorsque le sol est très dur. (BILLOT. 1982).

$$RP = \frac{M^2 \times h}{2(M + m)S} \times \frac{n}{\Delta Z}$$

RP : résistance à la pénétration en kg/cm^2

M : masse de la masselotte en kg

m : masse du reste du pénétromètre en kg

h : hauteur de chute de la masselotte en cm

S : section maximum de la pointe en cm^2

n : nombre de coups (sans unité)

ΔZ : enfoncement correspondant en cm

Par contre, avec le pénétromètre statique, le cône est enfoncé dans le sol par un procédé manuel, mécanique, hydraulique ou électrique. Une pointe conique spécifiquement dimensionnée est enfoncée en continu verticalement dans le sol à vitesse constante et standard, en commençant à partir de la surface du sol (Drummond, et al., 2000). L'angle de pointe du cône est de 30°. La résistance à la pénétration du sol est calculée en divisant la force nécessaire pour insérer le cône dans le sol à une vitesse standard de 30 mm/s par la surface de base du cône. Cette valeur calculée du cône s'appelle l'indice de cône (CI) et est exprimée par l'équation suivante (ASAE, 2002) :

$$CI = \frac{F}{A}$$

Où
CI : Cône index [MPa],
F : Force [N],
A : surface de base du cône [mm^2].

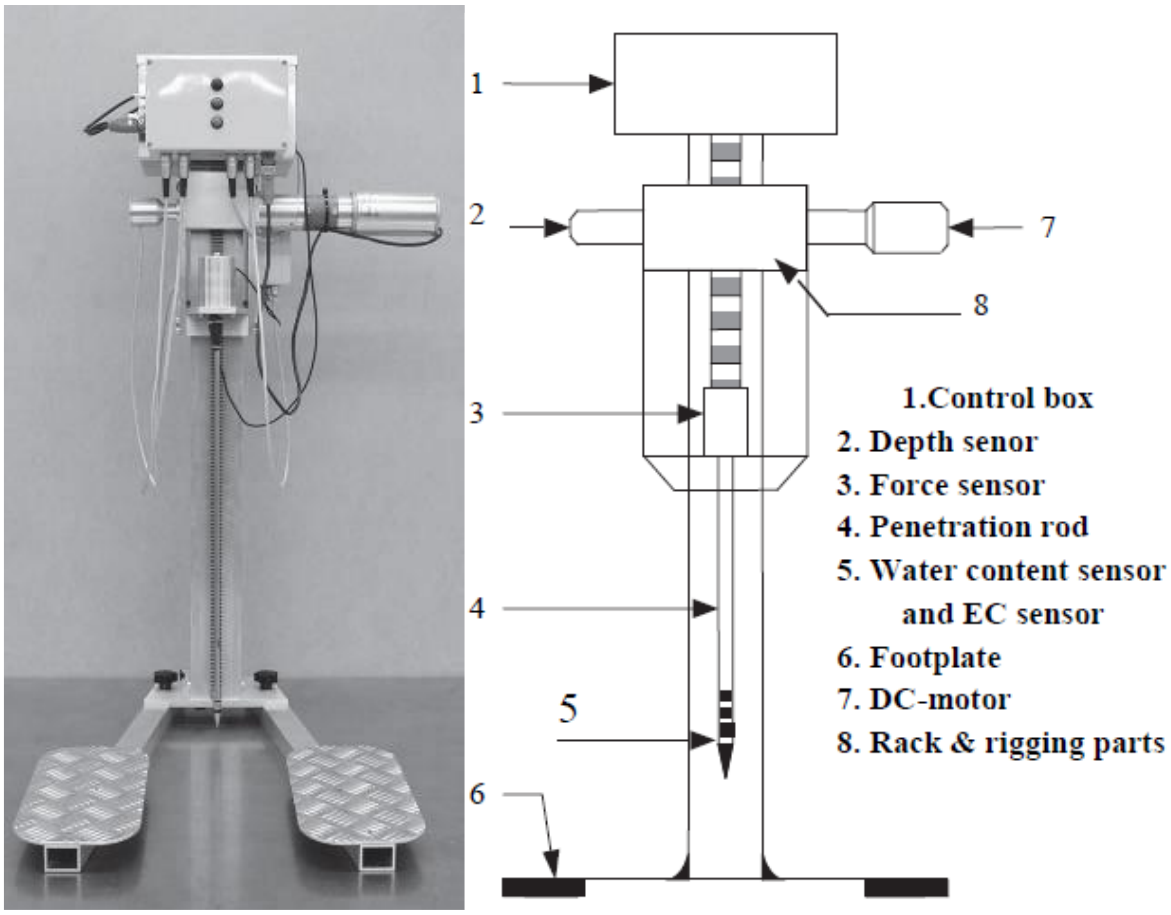


Figure 5 : Pénétromètre à cône vertical avec capteurs pour la teneur en eau du sol et la conductivité électrique apparente (Sun et al., 2008).

En raison de l'influence évidente de la vitesse de pénétration sur la mesure de l'indice de cône (CI). Sun, et al., (2007) ont conçu une méthode attrayante pour déterminer la force de pénétration d'un pénétromètre par exploitation de la variation du courant électrique d'un moteur à courant continu en utilisant un capteur Hall et en maintenant la vitesse de pénétration du cône constante. D'autres comme Zenga, et al., (2008) ont élaboré un pénétromètre horizontal à double capteur pour la mesure simultanée de la teneur en eau et de la force de résistance du sol. D'autres chercheurs ont développé d'autres, méthodes mais elles restent toujours ponctuelles et ne permettent pas une mesure en continu. Sun et al., (2008) ont développé une technique a multi capteurs pour mesurer les propriétés physiques du sol, y compris l'eau du sol, la résistance mécanique et la conductivité électrique (figure 5).

Pour des mesures ponctuelles, selon Campbell et O'Sullivan, (1991) même automatisés, les mesures d'indice de cône en pénétrométrie prennent beaucoup de

temps et sont très variables d'un point à l'autre. Cette méthode de mesure ne permet donc pas d'être embarquée sur des tracteurs, afin de réaliser des mesures en continues et en temps réel. Elle ne permet pas de réaliser des systèmes de contrôle des travaux du sol selon l'état actuel du sol afin de prendre des décisions in situ et en temps réel. D'autres travaux ont été faites dans ce domaine, afin de trouver des méthodes ou des systèmes de mesures, avec des capteurs adéquats de la résistance de sol en continu et en temps réel. On trouve parmi ces travaux, Abbaspour-Gilandeh, (2009) ont conçu et construit un système de mesures avec plusieurs tiges instrumentées pour mesurer la résistance mécanique du sol à des différentes profondeurs sur l'ensemble des 40 cm du profil du sol tout en se déplaçant à travers le champ. (Hemmata et al, 2014) ont mis au point un pénétromètre horizontal acoustique à multiple point de mesures, avec trois corps prismatiques d'un angle de 30° fixés horizontalement aux dents d'un cultivateur en forme de S et travaille à des profondeurs de 10, 20 et 30 cm. Les bouts travaillant à 10 et 30 cm de profondeur ont été équipés de microphones.

Des groupes de recherche à travers le monde se concentrent sur le développement de nouvelles techniques d'acquisition des données (Adamchuk et al., 2004). Cette recherche était basée sur la cartographie sur place des propriétés du sol en intégrant plusieurs capteurs dans une seule unité mobile. Dans cette étude, un système de cartographie intégrée des sols (ISMS) a été développé pour combiner trois différents capteurs du sol ensemble pour mesurer la teneur en humidité du sol, la matière organique du sol et la résistance mécanique du sol dans un seul passage au champ. Un capteur à capacitance a été utilisé pour mesurer la constante diélectrique du sol afin de calculer la teneur en eau du sol, ces données ont été utilisées pour prédire la teneur volumétrique en eau gravimétrique du sol (pondérale). Pour prédire la teneur en matière organique du sol, deux capteurs optiques ont été utilisés pour mesurer la réflectance du sol à deux longueurs d'onde différentes, une cellule de charge de type capacité a été développée pour mesurer la force de cisaillement du sol en un seul passage pour le calcul de la résistance mécanique du sol.

III. L'agriculture de précision

L'agriculture de précision (PA), est un concept moderne de gestion des entreprises agricoles, utilisant des techniques numériques pour contrôler et optimiser les processus de production agricole.

L'agriculture de précision part d'un constat : les sols agricoles sont hétérogènes. Texture, structure, pH, teneurs minérales, matière organique, topographie, teneur en eau du sol... : tous ces paramètres sont susceptibles de varier à l'intérieur d'une même parcelle (Zwaenepoel et Le Bars, 1997).

Par exemple, plutôt que d'appliquer la même quantité d'engrais sur une parcelle agricole entière, l'agriculture de précision mesure les variations dans les paramètres du sol et de la culture et adapte une stratégie de fertilisation ou de récolte en conséquence.

Les méthodes de l'agriculture de précision promettent d'augmenter la quantité et la qualité de la production agricole tout en utilisant moins d'intrants (Eau, énergie, fertilisants, pesticides ...).

L'objectif est d'économiser les coûts, de réduire l'impact sur l'environnement et de produire plus avec de meilleure qualité. Les méthodes de l'agriculture de précision (PA) reposent principalement sur une combinaison de nouvelles technologies de capteurs, la technique de navigation par satellites et de positionnement (GPS), et l'Internet des choses (IoT). L'agriculture de précision, l'agriculture numérique et l'Internet des choses sont des concepts incontournables dans l'agriculture moderne et future.

L'agriculture de précision obéie à la démarche suivante (Grenier. 2014):

- Acquisition de données sur l'hétérogénéité de la parcelle. Domaine de la Proxidétection/Télédétection (utilisation des capteurs et des techniques numériques) ;
- Visualisation, analyse de données et décisions (utilisation de SIG, modélisation, OAD, Internet, ...) ;
- Modulation des actions (utilisation de l'électronique embarquée avec de la géolocalisation des actions) ;
- Traçabilité et retour d'expérience (Localisation spatiale et temporelle avec stockage dans une base de données).

IV. Les capteurs

Un capteur est un transducteur capable de transformer un stimulus d'une grandeur physique (force, chaleur, pression, lumière ...) en une autre grandeur physique, généralement électrique (tension) exploitable par un système électronique et/ou informatique. Les capteurs ont une histoire de développement similaire aux machines. Les capteurs sont omniprésents dans la vie quotidienne, on les trouve partout, dans les machines dans les téléphones et dans les appareils. Plusieurs avancées récentes ont rendu les capteurs plus applicables à l'agriculture (tableau 3). Tout d'abord, ils ont été couplés aux convertisseurs analogique/numérique suivis d'une miniaturisation puis à la communication radio. Aujourd'hui, un capteur placé dans un champ peut mesurer un phénomène physique, convertir cette mesure en un signal électronique puis transmettre ce signal avec des ondes électromagnétiques à une station de base éloignée. Cette mesure détectée et sa transmission par ondes radio peuvent être effectuées automatiquement, libérant la nécessité pour quelqu'un d'être sur place pour récupérer les données enregistrées.

IV.1 Principales caractéristiques des capteurs

Etendue de mesure : valeurs extrêmes pouvant être mesurée par le capteur.

Résolution : plus petite variation de grandeur mesurable par le capteur.

Sensibilité : variation du signal de sortie par rapport à la variation du signal d'entrée.

Précision : aptitude du capteur à donner une mesure proche de la valeur vraie.

Rapidité : temps de réaction du capteur. La rapidité est liée à la bande passante.

Linéarité : représente l'écart de sensibilité sur l'étendue de mesure.

La performance d'un capteur peut être influencé par une grandeur physique autre que le mesurande dont la variation peut modifier la réponse du capteur :

- Température : modifications des caractéristiques électriques, mécaniques et dimensionnelles
- Pression, vibrations : déformations et contraintes pouvant altérer la réponse
- Humidité : modification des propriétés électriques (constante diélectrique ou résistivité). Dégradation de l'isolation électrique

- Champs magnétiques : création de fem d'induction pour les champs variables ou modifications électriques (résistivité) pour les champs statiques
- Tension d'alimentation : lorsque la grandeur de sortie du capteur dépend de celle-ci directement (amplitude ou fréquence)

Pour réduire l'influence de ces grandeurs physiques sur la performance d'un capteur il est nécessaire de :

- Réduire les grandeurs d'influence (table anti-vibration, blindage magnétique...)
- Stabiliser les grandeurs d'influence à des valeurs parfaitement connues
- Compenser l'influence des grandeurs parasites par des montages adaptés (pont de Wheatstone)

IV.2 Les différents types de capteurs

Il existe trois types de capteurs qui sont :

- Capteur actif,
- Capteur passif,
- Capteur composite.

IV.2.1 Capteur actif

C'est un capteur qui fonctionne comme un générateur, il transforme en énergie électrique la forme d'énergie propre à la grandeur physique à mesurer, énergie thermique, mécanique ou rayonnement. Les plus classiques sont, effet thermoélectrique, effet piézo-électrique, effet d'induction électromagnétique, effet photoélectrique, effet Hall, effet photovoltaïque.

IV.2.2 Capteur passif

Un capteur passif est considéré comme une impédance dont l'un des paramètres est sensible à la grandeur mesurée. Cette impédance doit ensuite être intégrée dans un circuit pour pouvoir retrouver une grandeur électrique en sortie. Le montage qui permet ceci est appelé conditionneur. Il existe plusieurs sortes de conditionneur comme le montage potentiométrique, le pont de Wheatstone, les circuits oscillants ou les amplificateurs opérationnels.

IV.2.3 Capteur composite

Un capteur composite est un capteur constitué d'un corps d'épreuve et d'un capteur actif ou passif. Le corps d'épreuve, quant à lui, est un capteur qui, soumis à la grandeur a mesurée, donne une grandeur physique non électrique appelée mesurande secondaire qui, elle va être traduite en une grandeur électrique par un capteur.

Tableau 3: Différents types de capteurs pour la prédiction des principales propriétés du sol (adapter par Viscarra Rossel et al., 2011 in **Ciza Thomas**)

	Sensor type							
Soil property[1]	Gamma-ray	X-ray	Optical	Microwave	Radio wave	Electrical	Electrochemical	Mechanistic
Chemical								
Total carbon	D	D	D					
Organic carbon	I		D					
Inorganic carbon	I		D					
Total nitrogen	D	D	D					
Nitrate-nitrogen			I		I	I	D	
Total Phosphorus	D	D	I					
Extractable phosphorus								
Total Potassium	D	D	D					
Extractable potassium			I				I	
Other major nutrients	D	D	D					
Micronutrients, elements	D	D	D					
Total Iron	D	D	D		I			
Iron oxides	I		D		I			
Heavy metals	D	D	I					
CEC	I		I			I		
Soil pH	I		I		D		D	
Buffering capacity and LR			I				I	
Salinity and sodicity					D	D	D	
Physical								
Color			D					
Water content	D		D	D	D	D		I
Soil matric potential	I					D		I
Particle size distribution	I		I		I	I		I
Clay minerals	I	D	D			I		I
Soil strength								D
Bulk density	I		I		D			I
Porosity								D
Rooting depth					I			D

1 – Soil properties directly (D) or indirectly (I) predictable using different types of proximal soil sensors

V. Les microcontrôleurs

Avec les progrès de la miniaturisation, et la standardisation des besoins des microprocesseurs et les contrôleurs de périphériques, toute une carte mère a pu être mise dans une seule puce appelée microcontrôleurs. L'usage de microcontrôleurs est actuellement en plein développement et est dans tous les domaines de la vie industrielle et courante, et à tous les degrés de complexité (de 8 pattes à près de 200 pattes et plus).

V.1 Structure interne d'un microcontrôleur

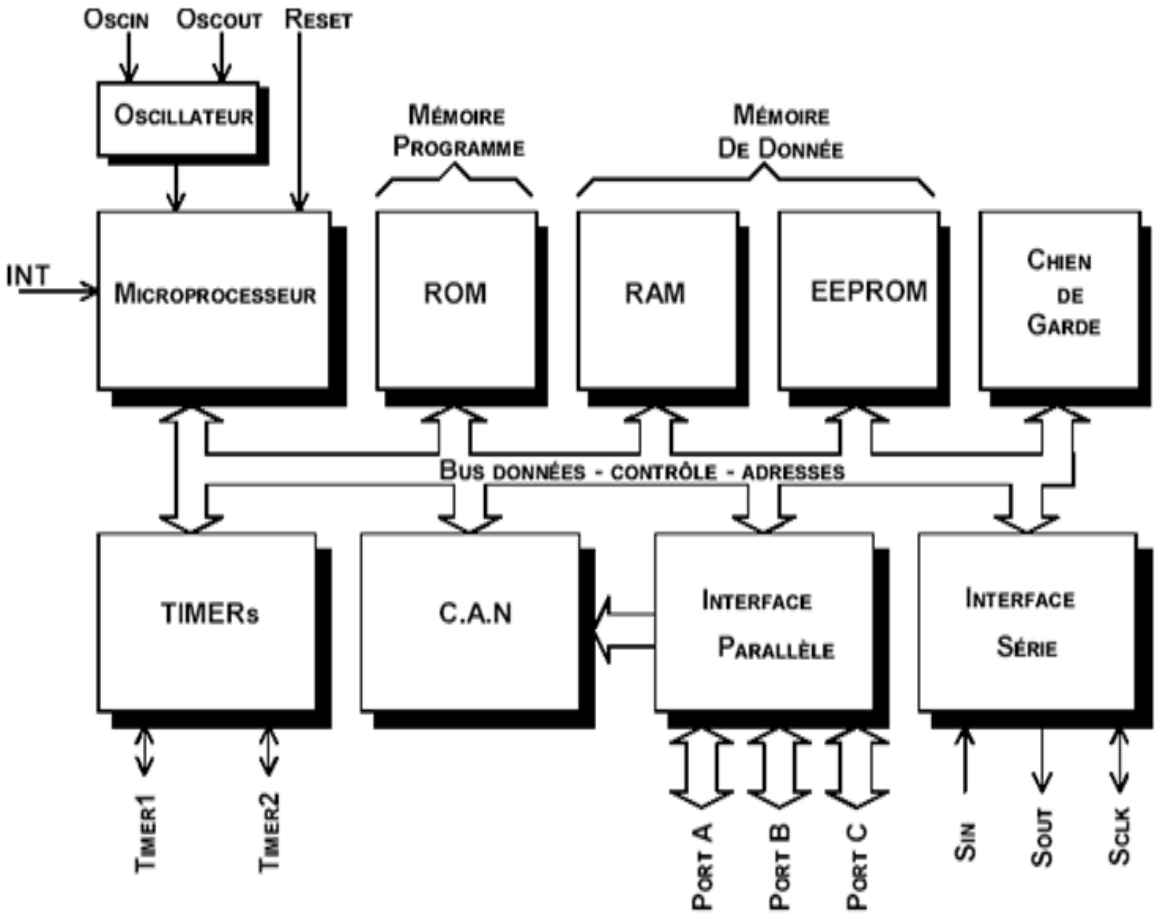


Figure 6 : Structure interne d'un microcontrôleur

Un microcontrôleur se présente sous la forme d'un circuit intégré réunissant tous les éléments d'une structure à base de microprocesseur. Généralement (figure 6) dans un microcontrôleur, on trouve les éléments suivants :

- Un processeur (CPU), avec un bus de données allant de 4 bits pour les modèles les plus basiques à 32 ou 64 bits pour les modèles les plus évolués ;
- Une mémoire vive (RAM) pour le stockage des données et des variables ;

– Une mémoire morte (ROM) généralement mémoire flash pour stocker le firmware. Mais différentes technologies peuvent être employées : EPROM, EEPROM ;
– Un oscillateur pour le cadencement. Il peut être un quartz, un circuit RC ou une PLL ;
– Des périphériques, pour effectuer des tâches spécifiques. Dont on peut citer :
    - Les convertisseurs analogiques-numériques (ADC),
    - Les convertisseurs numériques-analogiques (DAC),
    - Les générateurs de signaux à modulation de largeur d'impulsion (PWM pour Pulse Width Modulation),
    - Les timers/compteurs (compteurs d'impulsions d'horloge interne ou d'événements externes),
    - Les chiens de garde (watchdog),
    - Les comparateurs,
    - Les contrôleurs de bus de communication (UART, I2C, SSP, CAN, USB, one wire, Ethernet, etc.).

Par programmation, les microcontrôleurs peuvent se placer dans un état de sommeil, dans lequel ils présentent une très faible consommation électrique. Un signal envoyé par l'un de leurs périphériques (timer, broche d'entrée-sortie, watchdog, etc.) permet de les faire sortir de cet état de sommeil. Le périphérique peut être paramétré et commandé par le programme et/ou les entrées-sorties. Les périphériques peuvent générer une interruption qui oblige le processeur à quitter le programme en cours pour effectuer une routine de traitement de l'interruption, lorsque l'événement qui la déclenche survient. La plupart des microcontrôleurs ont un nombre restreint de broches, si bien qu'une broche donnée peut correspondre à plusieurs périphériques internes. La fonction choisie doit alors être sélectionnée par programme.

V.2 Familles de microcontrôleurs

Il existe plusieurs familles de microcontrôleurs dont les plus connues sont :

- la famille Atmel AT91 ;
- la famille Atmel AVR (utilisée par des cartes Arduino) ;
- la famille Intel 8051, et l'Intel 8085,

- le Freescale 68HC11 ;
- la famille Freescale 68HC08 ;
- la famille Freescale 68HC12 ;
- la famille des PIC de Microchip ;
- la famille des dsPIC de Microchip ;
- la famille des ST6, ST7, STM8, ST10, STR7, STR9, STM32 de STMicroelectronics ;
- la famille PICBASIC de Comfile Technology;

V.3 Environnements de développement

Un environnement de développement Intégré, EDI (ou IDE en anglais), est un logiciel regroupant un ensemble d'outils nécessaires au développement des applications dans un langage L de programmation et qui sont comme suit :

V.3.1 Le langage de programmation

Il faut noter que le programme d'un microcontrôleur est généralement appelé firmware. Le langage de programmation C/C++ est la référence pour programmer les microcontrôleurs, car il présente l'avantage d'être un langage de programmation orienté matériel. Le code machine généré par le compilateur est presque optimal en ce qui concerne d'une part la taille de la mémoire requise et d'autre part la vitesse d'exécution du programme.

V.3.2 L'éditeur de code

Un éditeur de code est un éditeur texte qui permet comme son nom l'indique d'éditer des fichiers texte, et qui propose des fonctions suivantes :

- La coloration syntaxique (syntax highlighting) qui sert à rendre le code source plus lisible en appliquant des couleurs en fonction de la syntaxe du langage (mots clés)
- Autocomplétion des mots clés du langage
- Rechercher & remplacer : ces fonctions permettent de rechercher du texte, et de le remplacer par un autre texte, très pratique lorsqu'on programme
- Gestion des différents encodages : UTF-8, Latin1
- Connexion à un serveur FTP
- Snippet de code

V.3.3 Le compilateur

Une fois le programme écrit, il faut le compiler, c'est à dire transformer le code source en code binaire instructions directement compréhensibles par le processeur du microcontrôleur. Le fabricant du microcontrôleur utilisé proposera un compilateur spécifique à son matériel, allant généralement de pair avec l'outil de développement. C'est le cas, par exemple, pour le compilateur XC qui accompagne l'outil MPLAB X, le compilateur Keil pour MDK-ARM ou le compilateur IAR pour l'outil du même nom.

V.3.4 Le programmateur

Une fois le programme compilé, il faut pouvoir l'inscrire dans la mémoire du microcontrôleur ciblé. Pour cela, il faut un outil matériel communément appelé un programmateur.

V.3.5 Le débogueur

Le débogueur permet d'exécuter le programme et de superviser son fonctionnement en temps réel depuis un ordinateur.

V.3.6 Le simulateur

Le simulateur et un logiciel fonctionnant sur un pc et qui permet de tester le code d'un programme pour microcontrôleur sans que la partie matérielle ne soit disponible.

Il faut retenir qu'un programme qui fonctionne sans problème sur un simulateur n'implique pas forcément qu'il fonctionnera aussi bien sur un microcontrôleur.

Note : ces outils vont nous servir pour le développement du système de mesures.

VI. L'ARDUINO

VI.1 Qu'est que L'ARDUINO

Arduino est une plate-forme de développements interactifs, constituée d'une carte électronique et d'un environnement de programmation, elle sert à la création de circuits électroniques plus ou moins complexes presque dans tous les domaines. Cet environnement matériel et logiciel permettent à l'utilisateur de formuler ses projets par l'expérimentation directe avec l'aide de nombreuses ressources disponibles en ligne. Pont tendu entre le monde réel et le monde numérique, Arduino permet d'étendre les capacités de relations humain/machine ou environnement/machine. Arduino est un projet en source ouverte, open source.

La carte Arduino repose sur un microcontrôleur associé à des entrées et des sorties qui permettent à l'utilisateur de brancher différents types d'éléments externes.

Comme le logiciel Arduino, le circuit électronique de cette carte est libre et ses plans sont disponibles sur internet. On peut donc les étudier et créer des dérivés. Plusieurs constructeurs proposent ainsi différents modèles de circuits électroniques programmables et utilisables avec le logiciel Arduino. Ainsi, il existe plusieurs variétés de cartes Arduino.

La première caractéristique des cartes Arduino est le type de microcontrôleur dont elles sont équipées. Actuellement, et probablement pour encore longtemps, deux familles de microcontrôleurs équipent les Arduinos : des AVR 8 bits de la société ATMEL et des ARM 32 bits de la série Cortex-M, conçus par la société Advanced RISC Machines et fabriqués par différents fabricants, ATMEL pour les cartes officielles, mais aussi Freescale ou STMicro pour des cartes compatibles gravitant autour de la galaxie Arduino. Les cartes Arduino à base d'AVR sont plus simples et comportent moins de mémoire que les cartes à base d'ARM. Leur capacité de calcul est également plus faible, mais elles sont généralement meilleur marché.

La carte la plus répondue est l'Arduino Uno. Il s'agit d'une carte équipée d'un microcontrôleur de la famille AVR 8 bits : l'ATMega328 (figure 7 et 8).

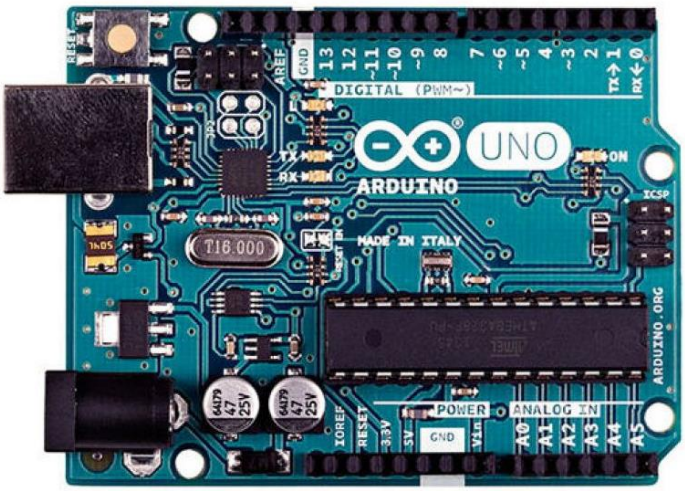


Figure 7 : Arduino Uno R3 Officiel

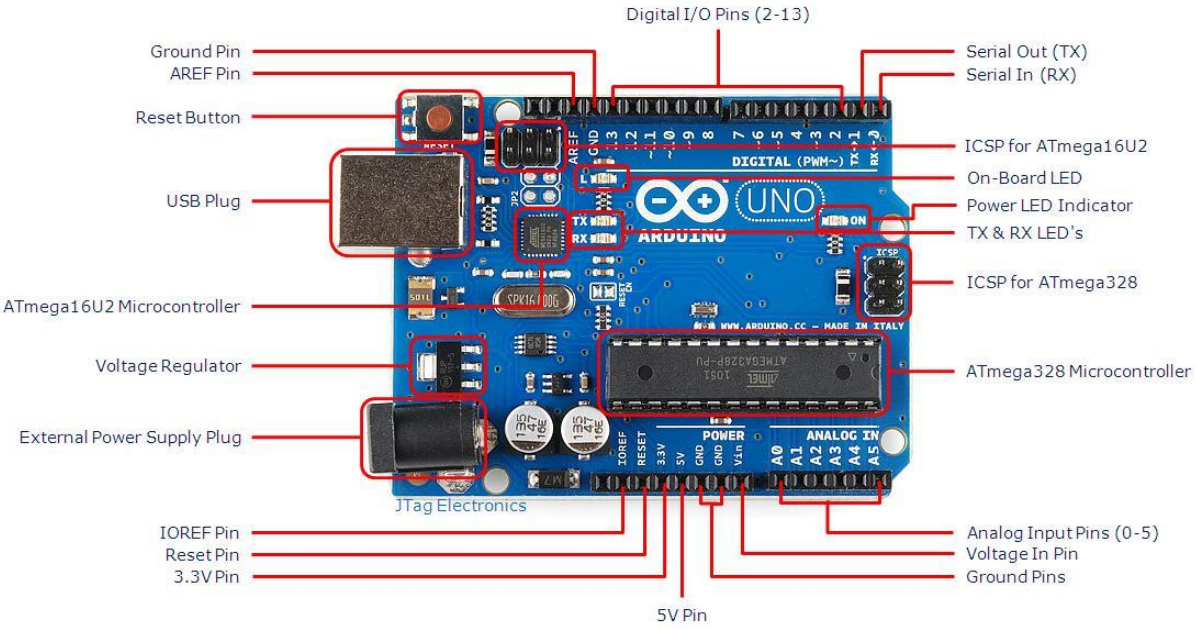


Figure 8 : Description carte Arduino Uno

Beaucoup de cartes sont équipées de l'ATMega328 (Uno, Pro Mini, Nano, ...) mais, parmi les AVR, on trouve aussi l'ATMega32u4. Ce dernier se distingue par une interface USB intégrée qui permet de se passer du convertisseur USB/série pour la communication avec l'ordinateur et par une quantité de mémoire vive de 25 % plus élevée. L'ATMega32u4 équipe entre autres le Leonardo, le Micro et le Pro Micro. En théorie, les cartes à base d'ATMega32u4 devraient être moins chères mais, ce n'est pas le cas.

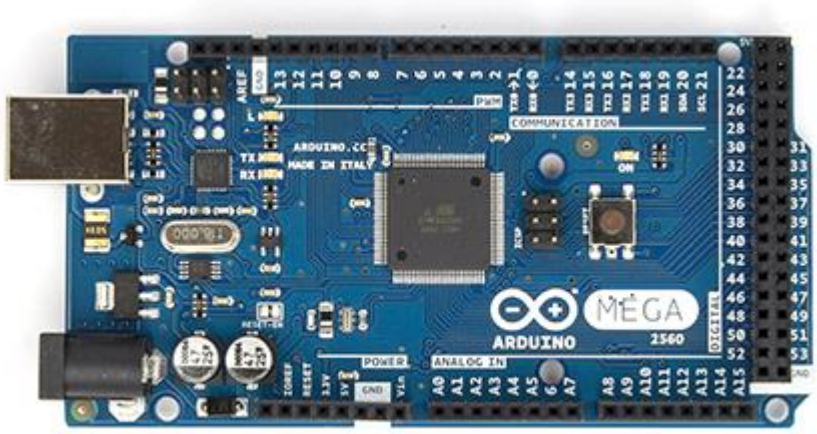


Figure 9 : Carte Arduino MEGA

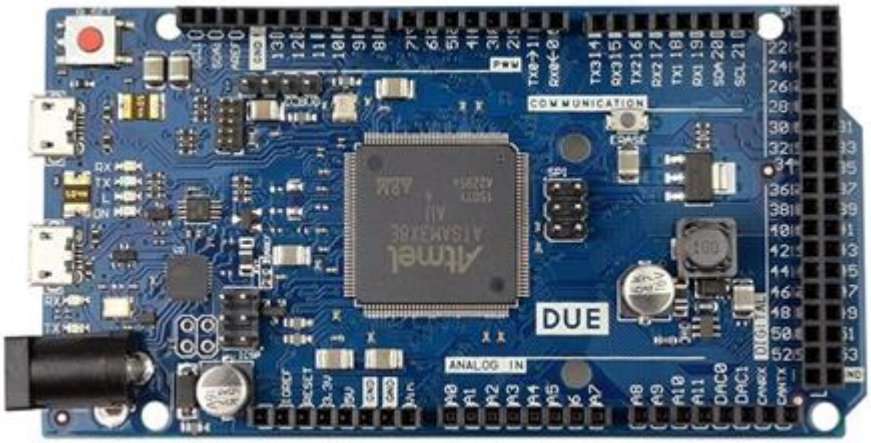


Figure 10 : Carte Arduino DUE

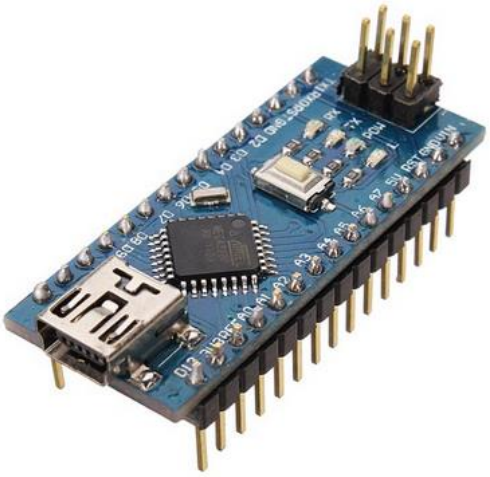


Figure 11 : Carte Arduino nano

VI.2 Capacités d'entrée/sortie d'une carte Arduino

Une carte Arduino communique avec son environnement par l'intermédiaire de ses broches d'entrées/sorties. Sur ces broches, vont être connectés des capteurs (dispositifs permettant de transformer une information de l'environnement en signal électrique) et des actionneurs (dispositifs permettant de transformer un signal électrique en action mécanique ou lumineuse). Le nombre de broches disponible sur la carte est un critère de choix important, car il détermine le nombre de capteurs et d'actionneurs que l'on va pouvoir connecter.

L'Arduino étant un ordinateur spécialisé dans la gestion de capteurs et d'actionneurs, c'est le programme qui se trouve dans le microcontrôleur qui va décider de la manière dont les capteurs et les actionneurs sont utilisés. C'est donc très différent de l'électronique traditionnelle où les fonctions qui relient les capteurs aux actionneurs, sont déterminées « en dur et simple » par le câblage entre les composants, et par les composants eux-mêmes. Quand les fonctions sont déterminées par un programme, elles peuvent être, d'une part, beaucoup plus élaborées et, d'autre part, modifiables à volonté sans changer le câblage.

VI.3 Arduino Uno

La version la plus courante d'Arduino est l'Arduino Uno. Cet avis est que la plupart des gens la pointe quand ils se réfèrent à un Arduino. L'Uno est l'une des cartes les plus populaires dans la famille Arduino et un excellent choix pour les débutants. Il existe différentes révisions de l'Arduino Uno, voir figure est la révision la plus récente (Rev3 ou R3).

L'Arduino Uno est basé sur un microcontrôleur l'ATmega328. Il possède 14 broches d'entrée/sortie numériques (dont 6 peuvent être utilisées comme sorties PWM), 6 entrées analogiques, un quartz de 16 MHz, une connexion USB, une prise de courant, une fiche ICSP et un bouton de réinitialisation. Elle contient tout le nécessaire pour exploiter le microcontrôleur ; il suffit de la connecter à un ordinateur avec un câble USB ou de l'alimenter avec un adaptateur AC-DC ou une batterie pour commencer.

VI.3.1 Fiche USB et fiche d'alimentation externe

Chaque carte Arduino doit être connectée à une source d'alimentation. L'Arduino Uno peut être alimenté à partir d'un câble USB provenant de l'ordinateur ou d'une

alimentation externe dont la plage recommandée est de 7 à 12 volts. La source d'alimentation est sélectionnée automatiquement. La connexion USB permet également de charger le code sur la carte Arduino.

VI.3.2 Régulateur de tension

Un régulateur de tension est un élément qui permet de stabiliser une tension à une valeur fixe, et qui est nécessaire pour les montages électroniques qui ont besoin d'une tension qui ne fluctue pas, ne serait-ce que peu. Un régulateur de tension peut être composé d'un ensemble de composants classiques (résistances, diodes zener et transistor par exemple), mais il peut aussi être de type "intégré" et contenir tout ce qu'il faut dans un seul et même boitier, pour faciliter son usage. C'est ce genre de régulateur intégré dont il est question dans cette carte. Il a pour rôle de délivrer 5V et 3,3V stable à partir d'une alimentation externe dont la plage recommandée est de 7 à 12 volts.

VI.3.3 Broche de puissance

Tension d'entrée V_{IN} - La tension d'entrée de la carte Arduino lorsqu'elle utilise une source d'alimentation externe (par opposition à 5 volts à partir de la connexion USB ou d'une autre source d'alimentation régulée).

Broche 5V - Cette broche délivre un 5V régulé du régulateur de la carte. La carte peut être alimentée soit à partir de la prise d'alimentation CC (7 - 12V), soit par le connecteur USB (5V), soit par la broche V_{IN} de la carte (7-12V). Alimenter la tension via les broches de 5 V ou 3,3 V contourne le régulateur et peut endommager la carte. Ce n'est pas recommandé.

Broche 3.3V - Une alimentation de 3,3 volts générée par le régulateur de bord. Le courant maximal délivrer est de 50 mA.

Broches de masse - Il y'a plusieurs broches GND sur l'Arduino, dont n'importe laquelle peut être utilisé pour la mise à la terre du circuit.

Broche IOREF - Cette broche fournit la référence de tension avec laquelle le microcontrôleur fonctionne. Elle est destinée à indiquer aux shields la tension de fonctionnement de l'Arduino. Sur un Arduino 5V, elle aura une tension de 5V et sur un 3,3V une tension de 3,3V.

VI.3.4 Broches d'entrée et de sortie

Chacune des 14 broches numériques de la carte Arduino Uno peut être utilisée comme entrée ou sortie. Elles fonctionnent en 5 volts, logique TTL. Ces broches peuvent être utilisées à la fois pour l'entrée numérique (comme pour indiquer si un bouton est poussé) et pour la sortie numérique (comme l'alimentation d'une LED). Chaque broche peut fournir ou recevoir un maximum de 40 mA et dispose d'une résistance pull-up interne (déconnectée par défaut) de 20-5k Ohms. En outre, certaines broches ont des fonctions spécialisées :

Serial Out (TX) & Serial In (RX) - (RX) utilisé pour recevoir et (TX) transmettre des données en série avec la technologie TTL. Ces broches sont connectées aux broches correspondantes de la puce ATmega 16U2 (USB-to-TTL). Elle est également employée pour dialoguer avec certains modules comme les modules radio XBee ou certains modules Wi-Fi ou encore Bluetooth.

Interruptions externes - les broches 2 et 3 peuvent être configurées pour déclencher une interruption sur une valeur faible, un front montant ou descendant ou un changement de valeur.

PWM - selon les Arduinos, de 6, pour l'Arduino Uno précède par le tilde ~ (3, 5, 6, 9, 10 et 11), à 15 broches, pour l'Arduino Mega, certaines broches peuvent être configurées en sortie PWM (Pulse-Width Modulation). Le rapport cyclique de la PWM est réglé par programme entre 0, le signal est constamment à l'état bas, 0V, à 255, le signal est constamment à l'état haut, 3,3V ou 5V selon l'Arduino, soit un rapport cyclique de 100%. La PWM trouve son emploi lorsqu'il s'agit de commander un moteur électrique à courant continu, ou une intensité lumineuse, par exemple une LED.

SPI - broches 10 (SS), 11 (MOSI), 12 (MISO), 13 (SCK). SPI signifie Interface Périphérique Série. Ces broches prennent en charge la communication SPI à l'aide de la bibliothèque SPI.

Broches d'entrée analogique - les Arduinos sont pourvus de 6, pour l'Uno, à 16 entrées analogiques, pour le Mega. Ce sont les broches étiquetées A suivie d'un nombre. Chacune fournissant 10 bits de résolution (c'est-à-dire 1024 valeurs différentes). Ces broches peuvent lire le signal d'un capteur analogique (comme un capteur de température) et le convertir en une valeur numérique utilisable. Le MCU des Arduinos

possède une référence de tension interne de 1,1V qui peut être sélectionné à la place de la tension d'alimentation. Cela permet une meilleure résolution, environ 1mV, sur les faibles valeurs de tension à convertir. Il est également possible de sélectionner la tension fournie sur la broche AREF comme tension de référence.

I2C - les broches A4 (SDA) et A5 (SCL) sont utilisées pour communiquer en utilisant la Bibliothèque Wire. Elles constituent le bus I2C. L'I2C est l'un des bus de communication disponibles pour dialoguer soit avec un autre Arduino soit avec des circuits périphériques disposant de ce bus. 2 broches sont employées pour l'I2C, l'horloge et l'adresse/donnée. De nombreux circuits périphériques disposent d'une interface I2C.

Broche de Réinitialisation - mettre cette ligne à l'état LOW réinitialise le microcontrôleur. Généralement utilisé pour ajouter un bouton de réinitialisation aux blindages qui bloquent celui sur la carte.

VI.3.5 Indicateurs LED

Témoin lumineux d'alimentation - juste au-dessous et à la droite du mot « UNO » sur la carte, il y a une LED minuscule à côté du mot « ON ». Cette LED doit s'allumer lorsqu'on branche l'Arduino à une source d'alimentation. Si cette lumière ne s'allume pas, il y'a une bonne chance que quelque chose ne va pas.

LED embarquée- Il y a une LED intégrée connectée à la broche numérique 13. Lorsque la broche est à valeur HAUT, la LED est allumée, quand la broche est BASSE, elle est éteinte. Ceci est utile pour vérifier rapidement si la carte n'a pas de problème, car certaines cartes ont un programme préchargé d'un simple clignotant LED en elle.

Voyants TX et RX - Ces LEDs donnent de belles indications visuelles chaque fois que la carte Arduino recevra ou transmettra des données (par exemple lors du chargement d'un nouveau programme sur la carte).

VI.3.6 Protection par fusible intégré

L'Arduino Uno a un fusible ré-armable automatiquement qui protège le port USB des court-circuits ou des surcharges (en général, le courant maximum que peut fournir une prise USB est de 500 mA).

VI.3.7 Auto-reset

Le circuit imprimé de l'Uno contient une piste qui peut être coupée pour supprimer

l'auto-reset. Bien qu'il n'est pas conseiller de procéder à cette modification, sous peine de perdre certaines fonctionnalités (notamment pour le téléchargement de code). Cette opération n'est pas irréversible : il suffit de ressouder les deux extrémités de cette piste ensemble. Il y a un marquage qui précise l'endroit où opérer : "RESET-EN". Il est aussi possible de supprimer l'auto-reset en connectant une résistance de 110 ohms entre le 5V et la ligne « reset ».

VI.3.8 Microcontrôleur ATmega328

Le microcontrôleur utilisé sur la carte Arduino UNO est un microcontrôleur ATMega328 au format DIP 28 pins. C'est un microcontrôleur ATMEL de la famille AVR 8bits (16MHz). Ce microcontrôleur est préchargé avec le bootloader Arduino UNO. L'ATMEGA328-PU est un CMOS basse puissance basé sur l'architecture RISC optimisée AVR. En exécutant des instructions puissantes en un seul cycle d'horloge, l'ATMEGA328-PU atteint des débits approchant les 1MIPS par MHz, permettant aux concepteurs système d'optimiser la consommation d'énergie par rapport à la vitesse de traitement.

- 131 instructions puissantes - la plupart avec une exécution en un seul cycle d'horloge
- Registres 32 × 8 à usage général
- Fonctionnement entièrement statique
- Débit jusqu'à 20MIPS à 20MHz
- Multiplicateur en 2 cycles sur la puce
- Cycles d'écriture/suppression : EEPROM 10 000 flash/100 000
- Section de code de démarrage en option avec bits de verrouillage indépendants
- Programmation in-situ par le programme de démarrage sur la puce
- Opération de lecture-écriture vraie
- Verrou de programmation pour la sécurité du logiciel
- Supporte la librairie Atmel® QTouch®
- Boutons tactiles capacitifs, curseurs et molettes
- Acquisition QTouch et QMatrix®
- Jusqu'à 64 canaux de mesure
- Deux compteurs/timers 8 bits avec prédiviseur séparé et mode de comparaison

- Compteur/timer 16 bits avec prédiviseur séparé, mode de comparaison et mode de capture
- Compteur temps réel avec oscillateur séparé
- Six canaux PWM
- USART série programmable
- SPI Maître/Esclave

VI.3.9 Microcontrôleur ATmega16U2

L'ATMega16u2 prend en charge la connexion USB. Le nouvel Arduino Uno R3 utilise un ATmega16U2 au lieu du 8U2 trouvé sur d'autres cartes anciennes. Celui-là permet des vitesses de transfert plus rapides et plus de mémoire.

VI.3.10 Connecteur ICSP

ICSP est synonyme de programmation série sur circuit. Il y a deux connecteurs ICSP sur la carte un pour ATmega16U2 et l'autre pour ATmega328. Ils sont utilisés pour mettre à jour ou charger le micrologiciel dans le microcontrôleur.

VI.3.11 Les mémoires

Mémoire flash

Elle est de 32 kB dont 0,5 kB sont utilisés pour le bootloader, son programme de démarrage. Cette mémoire est l'équivalent du disque dur pour l'ordinateur. C'est la place réservée pour le stockage du programme.

SRAM

Équivalent à la mémoire RAM, elle sert à stocker le résultat des variables. Sa taille est de 2 kB. Comme la RAM, cette mémoire est volatile, à l'extinction de la carte, les valeurs disparaissent.

EEPROM

Mémoire en dur, elle permet de sauvegarder des valeurs de variables et ceci même à l'extinction de la carte. Sa taille est de 1 kB. Ces mémoires sont limitées en taille. Il s'agit donc d'optimiser au maximum afin qu'une carte Uno puisse recevoir le programme et l'exécuter correctement.

VI.3.12 Utilisation

Notre projet de thèse utilise la carte Arduino nano et la carte Arduino MEGA pour plus de détails voire partie étude et réalisation.

Afin d'utiliser les cartes Arduinos le logiciel Arduino IDE est obligatoire. Comme a été dit au début l'Arduino est une plateforme open-source d'électronique programmée basée sur une simple carte à microcontrôleur (de la famille AVR) et un logiciel servant d'interface entre l'ordinateur et l'Arduino créant un environnement de développement qui va permettre d'écrire, de compiler et de transférer un programme vers la carte Arduino. L'environnement Arduino simplifie la façon de travailler avec les microcontrôleurs : il est peu coûteux, multiplateforme (Windows, Mac, GNU/Linux), son environnement de programmation est simple - clair - open source - extensible.

Pour écrire, modifier ou injecter (téléverser) les programmes (le code) dans le microcontrôleur, on utilisera une interface graphique appelée IDE (Integrated Development Environment). Cet environnement de développement intégré est téléchargeable sur le site Arduino.

VI.4 Arduino IDE

L'environnement de développement intégré (IDE) d'Arduino (figure 12) est une application multiplateforme écrite en langage de programmation Java. L'Arduino IDE dérive de Processing et les projets Wiring pour le développement de logiciels. Il inclut un éditeur de code avec des fonctionnalités telles que la mise en surbrillance de syntaxe, l'appariement des accolades et l'indentation automatique, et est également capable de compiler et de télécharger des programmes sur la carte en un seul clic. Un programme ou un code écrit pour Arduino est appelé "sketch".

Les programmes Arduino sont écrits en C ou C ++. L'IDE Arduino est livrée avec une bibliothèque de logiciels appelée « Wiring » du nom de projet d'origine Wiring, ce qui facilite beaucoup les opérations d'entrée / sortie courantes. Les utilisateurs n'ont besoin que de définir deux fonctions (void setup() et void loop()) pour créer un programme exécutable.

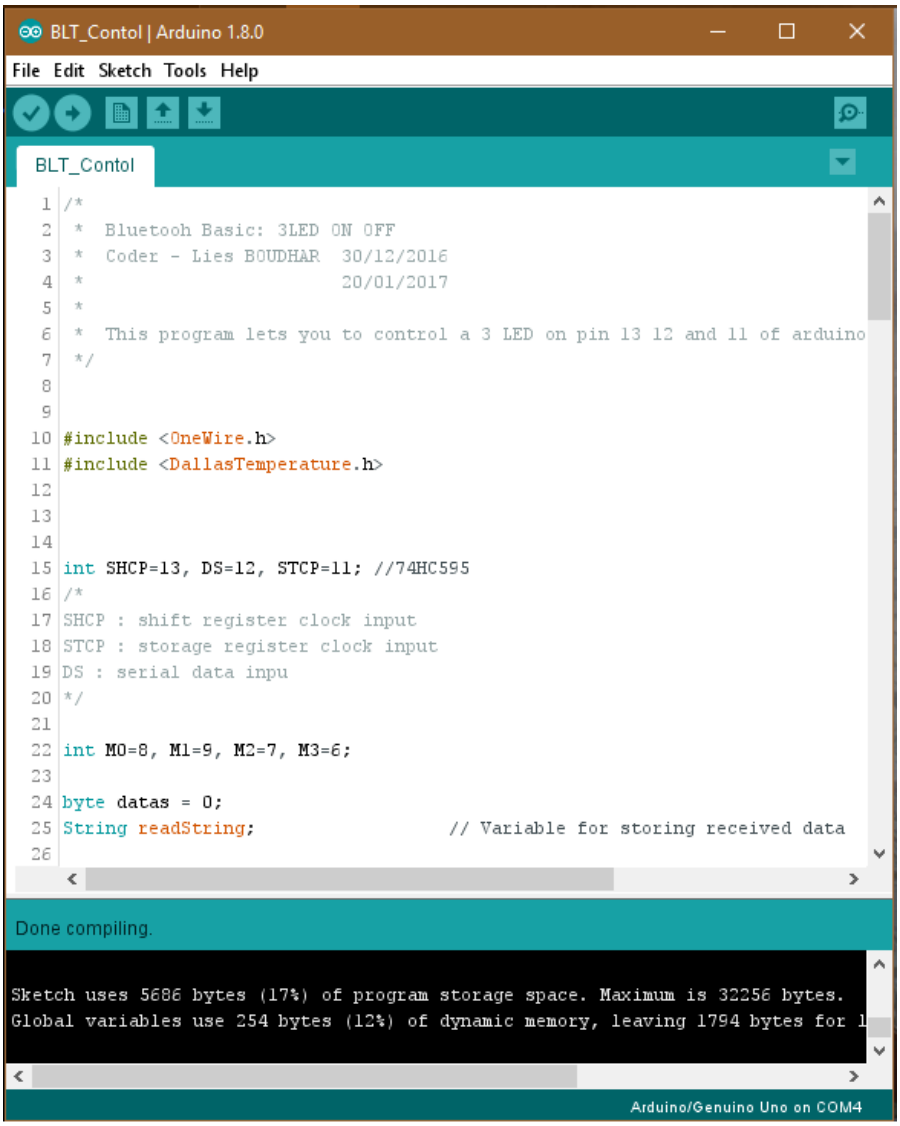


Figure 12 : Arduino IDE 1.8.0

On trouve :

- Barre de menus
- Barre d'outils (raccourcis les plus utiles de fonctions se trouvant dans les menus) : Vérifier le code | Téléverser le code dans le microcontrôleur | Nouveau code | Ouvrir | Enregistrer | Moniteur série (permet d'afficher des messages textes reçus de la carte Arduino et/ou d'envoyer des caractères vers la carte Arduino).
- Zone d'écriture du programme
- Zone de notification lors du téléversement où sera indiqué ce qui se passe et les éventuels messages d'erreur
- Barre de notification du port de communication employé (utile si on utilise plusieurs Arduino).

VI.5 Android

Android est un système d'exploitation open source a noyau Linux puissant, moderne, simple et flexible. Android est développé avec le langage de programmation java et ou C/C++, il s'adapte à des architectures différentes à processeur ARM.

Android offre aux développeurs la possibilité de créer et d'améliorer des applications tournant sur sa plateforme.

Le noyau Linux lui fournit une grande mémoire, la gestion de processus, le modèle de sécurité, le soutien des bibliothèques partagé...etc.

Android Studio qui est un IDE qui contient le kit de développement (SDK) d'Android, un ensemble complet d'outils de développement d'applications Android.

Le diagramme de la figure 13 illustre les principaux composants du système d'exploitation Android.

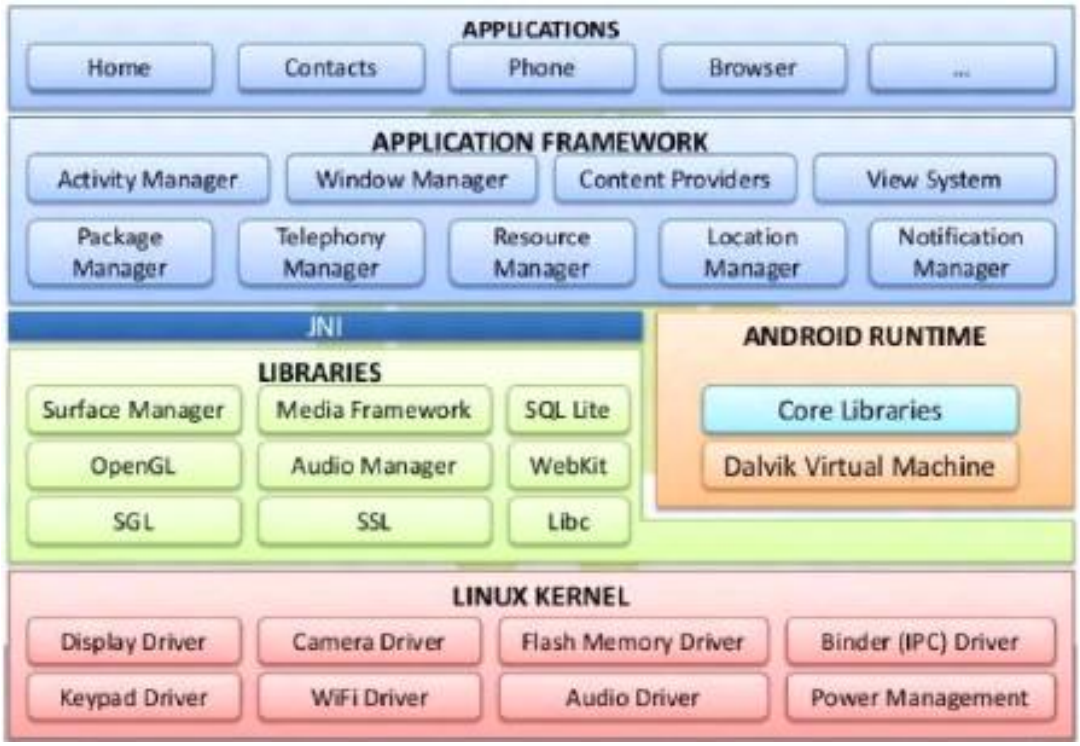


Figure 13 : Arduino IDE 1.8.0

VI.5.1 Applications

Android est fourni avec un ensemble de programmes de base, nommés applications natives, permettant d'accéder à plusieurs fonctionnalités comme les Mails, les SMS, le téléphone, le calendrier, les photos, le Web, ... etc. Ces applications sont

développées à l'aide du langage de programmation Java. Pour l'utilisateur final, c'est la seule couche accessible et visible.

VI.5.2 Le framework (Application Framework)

Android fourni une plateforme de développement ouverte, il offre aux développeurs la possibilité de créer des applications extrêmement riches et innovantes.

VI.5.3 Les bibliothèques (Libraries)

Android inclut un ensemble de bibliothèques C et C++ utilisées par de nombreux composants de la plateforme Android. Ces bibliothèques sont accessibles au développeur par l'intermédiaire du framework Android. En effet, le framework Android effectue, de façon interne, des appels à des fonctions C/C++ beaucoup plus rapides à exécuter que des méthodes Java standard.

VI.5.4 Moteur d'exécution Android (Android Runtime)

Chaque application Android s'exécute dans son propre processus, avec sa propre instance de machine virtuelle Dalvik. Android inclut un ensemble de bibliothèques qui fournit la plupart des fonctionnalités disponibles dans les bibliothèques de base du langage de programmation Java.

VI.5.5 Noyau Linux (Linux Kernel)

Android repose sur un noyau Linux qui gère les services du système, comme la sécurité, la gestion de la mémoire et des processus, la pile réseau et les pilotes. Il agit également comme une couche d'abstraction entre le matériel et la pile logicielle.

VII. Conclusion

Cette étude bibliographique est essentielle dans ce travail de recherche, car par laquelle on a peut conclure les choses suivantes.

L'agriculture de précision implique une production agricole basée sur la gestion de l'information afin de soutirer des avantages économiques, agronomiques et environnementaux. Une révolution technique est apparue dans le monde agricole qui aura un impact dans un avenir très proche sur l'agriculture de précision, cette révolution que j'appellerai agriculture numérique est la fusion des machines, des capteurs et des modèles.

Les propriétés mécaniques et physiques du sol ont une influence sur la croissance des plantes et sur la consommation d'énergie. La méthode conventionnelle de la collecte des données sur les propriétés du sol consiste à prendre des échantillons du sol sur le champ et de les envoyer au laboratoire d'analyses. Cette méthode en plus de prendre beaucoup de temps, elle exige beaucoup de travail et est Coûteuse, cette méthode ne produit pas de vraies cartes des propriétés du sol, car les points d'échantillonnage sont peu peuplés et ne suffisent pas à identifier la variabilité dans l'espace des propriétés du sol.

Les objectifs de cette étude étaient la conception d'un système de capteurs pour l'analyse en temps réel des propriétés physico-mécanique du sol, pour la mesure intégrée et en continu de la teneur en eau du sol, de la résistance mécanique du sol et être ouvert à d'autres propriétés du sol. Ces mesures permettent de réaliser une carte des propriétés du sol pour être réutilisé dans le cadre de l'agriculture de précision.

Deuxième partie :

MATERIELS ET METHODES

Introduction

Le fonctionnement du système qui est embarqué sur un tracteur se base sur les mesures en temps réel du glissement, du rayon dynamique des roues motrices, de la force de résistance à la traction, de l'humidité de sol, et sur les équations développer par plusieurs chercheurs et approuver, et qui mettent en relation la résistance du sol avec le glissement, la force de résistance à la traction, l'humidité, la porosité, et la texture.

La texture du sol qui permet d'indiquer les tendances du sol quant à ses qualités physiques est une constante à l'échelle humaine, on la prend comme constante dans le système pour un sol donné. Les variables mesurables en temps réel sont l'humidité, le rayon dynamique des roues motrices du tracteur, la force de résistance à la traction, les vitesses de déplacement théorique et réel du tracteur.

Pour la mesure des paramètres indiqués comme variable, on a dû réaliser un système électronique composé de plusieurs modules selon le besoin. Ce système comporte une carte principale à microcontrôleur l'Arduino méga 2560 sur laquelle vient se griffer des modules nécessaires pour le bon fonctionnement de l'ensemble, des cartes annexes comportant elles aussi des modules de mesures et de communication avec la carte principale. Dans cette grande partie, on explique la constitution matérielle du système, le fonctionnement de chaque module qui le constitue, leur rôle ainsi que son utilité dans l'ensemble. On détaillera la composition de chaque module, et on expliquera comment est fait le firmware qui le fait fonctionner. Cette partie sera divisée en trois chapitres, un chapitre pour étudier le développement matériel, un autre pour étudier le développement des programmes qui font fonctionner les modules et le système tout entier et un chapitre pour étudier les modèles mathématiques que le système utilise.

I. Les modèles mathématiques utilisés par le système

I.1 Détermination de la résistance du sol

I.1.1 Modèle utilisé

Dans cette section, nous expliquons comment le système par des applications sous Arduino et sous Android utilise les équations pour déterminer la résistance du sol. Brixius (1987) a développé un modèle plus générale pour les caractéristiques de traction des pneus à carcasse diagonale et radiale. Son approche est basée sur l'indice de mobilité appelé Bn qui est donné par l'équation (01) :

$$Bn = \frac{CI \cdot b \cdot d}{W}\left(\frac{1+K_1\frac{\delta}{h}}{1+K_2\frac{b}{d}}\right) \qquad (01)$$

Bn Indice de mobilité (mobility number) sans unité

CI Indice de cône (cone index) en kPas

W Poids Dynamique (vertical wheel load) en kN

δ Déflection du pneu (tyre deflection) en m

b Largeur du pneu (tyre section width) en m

d diamètre du pneu (tyre diameter) en m

h hauteur de la section du pneu (tyre section height) en m

$$\frac{T}{r \cdot W} = a_1(1 - e^{-a_2 \cdot Bn})(1 - e^{-a_3 \cdot S}) + a_4 \qquad (02)$$

$$\frac{MR}{W} = \frac{a_5}{Bn} + a_4 + \frac{a_6 \cdot S}{\sqrt{Bn}} \qquad (03)$$

Ou K_1, K_2, a_1, a_2, a_3, a_4, a_5, et a_6 sont des coefficients de traction selon le type du pneu (tableau 04)

$$\frac{P}{W} = \frac{T}{r \cdot W} - \frac{MR}{W} \qquad (04)$$

W = charge verticale sur la roue motrice ou Poids Dynamique (force réactive)

MR = force de résistance au roulement

T = couple au niveau de la roue motrice

P = force de traction nette

Les équations (02) (03) (04) donnent l'équation (05) :

$$\frac{P}{W} = \frac{T}{r \cdot W} - \frac{MR}{W} = a_1(1 - e^{-a_2 \cdot Bn})(1 - e^{-a_3 \cdot S}) + a_4 - \frac{a_5}{Bn} - a_4 - \frac{a_6 \cdot S}{\sqrt{Bn}}$$

$$\frac{P}{W} = a_1(1 - e^{-a_2 \cdot Bn})(1 - e^{-a_3 \cdot S}) - \left(\frac{a_5}{Bn} + \frac{a_6 \cdot S}{\sqrt{Bn}}\right) \qquad (05)$$

Soit : $A = (1 - e^{-a_3 \cdot S})$ et $C = a_6 \cdot S$

L'équation (05) devient l'équation (06)

$$\frac{P}{W} = a_1(1 - e^{-a_2 \cdot Bn})A - \left(\frac{a_5}{Bn} + \frac{C}{\sqrt{Bn}}\right)$$

$$\frac{P}{W} = a_1 \cdot A - \frac{a_1 \cdot A}{e^{a_2 \cdot Bn}} - \frac{a_5}{Bn} - \frac{C}{\sqrt{Bn}} \qquad (06)$$

L'équation (01) et l'équation (06) sont utilisées pour prédire la variation de la résistance du sol en temps réel avec des mesures en continu. Tout d'abord, le système calcule Bn en résolvant l'équation (06) (algorithme page 44) et dans un second temps, le système calcule CI à partir de l'équation (07) avec les coordonnées GPS.

$$\frac{Bn \cdot W}{b \cdot d\left(\frac{1 + K_1 \frac{\delta}{h}}{1 + K_2 \frac{b}{d}}\right)} = CI \qquad (07)$$

Recherche de Bn par la méthode d'itération

La méthode

La méthode consiste à introduire une suite (Bn_n) d'approximations successives de l'équation $f(Bn) = 0$.

$$\frac{P}{W} = a_1 \cdot A - \frac{a_1 \cdot A}{e^{a_2 \cdot Bn}} - \frac{a_5}{Bn} - \frac{C}{\sqrt{Bn}} \qquad (06)$$

$$\frac{a_1 \cdot A}{e^{a_2 \cdot Bn}} + \frac{a_5}{Bn} + \frac{C}{\sqrt{Bn}} + \frac{P}{W} - a_1 \cdot A = 0$$

Soit la fonction $f(B_n)$ tel que :

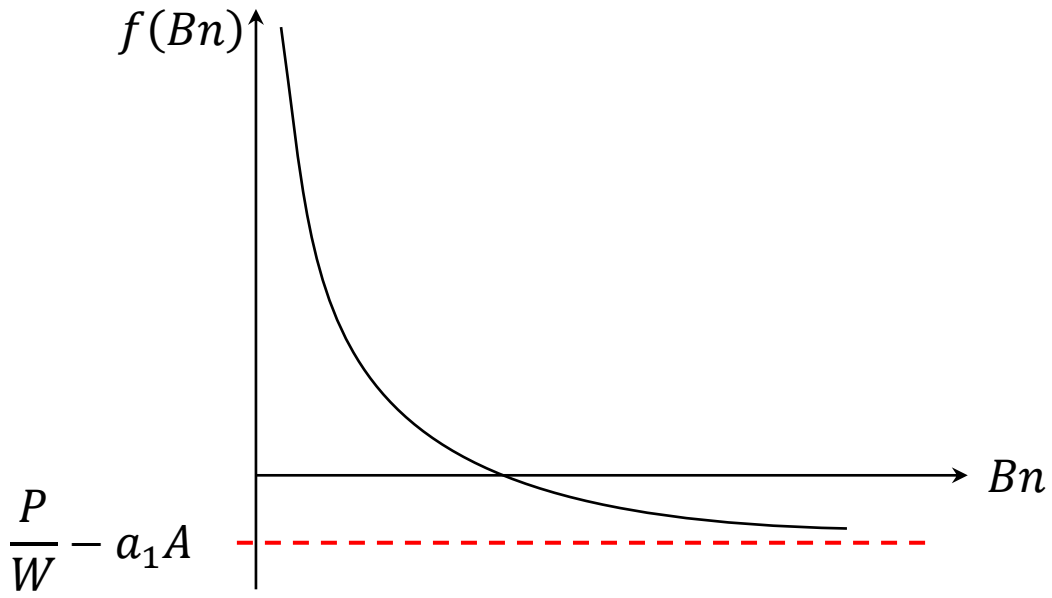
$$f(Bn) = \frac{a_1 A}{e^{a_2 \cdot Bn}} + \frac{a_5}{Bn} + \frac{C}{\sqrt{Bn}} + \frac{P}{W} - a_1 A = 0$$

Calcul de la dérivée de $f(Bn)$

$$f'(Bn) = \frac{a_1 a_2 A}{e^{a_2 \cdot Bn}} - \frac{a_5}{Bn^2} - \frac{C}{2\sqrt{Bn^3}}$$

$f'(Bn) < 0 \; pour \; Bn > 0 \; alors \; f(Bn) \; est \; decroissante$

$$\lim_{B_n \to 0} f(Bn) = +\infty \qquad et \qquad \lim_{B_n \to +\infty} f(Bn) = \frac{P}{W} - a_1 A$$



Si $\frac{P}{W} - a_1 A$ est ≥ 0 la fonction $f(Bn)$ n'admet pas de solution

Algorithme de recherche de Bn solution de $f(Bn) = 0$

On commence par mètre Bn a un et on l'introduit dans l'équation $f(Bn)$ et on vérifier si $f(Bn)$ est inférieure à zéro deux cas ce présentent :

- $f(Bn)$ est supérieure à zéro alors on incrémente Bn de un et on refait le calcul de $f(Bn)$ jusqu'à ce que $f(Bn)$ soit inférieure à zéro.
- $f(Bn)$ est inférieure à zéro, on soustrait un a Bn et on l'incrémente de 0,1 et on refait le calcul de $f(Bn)$ en incrémentant Bn de 0,1 jusqu'à ce que $f(Bn)$ soit inférieure à zéro, lorsque $f(Bn)$ sera inférieure à zéro, on soustrait 0,1 a Bn et en l'incrémente de 0,01 et on refait le calcul de $f(Bn)$ on incrémentant Bn de 0,01 jusqu'à ce que $f(Bn)$ soit inférieure à zéro, lorsque $f(Bn)$ sera inférieure à zéro, on arrête l'opération d'itération et la valeur finale de Bn – 0,01 est la solution de la fonction $f(Bn)$.

Le programme en C/C++ pour le calcul de Bn et de CI

```
#include <iostream>
#include <stdio.h>
using namespace std;
double A, C, U, Z, a1, a2, a3, a4, a5,a6;
double K1 = 5.0, K2 = 3.0;

double b = 0.25,  d = 1.5,  δ = 9.0,  h = 45.0;
int TyreType = 1;                // si 1 diagonal sinon radial
double S  = 0.2,  P  = 7.0,  W  = 36.0;  //pour vérification de fonctionnement du programme

double ConeIndex(double Bn){
      double CI=(Bn*W)/(b*d)*((1+K1*(δ/h))/(1+K2*(b/d)));
         return CI;
}
double BrixiusNum(double Bn){
            double Y = Z/exp(a2*Bn)+(a5/Bn)+(C/sqrt(Bn))+ U - Z;
         return Y;
}
double Iteration( double x, double epsilon, int &n){
         while (BrixiusNum(x) > 0){
            x = x + epsilon;
            n = n+1;
         }
         return x;
}
int main(){
   double x = 1;
   int n = 0;
   if (TyreType == 1){       // diagonal
      a1=0.88, a2=0.1, a3=7.5, a4=0.04, a5=1.0, a6=0.5;
   }else{       // radial
      a1=0.88, a2=0.1, a3=9.5, a4=0.032, a5=0.9, a6=0.5;
    }
   C = a6 * S;
   A = 1 - exp(-a3 * S);
   Z = a1*A;
   U = P/W;
   x = Iteration(x, 1.0, n);
   x = Iteration(x-1, 0.1, n);
   x = Iteration(x-0.1, 0.01, n);
   x = x - 0.01;
   printf ("Bn=%f\n n=%i\n", x, n );
   double CI = ConeIndex(x);
   printf("CI=%f\n", CI );
   return 0;
}
```

Pour trouver **Bn** on doit avoir **P**, **W**, **r** rayon dynamique et déflection **δ** de la roue motrice, **S** coefficient de patinage, ainsi que les coefficients K_1, K_2, a_1, a_2, a_3, a_4, a_5, et a_6 (tableau 4).

Pour mesurer le glissement, divers chercheurs ont utilisé différentes techniques comme l'effet Doppler radar, circuits électroniques utilisant des photo-capteurs, etc. pour une mesure précise du glissement ses techniques s'avèrent compliqués et coûteuses (Raheman and Jha, 2006).

Pour le calcul de **S**, le système utilise : $S = 1 - \frac{V}{\omega r}$ (08)

Le système calcule **ω** la vitesse angulaire de la roue motrice en mesurant **N** en tours par minute, **r** le rayon dynamique de la même roue en (m) et utilise la méthode GPS DGPS pour calculer la vitesse réelle V_r en (m/s). Pour mesurer **N** la vitesse de rotation des roues motrices (voir mesure de la vitesse rotation des roues motrices), le système utilise la position angulaire du gyroscope. Lorsque le gyroscope passe à travers la même position angulaire, il en résulte un tour, un demi-tour ou un quart de tour, mesurée pendant une durée t, ainsi le système calcule le nombre de tours par minute.

$\omega = 2\pi N$ et $V_t = \omega r$

Largeur **b** du pneu en (m), diamètre **d** en (m) et hauteur de section du pneu **h** en (m) sont des constantes selon les types de pneus (figure 17, annexe A) et sont introduits manuellement dans le système par le clavier. La déformation du pneu en charge **δ** (δ= rayon statique - rayon dynamique) est calculée par le système (voir mesure du rayon dynamique). **P** force de traction nette mesurée directement par le HX711 amplificateur pour jauge de contrainte reliée à la carte principale (voir mesure de la force de traction). Le Poids Dynamique **W** est déterminé comme suit :

Tableau 4: coefficients de traction pour pneus à carcasse diagonale et radiale

Coefficients de traction	Diagonale [a]	Radial	
		Valeur recommandée [a]	Valeur moyenne
K_1	5	5	5
k_2	3	3	3
a_1	0,88	0,88	0,88
a_2	0,1	0,1	0,1
a_3	7,5	8,50–10,50	9,5
a_4	0,04	0,03–0,035	0,032
a_5	1	0,9	0,9
a_6	0,5	0,5	0,5

a: de Brixius (1987).

I.1.2 Détermination du Poids Dynamique

Deux méthodes se présentent pour la détermination du poids dynamique ou charge verticale sur la roue motrice **Wr**, par mesure directe ou indirecte pour tracteur deux roues motrices (dans notre cas, on utilise la méthode indirect).

Par mesure directe :

La mesure directe consiste à placer les jauges de contraintes au niveau des essieux arrière de chaque côté, mais cette méthode n'est pas l'objet de ce travail.

Par mesure indirecte :

C'est-à-dire par utilisation du modèle dynamique du tracteur (voir figure 14), qui détermine les équations de mouvement longitudinal d'un tracteur agricole en conditions de travail sur un terrain agricole.

Les constantes dans le cas général sont :

Wt poids du tracteur,

Wo poids du système a dent et jauge,

X l'empattement du tracteur ou distance du centre de la roue arrière au centre de

la roue avant,

Xr distance du centre de gravité du tracteur à son essieu arrière,

Xf distance du centre de gravité du tracteur à son essieu avant,

Yg distance du centre de gravité du tracteur par rapport à Or,

Y distance du point d'application de la force de traction P par rapport à Or,

Hc distance du centre de gravité du tracteur au sol,

Xp distance du point d'application de la force de traction P à l'essieu arrière,

Xo distance du centre de gravité du système a dent et jauge à l'essieu arrière,

hp hauteur du point d'application de la force de traction P par rapport au sol,

e_r distance du point d'action de la réaction normale du sol au centre de la roue arrière Or,

e_f distance du point d'action de la réaction normale du sol au centre de la roue avant Of.

Les variables dans le cas général sont :

a vecteur accélération.

a_h composante horizontale du vecteur accélération.

a_n composante normale du vecteur accélération.

R_f force de résistance au roulement des roues avant du tracteur.

R_r force de résistance au roulement des roues arrière du tracteur.

Ra résistance de l'air.

Px composante parallèle de la force de traction.

Py composante normale de la force de traction.

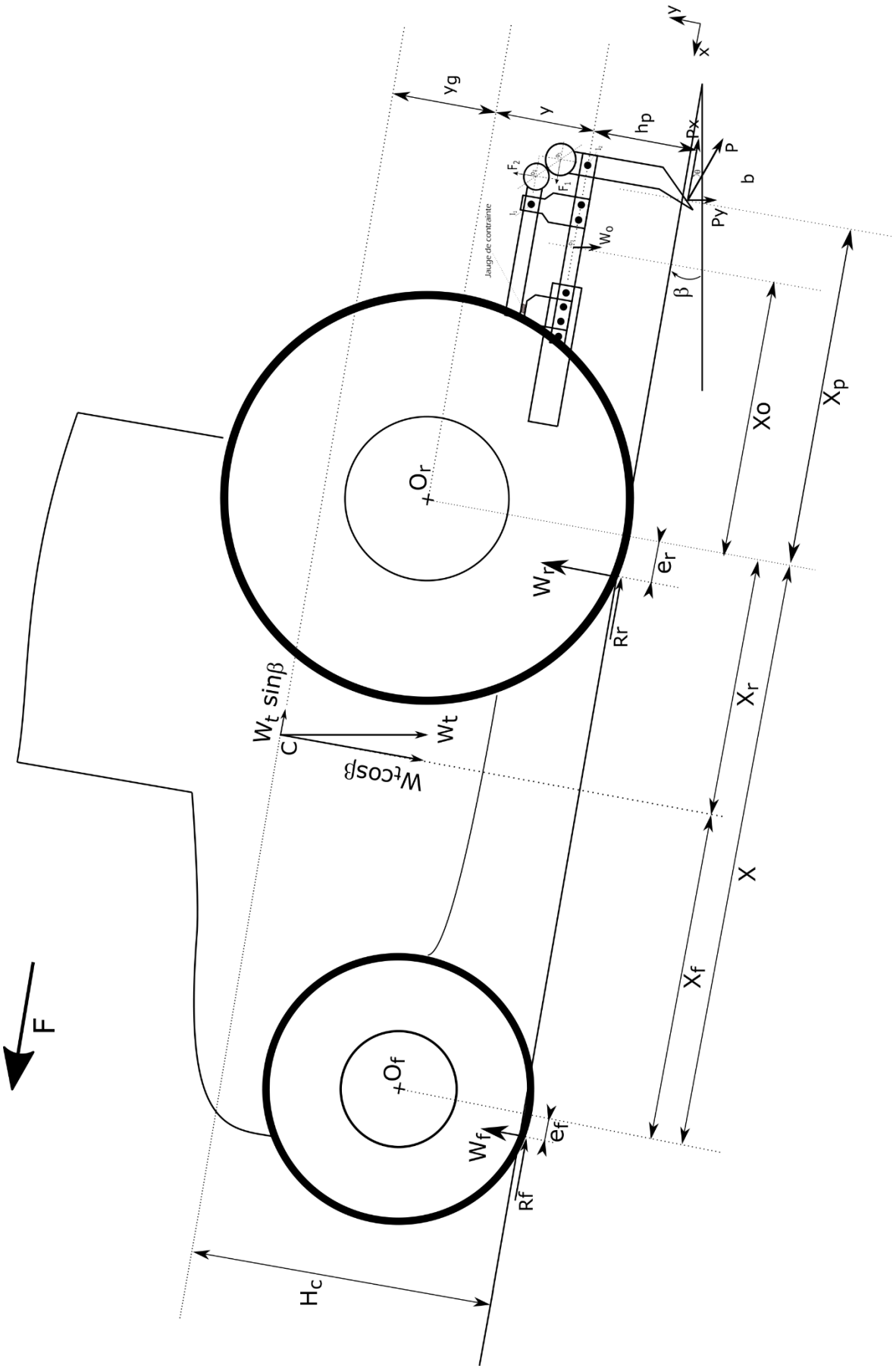


Figure 14 : forces au niveau du tracteur

Équations générales d'équilibre

- Les lignes d'action des forces de traction et les forces de résistances au roulement sont tangentes aux roues.
- Les lignes d'action de la réaction normale du sol passent à travers le centre des roues, dans ce cas e (ef, er) est nul sinon elle est supérieur à zéro.
- Le tracteur se déplace à vitesse constante donc l'accélération est nulle.
- La résistance de l'air est négligeable.

Les équations du mouvement sont:

La somme des forces parallèles à la direction du mouvement donne:

$$F - R_f - R_r - W_t sin\beta - \left(\frac{W_t}{g}\right) a_h - P_x - R_a = 0$$

De même, la somme des forces dans la direction normale à la surface de traction donne :

$$W_f + W_r - W_t cos\beta - \left(\frac{W_t}{g}\right) a_n - P_y = 0$$

Puisque l'accélération est nulle ainsi que la résistance de l'air alors les équations du mouvement deviennent :

$$F - R_f - R_r - W_t sin\beta - P_x = 0$$

$$W_f + W_r - W_t cos\beta - P_y = 0$$

Somme des moments

Somme des moments de toutes les forces par rapport au point d'intersection à l'interface sol-roue arrière :

$$W_f(e_f + X - e_r) + W_t H_c sin\beta - W_t(X_r - e_r)cos\beta + P_x h_p$$
$$+ P_y(X_p + e_r) + W_o h_p \sin\beta - W_o(X_o + e_r)cos\beta + R_a h_a + \left(\frac{W_t}{g}\right) a_h H_c$$
$$+ \left(\frac{W_t}{g}\right) a_n (X_r - e_r) + I_t \alpha_t + I_r \alpha_r + I_f \alpha_f = 0 \qquad (\mathbf{1})$$

Somme des moments de toutes les forces par rapport au point d'intersection à l'interface sol-roue avant :

$$-W_r\left(e_f + X - e_r\right) + W_t H_c sin\beta + W_t\left(X_f + e_f\right)cos\beta + P_x h_p$$
$$+ P_y\left(X_p + X + e_f\right) + W_o h_p \sin\beta + W_o(X_o + e_r)cos\beta + R_a h_a + \left(\frac{W_t}{g}\right)a_h H_c$$
$$+ \left(\frac{W_t}{g}\right)a_n(X_r - e_r) + I_t\alpha_t + I_r\alpha_r + I_f\alpha_f = 0 \qquad (\mathbf{2})$$

Puisque l'accélération est nulle ainsi que la résistance de l'air alors les équations des moments (1) et (2) deviennent :

$$W_f\left(e_f + X - e_r\right) + W_t H_c sin\beta - W_t(X_r - e_r)cos\beta + P_x h_p + P_y\left(X_p + e_r\right)$$
$$+ W_o h_p \sin\beta - W_o(X_o + e_r)cos\beta = 0 \qquad (\mathbf{1})$$

$$-W_r\left(e_f + X - e_r\right) + W_t H_c sin\beta + W_t\left(X_f + e_f\right)cos\beta + P_x h_p + P_y\left(X_p + X + e_f\right)$$
$$+ W_o h_p \sin\beta + W_o(X_o + e_r)cos\beta = 0 \qquad (\mathbf{2})$$

$P_x = Psin\theta$

$P_y = Pcos\theta$

De l'équation (2) on tire Wr

$$W_r = \frac{W_t H_c sin\beta + W_t\left(X_f + e_f\right)cos\beta + P_x h_p + P_y\left(X_p + X + e_f\right) + W_o h_p \sin\beta + W_o(X_o + e_r)cos\beta}{\left(e_f + X - e_r\right)}$$

On prend $e_f = e_r = 0$ négligeable devant X

$$W_r = \frac{W_t H_c sin\beta + W_t\left(X_f\right)cos\beta + P_x h_p + P_y\left(X_p + X\right) + W_o h_p \sin\beta + W_o(X_o)cos\beta}{(X)}$$

Les constantes sont Wt, Wo, X, Xf, Xr, Xp, Xo, Hc, hp.

Les variables sont Px, Py, β, θ.

Pour trouver Wr, les constantes sont introduites une fois pour toute dans l'application, pour les variables elles sont mesurées par le système ou fixées par l'utilisateur du système, on peut aussi fixer la valeur de e_r et e_f déférente de zéro.

Le système commence par la mesure de β, θ et P puis il calcule Px et Py pour enfin il calcule Wr.

I.1.3 Mesure de la force de traction

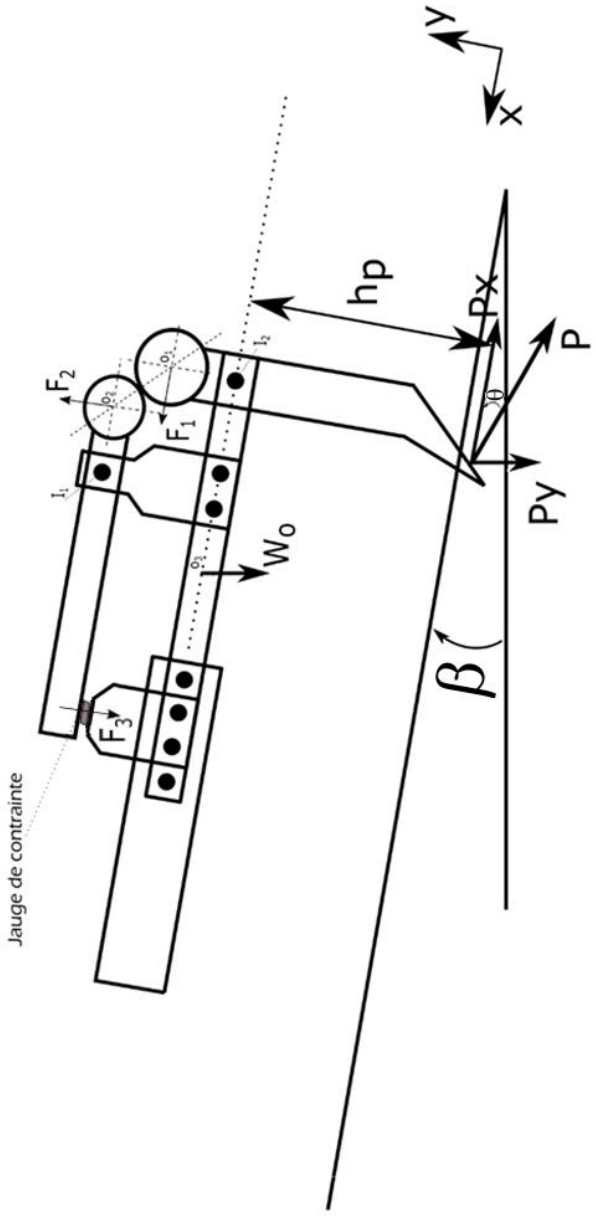


Figure 15 : Système a dent et jauge de contrainte

Pour la mesure de la force de traction P on utilise le système à dent et jauge de contrainte de la figure 15.

L'action de la force de traction P est transférée directement sur la jauge de contrainte grâce aux points de rotation I_1 et I_2. La jauge de contrainte mesure Px et non pas P.

$$P_x \mathrm{b} = F_1 a \;\Rightarrow\; P_x = F_1 \frac{a}{b}$$

$$F_2 \mathrm{c} = F_3 d \;\Rightarrow\; F_2 = F_3 \frac{d}{c}$$

$$F_1 = F_2$$

Ces trois équations donnent

$$P_x = F_3 \frac{a}{b}\frac{d}{c}$$

Si $a = c \; et \; b = d \quad alors \; on \; aura \quad P_x = F_3$

Donc la jauge de contrainte mesure Px et non pas P, pour trouver P on divise Px par sinθ.

L'angle θ est pris comme une constante et est introduit lors de de l'introduction des constantes.

I.1.4 Mesure de la vitesse réelle d'avancement

La mesure de la vitesse réelle d'avancement du tracteur est mesurée par gps, soit avec la carte Arduino Mega par utilisation du module GPS Ublox Neo-6M, ou bien avec Android et dans ce cas-là le système Android doit être sur le tracteur.

La vitesse se définit comme une variation, positive ou négative, de la position d'un mobile par unité de temps. Elle s'exprime dans le Système international d'unités (SI) en mètre par seconde (m/s).

Pour mesurer la vitesse réelle d'avancement du tracteur on utilise la méthode indirecte, aussi appelée méthode positions, est une façon rapide et simple de calculer la vitesse à l'aide des positions obtenues par mesures GPS. Elle consiste à calculer la distance parcourue entre deux positions consécutives de l'antenne GPS puis de faire le rapport entre cette distance et le temps pris pour la parcourir. On prend les

coordonnées gps à chaque intervalle de temps t constant, la vitesse d'avancement sera

$(x_1 - x_0) / t$

Il existe un autre moyen plus efficace pour la mesure de la vitesse d'avancement, est l'utilisation du radar a effet Dopple le MDU1130 (figure 16) qui s'adapte bien à notre système mais il est introuvable sur la marche algérienne.

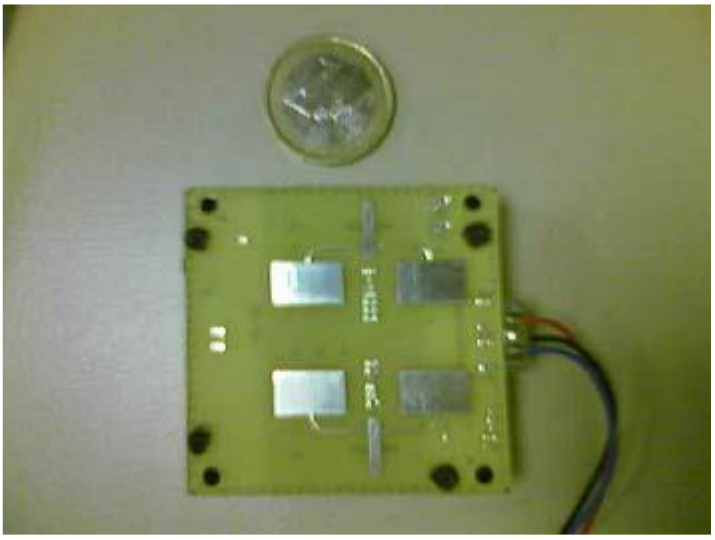


Figure 16 : Radar Doppler MDU1130

I.1.5 Mesure de la vitesse théorique d'avancement

Pour cela on utilise deux capteurs des vitesses de rotation des roues arrières pour la mesure de la vitesse théorique d'avancement.

Plusieurs méthodes sont possibles parmi eux, on cite ; mesure de la vitesse de rotation avec un générateur de fréquence magnétique, mesure de la vitesse de rotation avec un générateur de fréquence optique, mesure de la vitesse de rotation avec un générateur de tension, et enfin avec une série de micro-interrupteurs.

Dans notre cas, aucune des méthodes citées n'a été utilisée. Pour mesurer la vitesse de rotation d'une roue, on utilise la détection de l'angle zéro d'un gyroscope (gy571) en rotation.

On choisit un axe de rotation parmi les trois axes (x, y, z), et on suit le mouvement de rotation du gyroscope entrainé par la roue en rotation. Le nombre de rotations de la roue est le nombre de passages du gyroscope par l'angle zéro dans un temps t.

I.1.6 Mesure du rayon dynamique

Pour la mesure du rayon dynamique on utilise des captures a ultra son de 40kHz pour l'émission et la réception selon le principe suivant.

Un capteur de distance à ultrason utilise des ondes sonores pour mesurer des distances, et se base sur la vitesse du son.

On envoie une impulsion HIGH de 10µs sur la broche TRIGGER du capteur, le capteur envoie alors une série de 8 impulsions ultrasoniques à 40KHz, les ultrasons se propagent dans l'air jusqu'à toucher un obstacle et retourne dans l'autre sens vers le capteur, le capteur détecte l'écho et clôture la prise de mesure.

Les mesures se font en continu et on ne prend en considération que les mesures des angles 0° et 180° du gyroscope (figure 18).

Le rayon dynamique sera donc : $r = r_x + d_i / 2$

rx distance mesurée par le capteur a ultra son

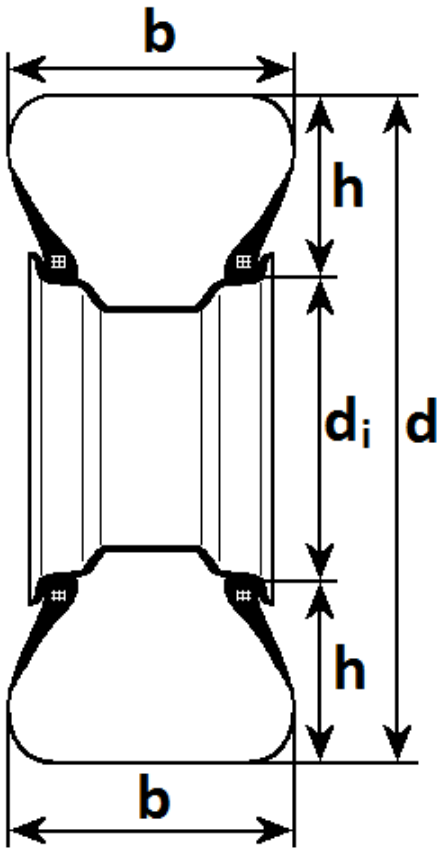


Figure 17 : Dimension d'un pneu

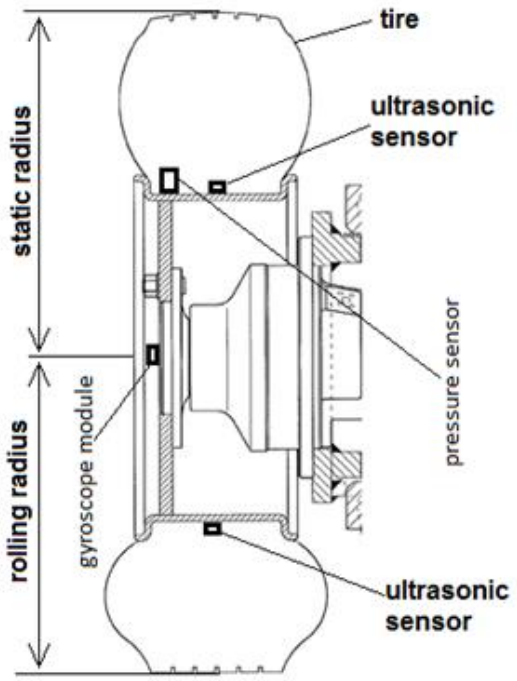


Figure 18 : Emplacement des capteurs à l'intérieur des roues

En plus du cône index on détermine aussi la texture du sol l'humidité du sol et les performances de traction du tracteur.

I.2 Détermination des performances de traction du tracteur

Brixius (1987) a présenté de nouvelles équations qui ont amélioré les prédictions en performances de traction des tracteurs agricoles et a aussi étendu la gamme d'application de ses équations par rapport aux équations de Wismer et Luth. Ces équations sont devenues les plus communément acceptées pour le calcul en performances de traction. Les modèles Brixius sont basés sur les tests de traction du tracteur agricole réalisée par John Deere Co. aux Etats-Unis. Les équations ont été développées en utilisant la technique des courbes d'ajustement, pour prédire la performance de traction des pneumatiques à carcasse diagonale sur des sols à friction cohésive. Le couple moteur, la résistance au roulement, la traction nette et l'efficience de traction sont prédits en fonction de la résistance du sol, de la charge sur les pneus, de la réduction du déplacement (glissement), de la taille des pneus et de la déflexion des pneus. Les équations suivantes sont limitées aux pneus avec un rapport b/d variant

entre 0,1 et 0,7, les déflexions statiques radiaux des pneus variant de 10% à 30% de la hauteur de la section du pneu non défléchie et des valeurs W / (bd) comprises entre 15 et 55kN / m2 (normes ASABE, 2011).

Connaissant B_n (Brixius number) et S (coefficient de patinage) ainsi que les coefficients a_1, a_2, a_3, a_4, a_5, a_6, on peut calculer les paramètres de traction suivants.

- Coefficient de traction brut (GTR: Gross Traction Ratio or torque traction)

$$GTR = \frac{T}{r \cdot W} = \frac{F_b}{R_r} = a_1(1 - e^{-a_2 \cdot B_n})(1 - e^{-a_3 \cdot S}) + a_4$$

- Coefficient de résistance au roulement (MRR: Motion Resistance Ratio)

$$MRR = \frac{MR}{W} = \frac{a_5}{B_n} + a_4 + \frac{a_6 \cdot S}{\sqrt{B_n}}$$

- Coefficient de traction net (GTR: Net Traction Ratio or pull ratio)

$$NTR = GTR - MRR$$

- Efficience de traction (**TE**: Traction Efficiency)

$$TE = \frac{NTR}{GTR}(1 - S)$$

Algorithme (performances de traction)

```
Variable (Bn, S, GTR, MRR, NTR, TE) réelles
Variable (type de pneu) entière
Début
   Début
      Lire " Bn "
      Lire " S "
      Lire " type de pneu "
   Fin

   Début si
      Si type de pneu=radial
      Alors a1=0,88, a2=0,1, a3=9,5, a4=0,032, a5=0,9, a6=0,5,
      Sinon a1=0,88, a2=0,1, a3=7,5, a4=0,04, a5=1, a6=0,5,
   Fin si

   GTR = a1 (1 - exp(- a2 · Bn ) )(1- exp(- a3 · S) ) + a4
```

MRR = a5 / Bn + a4 + (a6·S) / √ (Bn)

NTR = GTR –MRR

TE = NTR / (GTR) (1 - S)

Fin.

Note : Le programme de cet algorithme est écris en java dans l'application Android SoilTractorPrediction.

II. Développement du système

Le système est embarqué sur un tracteur, il est basé sur des mesures en temps réel du glissement, du rayon dynamique des roues motrices, de la force de résistance à la traction, de l'humidité du sol, et sur les équations développées par plusieurs chercheurs et approuvées. Il relie la résistance pénétrométrique du sol au glissement, charge sur les roues motrices, l'effort de traction, l'humidité du sol, la porosité et la texture.

La texture du sol qui peut indiquer les tendances du sol sur ses qualités physiques est constante à l'échelle humaine. Elle est considérée comme une constante dans le système pour un sol donné. Les variables mesurables en temps réel sont l'humidité du sol, le rayon dynamique des roues motrices du tracteur, la force de traction et la vitesse théorique et réelle du tracteur.

Pour la mesure des paramètres spécifiés comme variables, nous avons dû mettre en place un système électronique composé de plusieurs modules selon les besoins. Ce système comprend une carte principale à microcontrôleur, l'Arduino Mega 2560, sur lequel sont connectés plusieurs modules nécessaires au bon fonctionnement de l'ensemble du système. Le système comporte aussi des modules annexes de mesure et de communication permettant la communication avec la carte principale.

II.1 L'unité principale

La carte principale (figure 19) est réalisée autour de L'ARDUINO MEGA, une carte à microcontrôleur basée sur l'ATmega2560, conçu pour des projets plus complexes. Il dispose de 54 broches numériques d'entrée/sortie (dont 15 broches peuvent être utilisés comme sorties PWM), 16 entrées analogiques, 4 UART (ports matériel série), un oscillateur à quartz de 16 MHz, un port USB, une entrée d'alimentation, un connecteur ICSP (InLine Circuit Programming : Programmation du Circuit In-Situ), et un bouton de remise à zéro. Il contient tout le nécessaire pour un système à microcontrôleur. L'ATmega 2560 a 256KB de mémoire FLASH pour stocker le programme dont 8KB utilisé par le bootloader. L'ATmega 2560 a aussi 8 Ko de SRAM et 4 Ko de mémoire EEPROM.

L'Arduino Mega 2560 peut être alimenté soit par le biais de la connexion USB (qui fournit jusqu'à 500mA 5V) ou en utilisant une alimentation externe de 7 à 12 volts. Il peut fournir une tension de sortie de 3,3V.

La carte Arduino Mega 2560 (figures 9,19) peut se programmer avec le logiciel Arduino. Le bootloader du microcontrôleur ATMega2560 permet de modifier le programme sans passer par un programmateur.

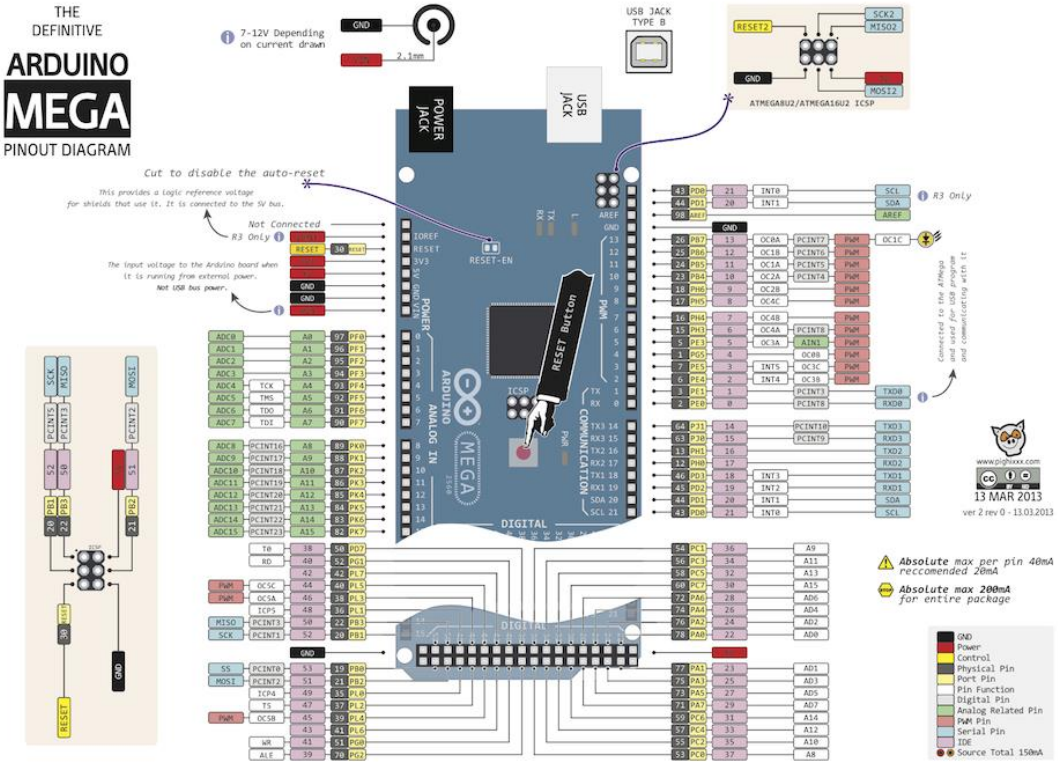


Figure 19 : Brochage de la carte Arduino Mega 2560

La carte principale (figure 20 et 21) contient un firmware dans sa mémoire FLASH qui gère les différents modules de mesures, le dialogue avec l'utilisateur, ainsi que la communication avec les unités annexes et le système Android. Les modules connectés à la carte principale sont :

— Un module Bluetooth HC-06 pour communiquer avec le système Android ;

— Un module RF NRF25L01 pour communiquer avec les cartes annexes ;

— Un module affichage LCD 20X4 montrant des informations aux utilisateurs ;

— Un clavier matriciel 4x4 pour interagir directement avec le système ;

— Une horloge temps réel RTC 1307 ;

— Un conditionneur pour jauge de contrainte HX711 ;

— Une jauge de contrainte ;

— Module inertielle 9 axes Accéléromètre gyroscope Magnétomètre ;

— Un capteur d'humidité du sol pour mesurer la permittivité diélectrique du sol qui est une fonction de la teneur en eau et, avec laquelle le système peut calculer la teneur en eau du sol.

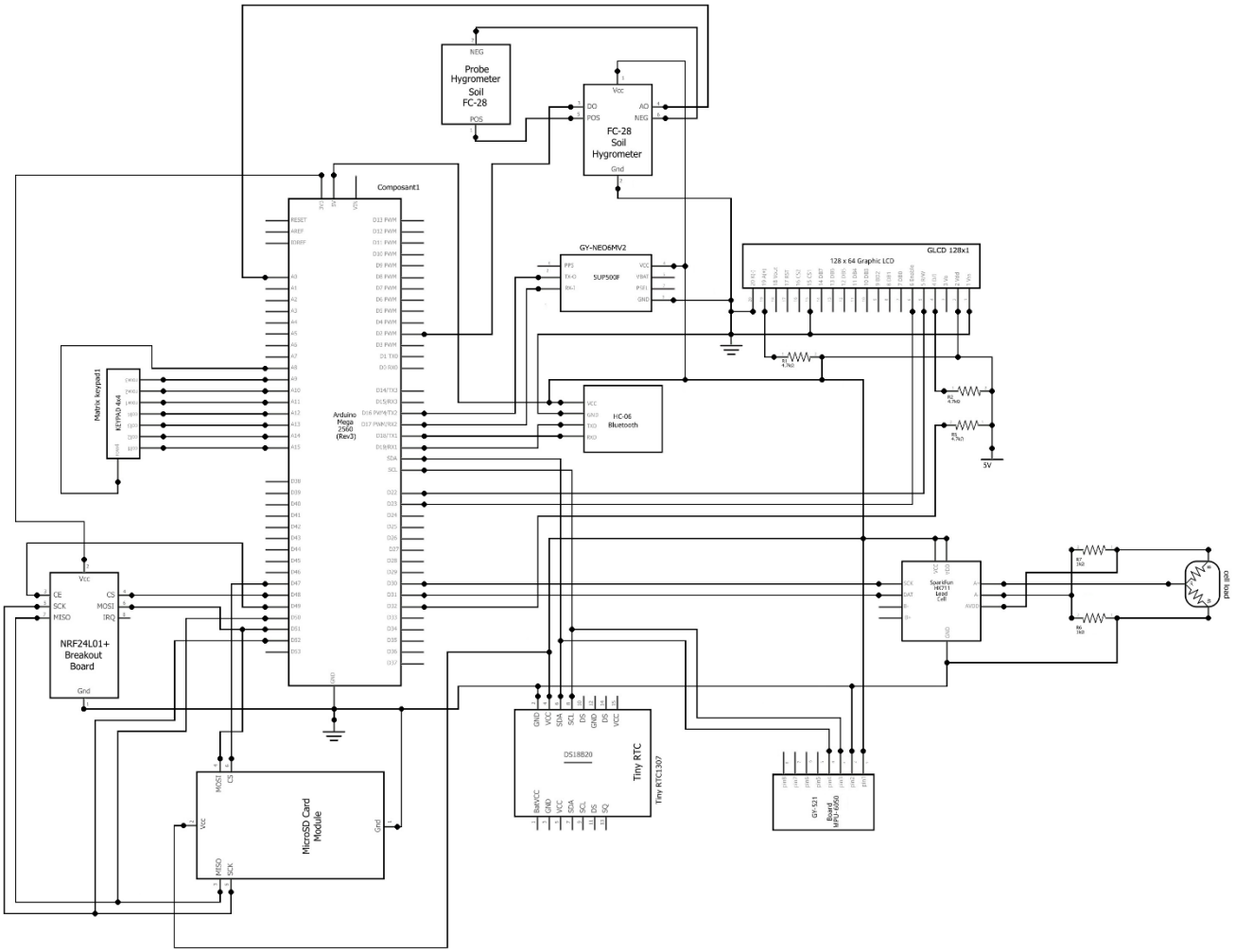


Figure 20 : Schéma électrique de l'unité principale

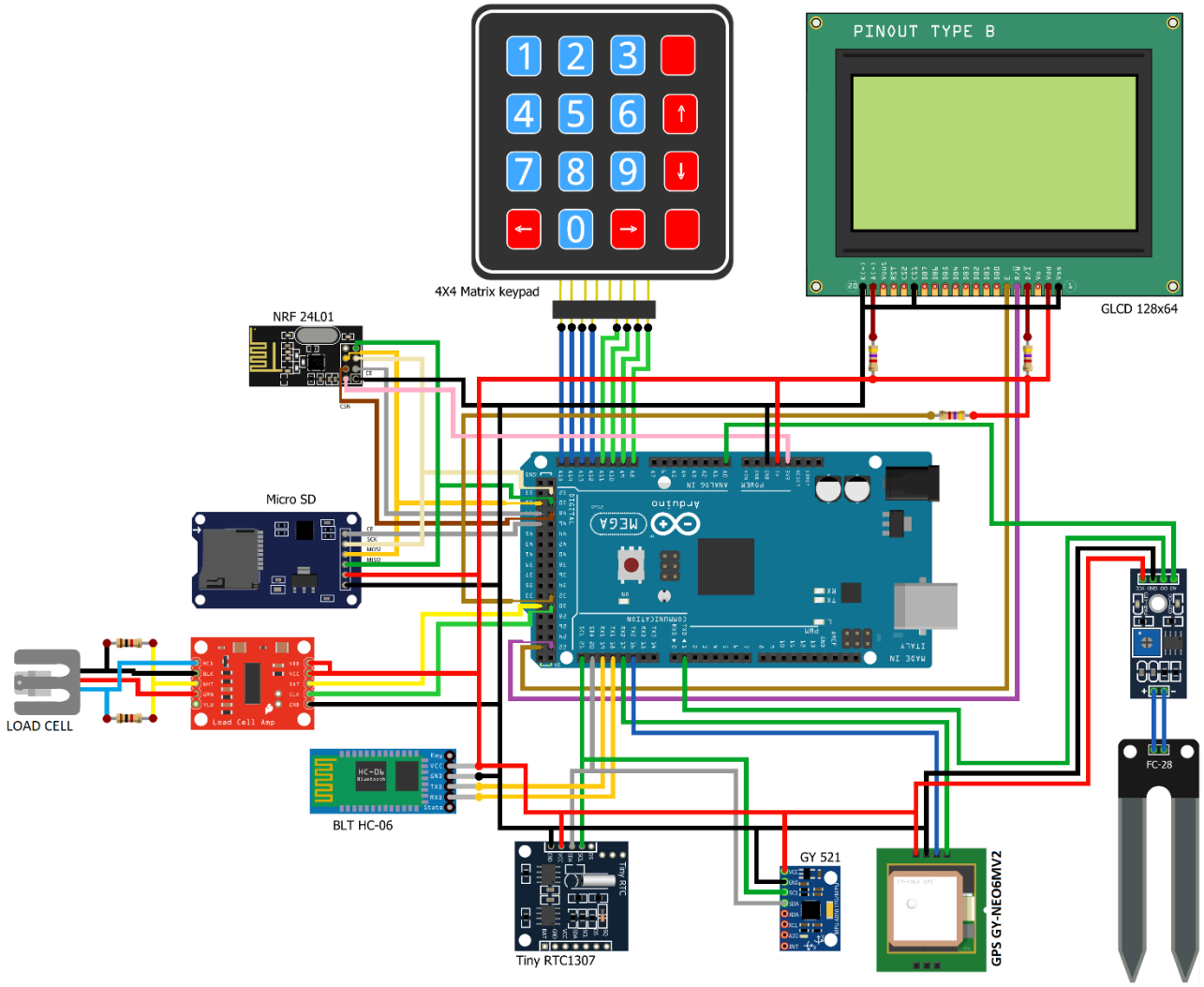


Figure 21 : Schéma de connexion de l'unité principale

II.1.1 Les modules connectés à la carte principale

II.1.1.1 Clavier matriciel 4x4

Le clavier est une matrice de 4 x 4 touches (figure 22), ce type de clavier est composé de 4 lignes et de 4 colonnes à l'intersection desquelles se trouvent les touches qui viennent donner le contact électrique. Une touche enfoncée se traduit par un court-circuit entre la colonne et la ligne à l'intersection.

a) Principe de fonctionnement

Le clavier comporte 16 touches, dont 10 numériques (0-9) et 6 touches marquées # * A B C D. Il est matriciel : au lieu d'avoir 16 fils (1 par touche) et une masse, le multiplexage n'utilise que 8 sorties : 4 lignes et 4 colonnes.

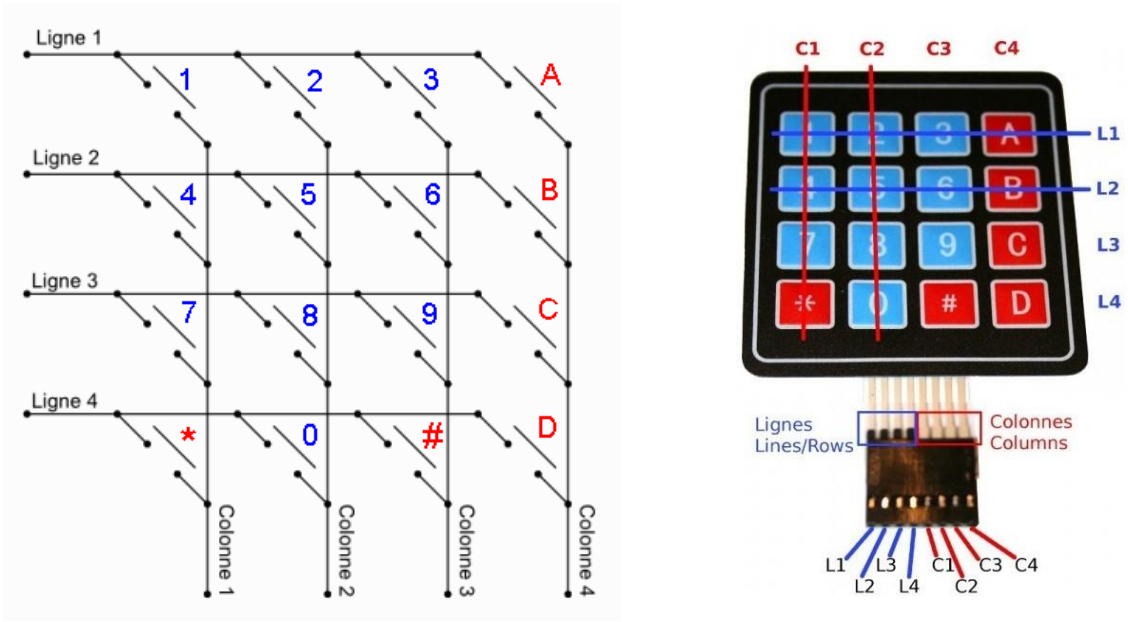


Figure 22 : Clavier matriciel 4x4

Les lignes sont des sorties. Les colonnes sont des entrées maintenues au niveau haut par une résistance interne dans Arduino. Le système envoie par balayage un niveau bas sur chaque ligne (1 seule à la fois) et balaye les colonnes en lecture. Quand il lit un niveau bas, c'est que la colonne est reliée par une touche appuyée à la ligne qui est basse à ce moment.

On n'a pas besoin de programmer ce balayage, c'est la bibliothèque qui le fait. La bibliothèque keypad.h permet de gérer facilement un tel clavier matriciel.

On le branche sur les 8 entrées successives de la carte Arduino Mega 2560 de A8 à A15 indiqué par la figure 23.

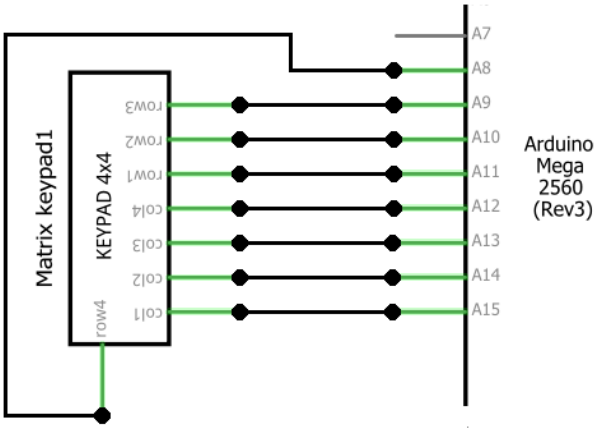


Figure 23 : Brochage du clavier matriciel 4x4 à l'Arduino Mega 2560

b) Utilisation

Afin que l'on puisse utiliser le clavier on fait appel à la librairie « Keypad.h », pour l'intégrer dans le fichier source, il suffit de cliquer sur le menu « Import Library » et d'aller chercher la bonne. Une ligne #include <Keypad.h> apparait en haut de la page du sketch.

On définit les constantes,

```
const byte ROWS = 4;
const byte COLS = 4;
char hexaKeys[ROWS][COLS] = {
  {'0','4','8','C'},
  {'1','5','9','D'},
  {'2','6','A','E'},
  {'3','7','B','F'}
};
byte rowPins[ROWS] = {A8, A9, A10, A11};
byte colPins[COLS] = {A12, A13, A14 A15};
```

Et on crée une instance à partir de l'objet ***Keypad*** appelée ***MyKeypad*** en écrivant :

```
Keypad MyKeypad = Keypad( makeKeymap(hexaKeys), rowPins, colPins, ROWS, COLS);
```

Pour la gestion du clavier on utilise les fonctions de la librairie Keypad.h.

II.1.1.2 L'afficheur graphique

C'est un module LCD (figure 24) qui dispose d'un écran de 128x64 pixels avec un rétro-éclairage. Ce module LCD est livré avec une puce à contrôleur ST7920 intégrée qui gère les données qui lui sont envoyées. L'affichage est entièrement programmable et peut afficher une combinaison de graphiques et de texte. Il peut fonctionner en mode parallèle et en mode série (SPI) qui peut être configuré par la broche PSB externe. En mode SPI, seules 3 broches de données sont requises pour conduire cet afficheur. Aucun potentiomètre n'est nécessaire pour régler le contraste, car il est prédéfini au niveau optimal.

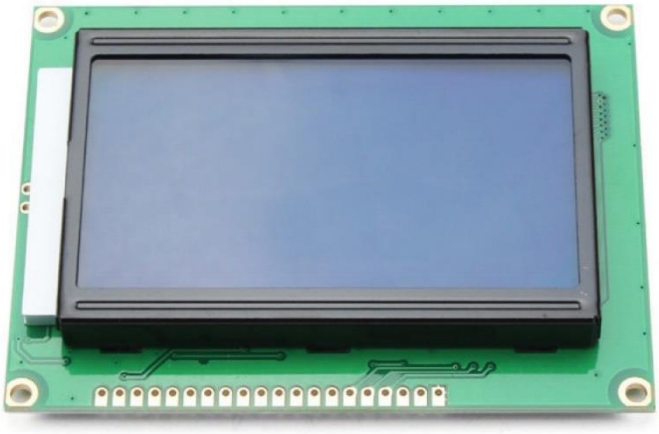


Figure 24 : Module LCD 128x64

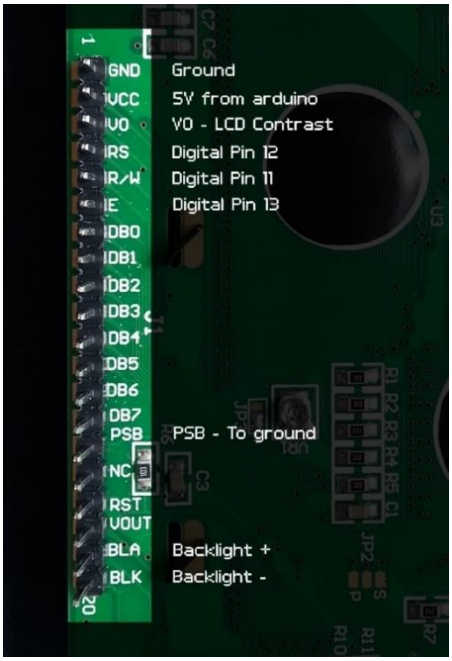


Figure 25 : Brochage du module LCD 128x64

L'écran dispose de 20 broches (figure 25), numérotées de haut en bas, comme on peut le voir dans l'image ci-dessus. Il peut être utilisé pour communiquer avec l'Arduino à la fois de manière série et parallèle. Dans notre cas, où le taux de rafraîchissement des écrans LCD n'est pas un point critique, on a utilisé la communication série SPI, occupant seulement 2 ports Arduino.

La bibliothèque qu'on a utilisée est l'U8Glib, une bibliothèque très complète et complexe.

a) Caractéristiques :

- Interface de contrôle : 8 ou 4 bits en mode parallèle (mode par défaut) ou 3 bits en mode sériel (SPI)
- Alimentation : 4.5 - 5.5V CC
- Contrôleur : ST7920
- Résolution : 128×64 pixels
- Couleur d'affichage : caractères noir sur fond jaune
- Angle de vision : 170°
- Type d'écran LCD : STN
- Contrôle du rétro-éclairage et du contraste
- Dimensions du module : 93 x 70 x 22 mm
- Dimensions de l'affichage : 73 x 39 mm

b) Câblage :

Commun :

GND -> la masse (0v)

VCC -> +5v

VO -> contraste (entre 0v et 5v)

RST -> reset (actif à LOW)

PSB -> choix mode (LOW = SPI, HIGH = parallèle)

En mode parallèle (comme un écran LCD classique) :

RS -> "Register select"

R/W -> "Read/Write control"

E -> "Enable"

Do ~ D7 -> bus de données

En mode SPI (figure 26):

RS -> "Chip Select" (CS)

R/W -> "Serial input" (MOSI) vers port 23 de l'Arduino Mega 2560

E -> "Serial clock" (SCLK) vers port 22 de l'Arduino Mega 2560

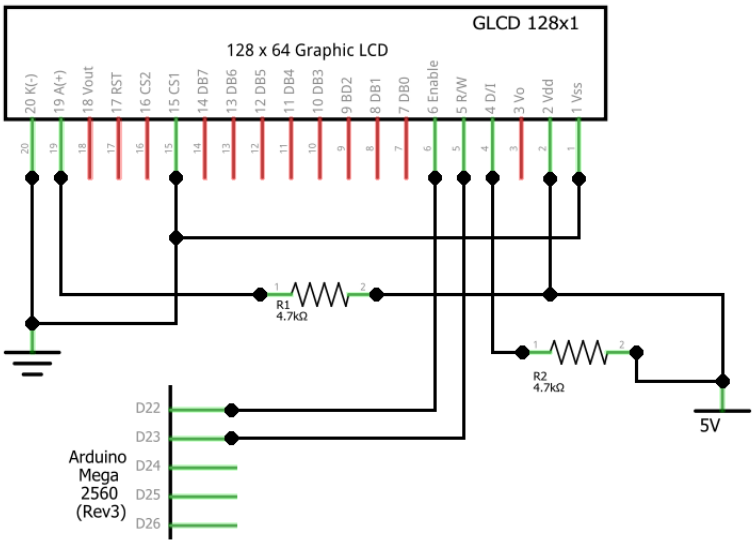


Figure 26 : Brochage du module LCD 128x64 a l'Arduino Mega 2560

c) Utilisation

Afin que l'on puisse utiliser, dans notre cas le module LCD 128x64 en mode SPI, on fait appel à la librairie « U8glib.h », pour l'intégrer dans le fichier source, il suffit de cliquer sur le menu « Import Library » et d'aller chercher la bonne. Une ligne #include < U8glib.h> apparait en haut de la page du sketch.

On crée une instance à partir de l'objet U8GLIB_ST7920_128X64_1X appelée Display en écrivant :

U8GLIB_ST7920_128X64_1X Display (22, 23, 24, U8G_PIN_NONE);

Pour la gestion de l'afficheur graphique on utilise les fonctions de la librairie U8glib.h. (https://github.com/olikraus/u8glib/wiki/userreference)

II.1.1.3 Horloge temps réel

Une horloge temps réel ("Real Time Clock" ou RTC) est un circuit intégré dont la principale fonction consiste à mesurer le temps (figure 27 et 28). Le module utilisé est basé sur le circuit intégré DS1307 de Dallas/Maxim, qui peut facilement être branché à un Arduino (protocole i2c) : chaque fois que l'Arduino a besoin de connaître la date ou l'heure, il n'a qu'à interroger le module RTC.

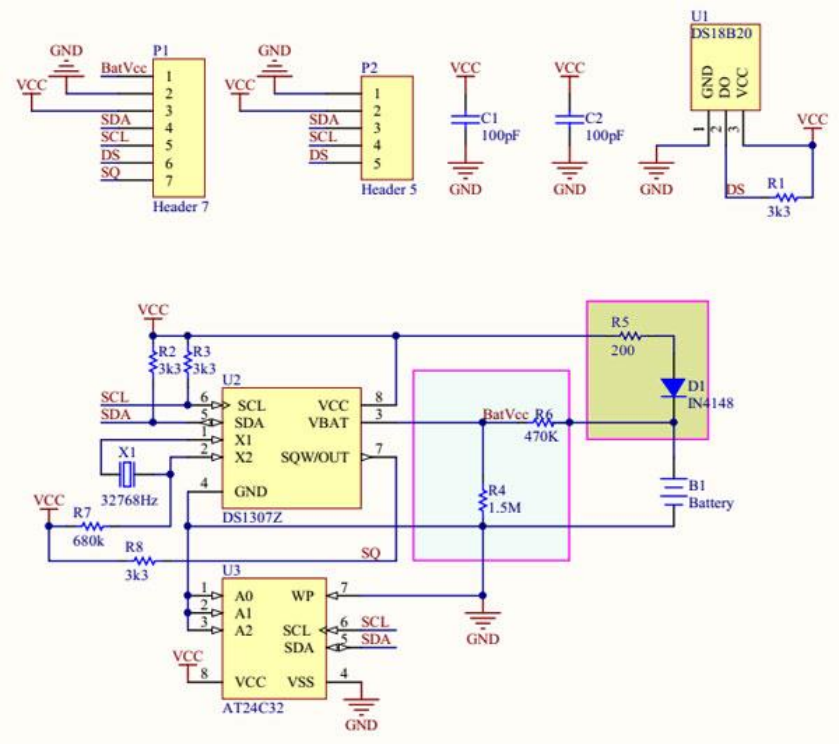


Figure 27 : Schéma électrique du RTC DS1307

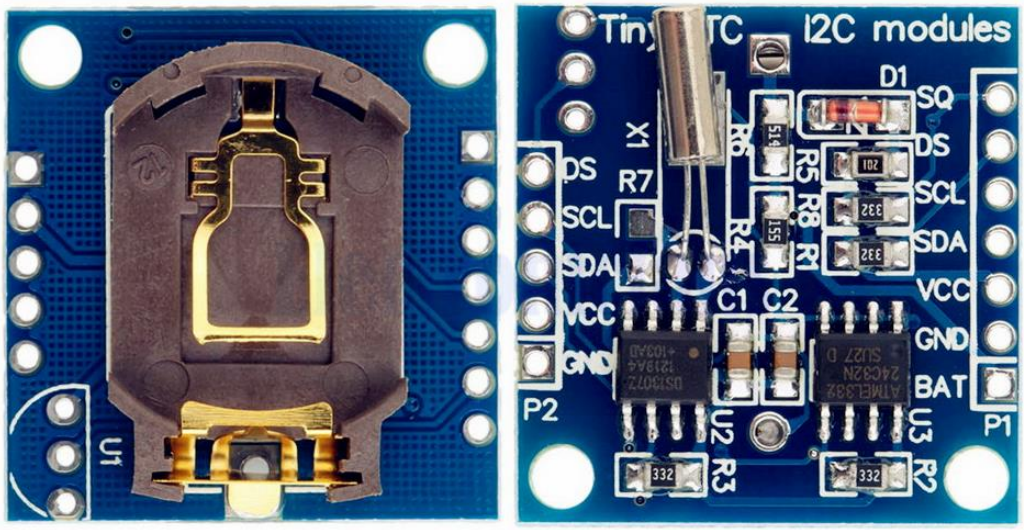


Figure 28 : RTC DS1307 Tiny

La puce RTC utilise le bus I²C pour communiquer, et cette connexion est réalisée au moyen de 2 lignes :

SDA (Serial Data Line) : ligne de données bidirectionnelle,

SCL (Serial Clock Line) : ligne d'horloge de synchronisation bidirectionnelle.

Ces 2 ports sont identifiés sur l'Arduino, et cela dépend du modèle :

Uno, Ethernet: A4 (SDA), A5 (SCL)

Mega2560: 20 (SDA), 21 (SCL)

Leonardo: 2 (SDA), 3 (SCL)

Due : 20 (SDA), 21 (SCL), SDA1, SCL1

a) Caractéristiques :

- Interface I2C deux fils
- Heures : Minutes : Secondes AM / PM
- Jour Mois, Date - Année
- Compensation de l'année
- Calendrier précis jusqu'en 2100
- La broche de sortie de 1Hz
- RTC à base de DS1307 avec la batterie LCR2032

b) Câblage (figure 29):

- VCC –> +5v
- GND –> GND
- SDA –> pin 20 Data
- SCL –> pin 21 Clock

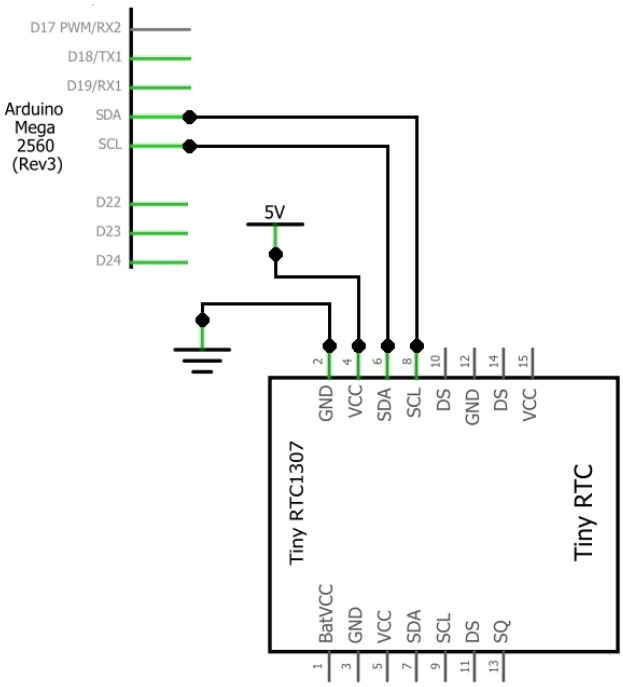


Figure 29 : Brochage du RTC DS1307 Tiny à l'Arduino Mega 2560

c) Utilisation

Le module RTC DS1307 Tiny servira pour la mesure des vitesses angulaires et linéaires, ainsi que la fréquence de réalisation des mesures des paramètres du sol et autres mesures liées au projet. Pour pouvoir utiliser le module RTC DS1307 Tiny, on utilise la librairie « RTClib.h». Pour l'intégrer c'est très simple, il suffit de cliquer sur le menu « Import Library » et d'aller la chercher. Une ligne #include < RTClib.h> apparait en haut de la page du sketch. On crée une instance à partir de l'objet RTC_DS1307 appelé RTC en écrivant : *RTC_DS1307 RTC;* et on démarre l'instance dans setup en écrivant : *RTC.begin();*

Pour la gestion du module RTC DS1307 Tiny on utilise les fonctions de la librairie RTClib. (Https://github.com/adafruit/RTClib)

II.1.1.4 Module Bluetooth HC-05

a) Qu'est-ce que le Bluetooth ?

Le Bluetooth, développé par la société Ericsson à la fin des années 1990 et qui n'a vraiment percé que dans les années 2000. Est un protocole de communication sans fil, qui a subi de nombreuses révisions et évolutions pour atteindre aujourd'hui la version 4.1 depuis la fin 2013.

Ce protocole est un cousin du Wi-Fi. En effet, ils respectent tous deux une même spécification IEEE et utilisent la même gamme de fréquences : 2.4 GHz (tout comme les téléphones portables et le zigbee par exemple). C'est une communication bidirectionnelle, deux modules peuvent communiquer ensemble en même temps. Le comportement utilisé est « maître/esclave ». Un esclave pourra parler avec un seul maître, mais un maître pourra dialoguer avec plusieurs esclaves. Pour son utilisation, elle se passe en plusieurs étapes :

1. Le maître se met en mode « reconnaissable »
2. L'esclave trouve le maître et demande à s'y connecter
3. Le maître accepte la connexion
4. Les périphériques sont alors appairés (ou associés)
5. La communication peut commencer

b) Présentation

Il existe deux sortes de module Bluetooth, tous deux compatibles Arduino. On les distingue par le nombre de pattes d'entrée /sorties :

- HC-05 : 6 pattes d'entrées /sorties. Ce module peut être maître ou esclave. On mode maître, il peut proposer à un autre élément Bluetooth de s'appairer avec lui, par contre on mode esclave, il ne peut que recevoir des demandes d'appairage.
- HC-06 : 4 pattes d'entrées / sorties. Ce module ne peut être qu'esclave, il ne peut recevoir que des demandes d'appairage.

Les deux modules peuvent être utilisés en mode COMMANDE, pour les programmer avec des « commandes AT », ou en mode DATA, pour échanger des données.

Le module Bluetooth HC-06 ou HC-05 (figure 30) permet d'établir une liaison Bluetooth (liaison série sans fil) entre une carte Arduino et un autre équipement possédant une connexion Bluetooth (Smartphone, tablette, seconde carte Arduino, etc....).

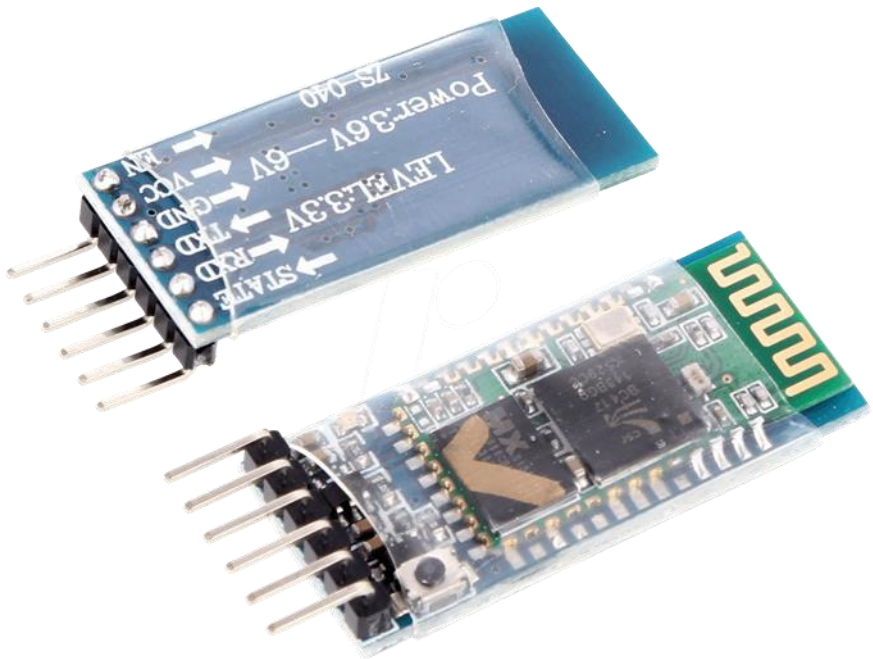


Figure 30 : Module Bluetooth HC-05

c) Caractéristiques

- Module de type JY-MCU / HC-05 maître ou esclave
- Paramétrage par défaut du port série : 9600, N, 8, 1.
- Le baudrate est paramétrable de 4800 à 1382400 (par commande AT, uniquement si le module n'est pas associé, et maxi 115200 pour pouvoir l'utiliser avec une carte Arduino).
- Alimentation de 3.3 à 5V DC
- LED indicatrice : statut de connexion.
- Fonctionnement Bluetooth sur la bande 2.4 GHz, modulation GFSK.
- Le module est apparié avec un mot de passe (1234) modifiable.
- L'identificateur du module est généralement HC-05 modifiable.

d) Détails techniques du module

- PIO8 est branchée sur une cathode de LED avec une résistance de 470 Ohm en série. Le moins de la LED se branche à la masse. Utilisé pour indiquer l'état de fonctionnement du module selon le type de clignotement de la LED après mise sous tension.
- PIO9 est utilisé pour contrôler la LED de contrôle qui indique l'appariement. Elle reste éclairée en continu quand l'appariement est réalisé.
- PIO11, pin d'état du module. Si HIGH réponse à une commande AT ; si LOW ou flottant statut de fonctionnement régulier.
- Avec un circuit de reset intégré, le reste est automatiquement activé à la mise sous tension.

e) Câblage à réaliser (figure 31) :

Module HC-05 (4 pins) vers Arduino

- VCC –> +5V
- GND –>GND
- TXD –> RX
- RXD –> TX

f) Important

- Il faut croiser Rx et Tx entre le module et l'Arduino.
- La tension d'alimentation de ces modules doit être comprise entre 3,3 et 6 V, mais la broche RX ne peut recevoir qu'une tension maximale de 3,3 V. Il faudra prévoir un pont diviseur de tension pour ramener la tension 5 V délivrée par la carte Arduino pour ne pas endommager la broche RX du module Bluetooth.

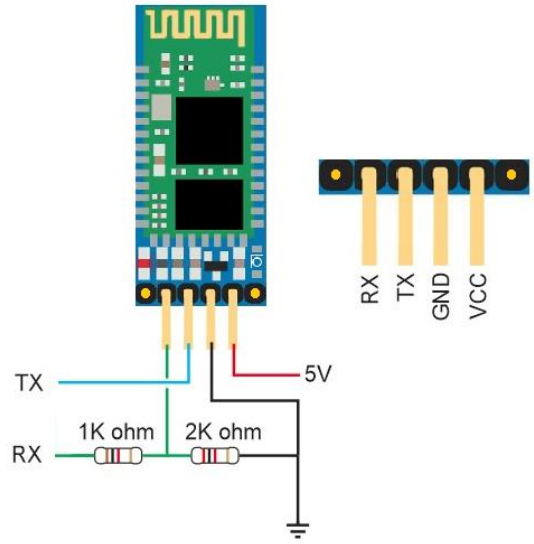


Figure 31 : Brochage du module HC-06 a l'Arduino Mega 2560

g) Utilisation

Le module Bluetooth HC-06 est utilisé pour l'échange des informations entre la carte Arduino Mega et l'application Android SoilTractorPrediction. Pour pouvoir utiliser le module HC-06, on utilise la librairie « SoftwareSerial.h» par l'intégration de la ligne #include < SoftwareSerial.h> dans le sketch et on crée une instance à partir de l'objet SoftwareSerial appelé BltHC06 en écrivant : SoftwareSerial BltHC06(19, 18); puis on démarre l'instance dans setup en écrivant : BltHC06.begin(9600);

Pour la gestion du module HC-06 on utilise les fonctions de la librairie SoftwareSerial. (Https://github.com/PaulStoffregen/SoftwareSerial)

II.1.1.5 Le module nRF24L01

Le module radio nRF24L01 (figure 32, 33 et 34) est un module radio tout intégré du fabricant Nordic Semiconductor. Il s'agit d'un module radio intégrant tout le nécessaire pour émettre et recevoir des données sur la même gamme de fréquences que le WiFi et le Bluetooth, mais n'est pas compatible avec ceux deux protocoles, il utilise le protocole de communication propriétaire de Nordic nommée "ShockBurst" intégrée dans le hardware du module et n'est pas désactivable. Ce protocole de communication permet au nRF24L01 d'être considéré comme un modem complet, avec adressage, gestion des erreurs de transmission et retransmission automatique en cas de non-réponse du destinataire.

Il existe deux versions du module nRF24L01 : la version classique et la version "+". Le nRF24L01+ est la nouvelle révision du chipset radio. Cette nouvelle version apporte énormément d'amélioration au module radio. Les deux versions sont compatibles d'un point de vue logiciel, mais d'un point de vue physique, la version "+" est bien plus intéressante. La version "+" a l'avantage d'avoir plus de mémoire, plusieurs canaux de communication simultanés et une fonction de renvoi automatique en cas de non-réponse du destinataire. La version "+" a aussi une plus grande portée, une meilleure sensibilité en réception et un débit plus important en transmission.

Le débit maximum du nRF24L01+ est de 2Mbps, mais à cette vitesse la portée n'est que de quelques mètres. La portée dépend beaucoup de l'antenne. Une antenne "trace" (directement gravé sur la carte) n'a pas une grande portée, une antenne céramique a plus de portée, le mieux est bien évidemment d'utiliser une "vraie" antenne avec un connecteur SMA (pas de vis plaqué or). Dans notre cas, la portée souhaitée est de deux mètres à trois mètres au maximum, une antenne "trace" suffit donc largement.

Le nRF24L01+ dispose de 32 octets de mémoire en émission et en réception par canaux de communication, pour un total de 6 canaux de communication simultanés. La taille maximum d'un paquet est de 32 octets, ce qui est assez faible, mais largement suffisant pour transmettre des mesures, ou des commandes donc pas de problèmes pour notre cas.

Figure 32 : Module nRF24L01

a) Caractéristiques

- Fréquence : 2.4 GHz ISM (Industrial, Scientific and Medical) band
- Tension d'alimentation : 1.9 à 3.6 V
- Interface SPI jusqu'à 10 Mb/s (tolérant 5 V)
- Vitesses de transmission : 250 kb/s, 1 Mb/s et 2 Mb/s
- Très basse consommation (plusieurs mois, voire années avec une pile bouton ou des piles AA/AAA). 900 nA deep sleep mode. 13.3 mA Radio RX at 2 Mb/s on-air data-rate.
- Portée : quelques mètres avec une antenne PCB (modèle présenté ici) et jusqu'à un kilomètre avec une antenne externe.
- Protocole propriétaire *Enhanced ShockBurst™* qui permet la communication bidirectionnelle avec mise en mémoire tampon des paquets de données, confirmation des paquets reçus et retransmission automatique des paquets perdus.

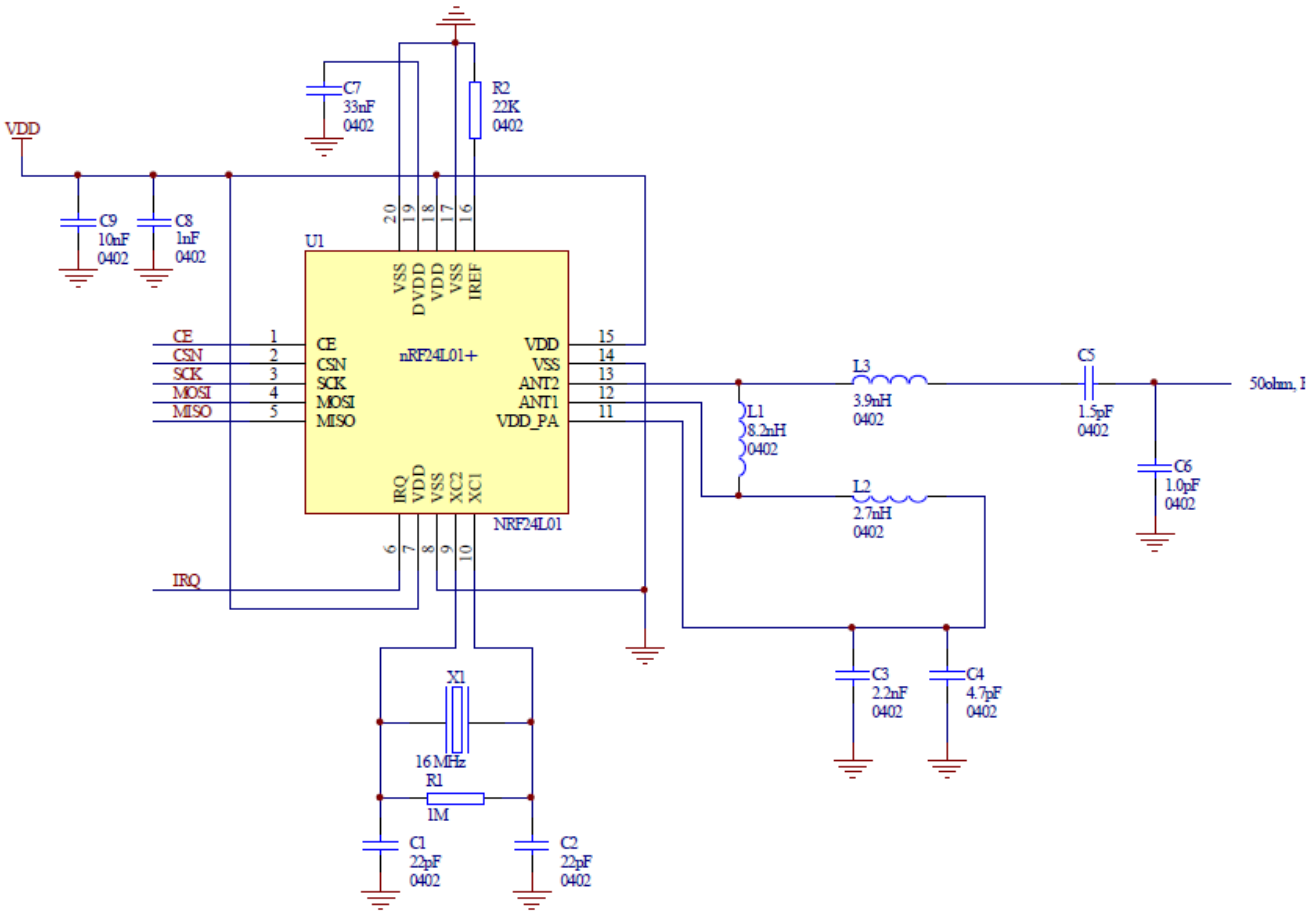


Figure 33 : Schéma électrique du nRF24L01 avec sortie RF de 50Ω

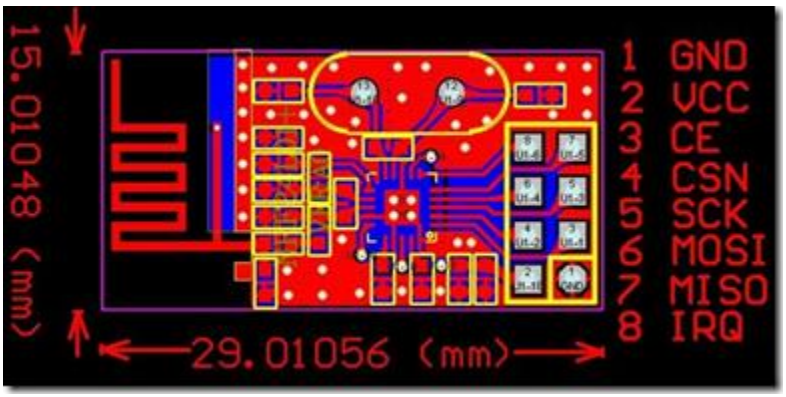


Figure 34 : Module nRF24L01+ avec ses broches

Le tableau 5 décret la fonction de chaque broche du module NRF24L01 en mode SPI.

Tableau 5: Fonctions des connexions du NRF24L01

Signal	Direction	Description
		Chip Enable
CE	input	Ce signal est actif à 1 et sert à configurer le module en mode de réception (RX) ou de transmission (TX)
CSN	input	*SPI Chip Select*
SCK	input	*SPI Clock*
MOSI	input	*SPI Slave Data Input*
MISO	output	*SPI Slave Data Output, with tri-state option*
		Maskable interrupt pin
IRQ	output	Ce signal est actif à 0 et est contrôlé par trois sources d'interruption masquables

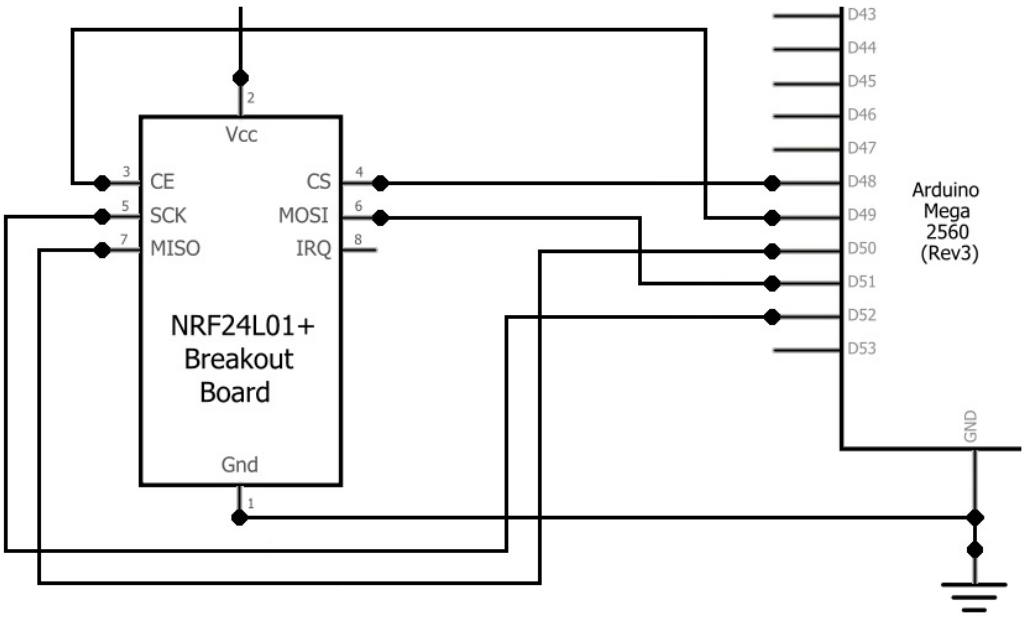


Figure 35 : Brochage du module nRF24L01+ a l'Arduino Mega 2560

Le tableau 6 décret le brochage du module NRF24L01 avec la carte Arduino Mega.

Tableau 6: Connexion NRF24L01 avec Arduino Mega

Broche	NRF24L01+	Arduino Mega
1	GND	GND
2	VCC	3.3V
3	CE	D49
4	CSN	D48
5	SCK	D52
6	MOSI	D51
7	MISO	D50
8	IRQ	-

b) Utilisation

Pour pouvoir utiliser le module NRF24L01, les librairies suivantes « SPI.h, Mirf.h, nRF24L01.h, et MirfHardwareSpiDriver.h » doivent être ajoutées au sketch par l'ajout des lignes suivantes.

```
#include <SPI.h>                    // Pour la communication via le port SPI
#include <Mirf.h>                   // Pour la gestion de la communication
#include <nRF24L01.h>               // Pour les définitions des registres du nRF24L01
#include <MirfHardwareSpiDriver.h>  // Pour la communication SPI
```

Et dans la fonction d'initialisation on initialise par.

```
Mirf.cePin = 49;                // Broche CE sur D49
Mirf.csnPin = 48;               // Broche CSN sur D48
Mirf.spi = &MirfHardwareSpi;    // On veut utiliser le port SPI hardware
Mirf.init();                    // Initialise la bibliothèque
Mirf.channel = 1;               // Choix du canal de communication 128 canaux
Mirf.payload = 32;              // Taille d'un message (maximum 32 octets)
Mirf.config();                  // Sauvegarde la configuration dans le module radio
```

```
Mirf.setTADDR((byte *) "nrf02"); // Adresse de transmission
Mirf.setRADDR((byte *) "nrf01"); // Adresse de réception
```

Pour la gestion du module NRF24L01 on utilise les fonctions de la bibliothèque Mirf disponible sur GitHub.

II.1.1.6 Module HX711

a) Détails techniques du module

Ce module utilise le HX711 (figure 36, 37 et 38) qui est un convertisseur analogique-numérique (ADC) d'une précision de 24 bits. Il est spécialement conçu pour les applications de mesure électroniques de force de haute précision et de contrôle industriel à interfacer directement avec un capteur à base de pont de Wheatstone. Il possède deux canaux d'entrée analogique avec l'intégration interne d'un amplificateur à gain programmable de 32,64 à 128 fois. L'interface d'entrée est compatible avec le port Arduino d'E/S.

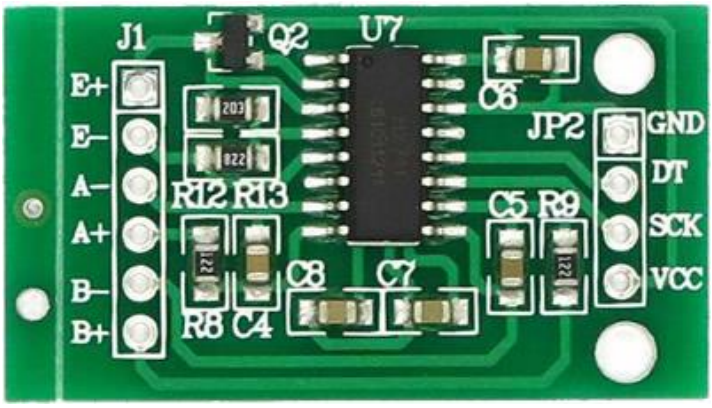


Figure 36 : Le module HX711

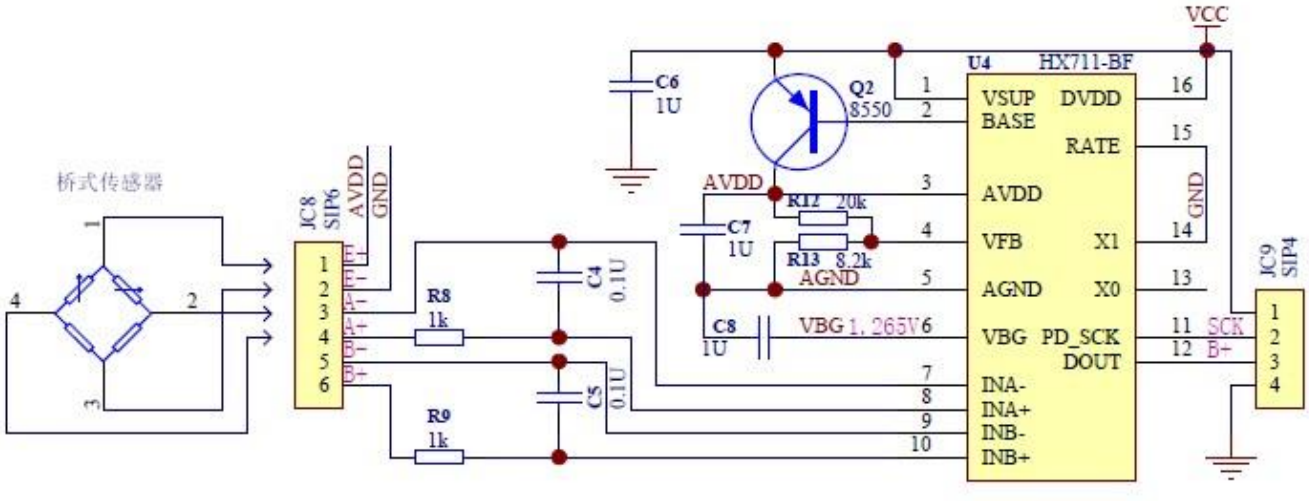


Figure 37 : Schémas électrique du module HX711

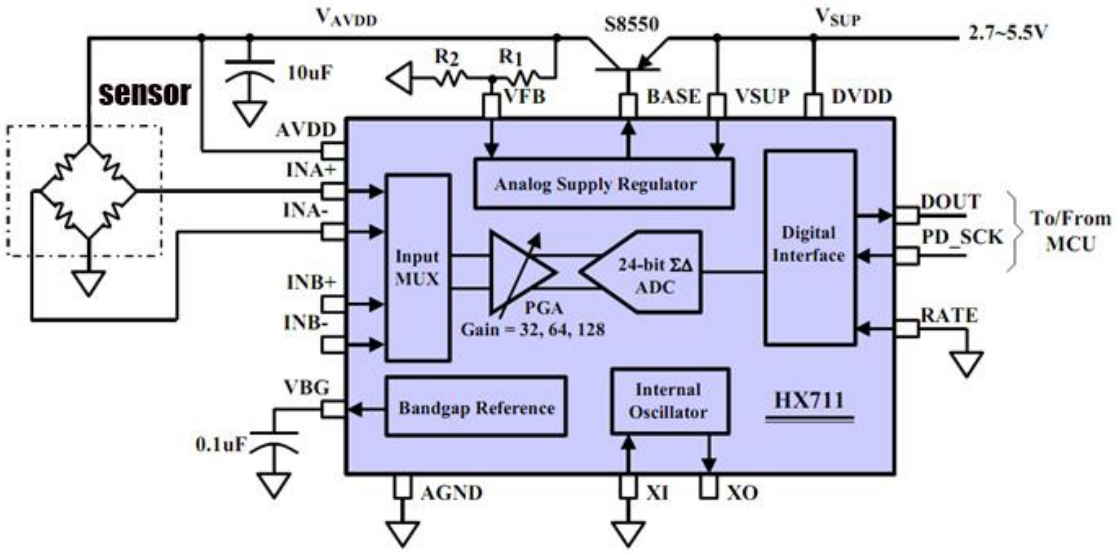


Figure 38 : Schémas synoptique du HX711

b) Caractéristiques

- Tension d'alimentation : 2.6V ~ 5.5V
- Courant : <1,6mA
- Température de fonctionnement : -40 ~ 85 ° C
- Gain réglable de 32,64 à 128
- Fréquence : 50 et 60hz
- Sensibilité en sortie : 1.0±0,15 mV/V
- Capacité de surcharge : 200 % F.S

c) Jauge de contrainte

Il s'agit d'un capteur de force à demi-pont (cellule de charge) de mesure allant jusqu'à 50 kg (pour des essais au laboratoire). Il peut être utilisé pour mesurer le poids, la force linéaire ou même la poussée.

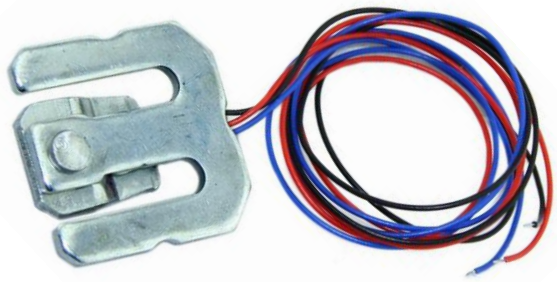


Figure 39 : Jauge de contrainte

Spécification pour jauge de contrainte:

- Charge maximale: 50kg
- Tension de sortie: 1.0 ± 0.1mV / V
- Non linéarité: 0.08% F.S
- Hystérésis: 0,1% F.S
- Répétabilité: 0,05% F.S
- Effet de la température en sortie: 0.02% F.S / ° C
- Effet de température sur zéro: 0,02% F.S / ° C
- Zéro équilibre: ± 0,1000 mV / V
- Impédance de sortie: 1000 ± 10% Ω
- Plage de température de fonctionnement: -20 ~ 65 ° C
- Tension d'excitation recommandée: 5 VDC
- Tension de fonctionnement maximale: 8VDC
- Niveau de protection: IP65
- Matériau: alliage d'aluminium
- Câble: ϕ0.8 × 460 mm

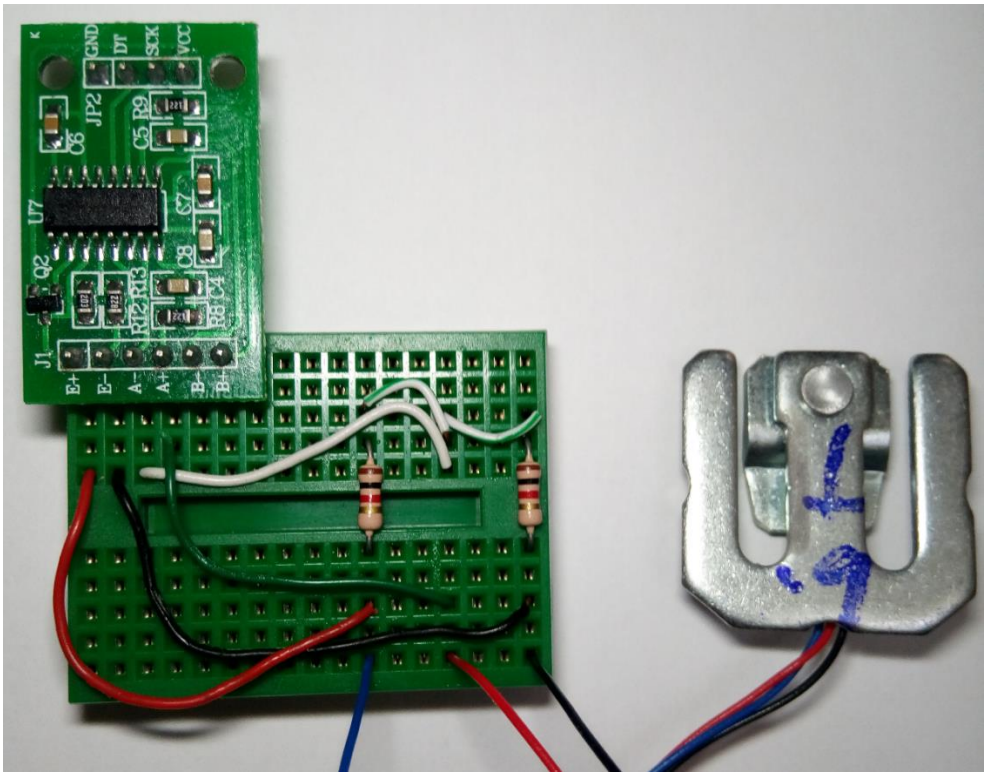


Figure 40 : Connexions HX711 et Jauge de contrainte

d) Câblage à réaliser

Le câblage total à réaliser est indiqué dans le tableau 7 et schématiser par les figures 40 et 41.

Tableau 7: Connexions Arduino Mega, HX711 et Jauge de contrainte

Arduino mega	HX711		Jauge de contrainte
5V	VCC	E+	ALIM + (BLEU)
GND	GND	E-	ALIM - (NOIR)
D30	SCK	A-	SIGNAL - (JAUNE)
D31	DT	A+	SIGNAL + (ROUGE)

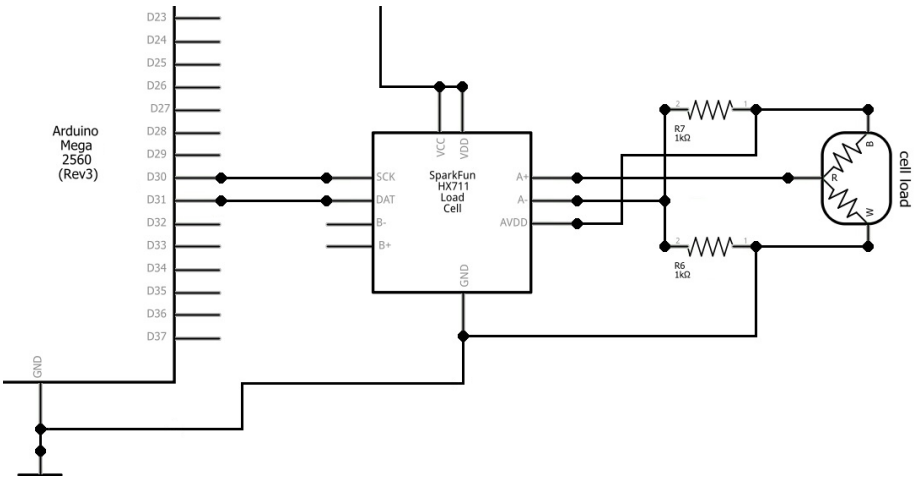


Figure 41 : Connexions Arduino Mega, HX711 et Jauge de contrainte

e) Important

Pour l'expérimentation, on connecte une cellule de charge de 50 kg au module d'amplificateur de cellule de charge HX711. Le HX711 est un convertisseur analogique-numérique analogique (ADC) de précision à 24 bits conçu pour la mesure de force et les applications de contrôle industriel pour s'interfacer directement avec un capteur de pont. Le multiplexeur d'entrée sélectionne l'entrée différentielle du canal A ou B dans l'amplificateur de gain programmable à faible bruit (PGA). Le canal A peut être programmé avec un gain de 128 ou 64, correspondant à une tension d'entrée

différentielle à pleine échelle de ± 20mV ou ± 40mV respectivement, lorsqu'une alimentation de 5V est connectée à la broche d'alimentation analogique AVDD. Le canal B a un gain fixe de 32. Le régulateur d'alimentation de la puce élimine le besoin d'un régulateur d'alimentation externe pour fournir une alimentation stable au CAN et au capteur. Les circuits de réinitialisation à la mise sous tension de la puce simplifient l'initialisation de l'interface numérique. Il n'y a pas de programmation nécessaire pour les registres internes.

f) Utilisation

Pour pouvoir utiliser le module HX711, la librairie suivante « hx711.h » doit être ajoutée au sketch avec la définition des entrées par l'ajout des lignes suivantes.

```
#include "HX711.h"
#define DOUT  3
#define CLK  2
HX711 scale(DOUT, CLK);
float calibration = -96650;
```

Et dans la fonction d'initialisation on initialise par.

```
scale.set_scale();
scale.tare();
long zero_factor = scale.read_average();
```

II.1.1.7 Le module GY 521

Le module GY 521 (figure 42) est équipé d'un MPU-6050 une puce MEMS (systèmes micro électromécaniques) très précise qui contient un accéléromètre 3 axes, un gyroscope 3 axes, un processeur de mouvement numérique DMP (Digital Motion Processor) et un capteur de température. Le processeur de mouvement numérique peut être utilisé pour traiter des algorithmes complexes directement sur la carte. Les valeurs des capteurs après une conversion analogique-numérique sur 16 bits simultanée sur chaque canal sont récupérées en utilisant le bus de données série I2C, qui ne nécessite que deux fils (SCL et SDA). La lecture des mesures brutes de ce capteur est facile. Le capteur contient un registre FIFO de 1024 octets que le microcontrôleur Arduino peut lire, étant prévenu par un signal d'interruption.

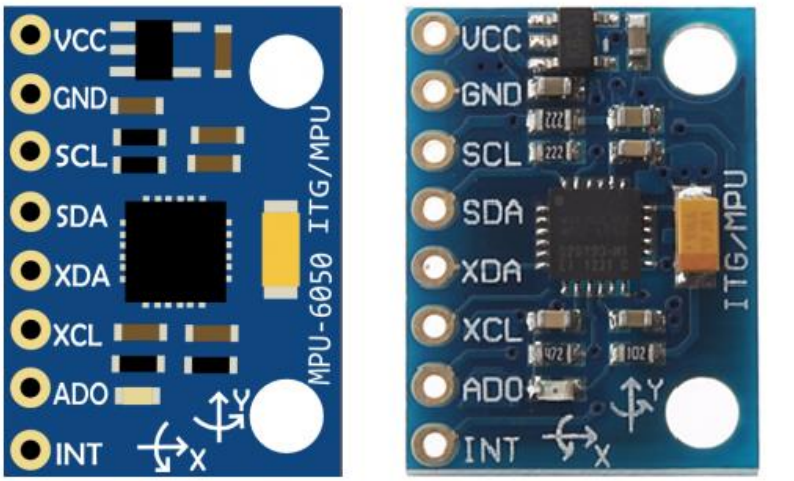


Figure 42 : Le module GY 521

a) Caractéristiques

Caractéristiques du gyroscope

- Capteurs de vitesse angulaire (gyroscopes) à 3 axes X, Y et Z à sortie numérique avec une plage en pleine échelle programmable par l'utilisateur de ± 250, ± 500, ± 1000 et ± 2000 ° / s
- Le signal de synchronisation externe connecté à la broche FSYNC prend en charge la synchronisation de l'image, de la vidéo et du GPS
- ADC 16 bits intégrés permettant l'échantillonnage simultané des axes X, Y et Z
- Le biais amélioré et la stabilité de température de sensibilité réduit le besoin d'étalonnage d'utilisateur
- Amélioration des performances de bruit à basse fréquence
- Filtre passe-bas programmable numériquement
- Courant de fonctionnement du gyroscope : 3.6mA
- Courant de veille : 5µA
- Facteur d'échelle de sensibilité calibré en usine
- Autotest utilisateur

Caractéristiques de l'accéléromètre

- Accéléromètre à 3 axes à sortie numérique avec une gamme d'échelle programmable de ± 2g, ± 4g, ± 8g et ± 16g

- Les CAN intégrés 16 bits permettent l'échantillonnage simultané d'accéléromètres sans nécessiter de multiplexeur externe
- Accéléromètre courant de fonctionnement normal: 500µA
- Courant de mode d'accéléromètre de basse puissance: 10µA à 1.25Hz, 20µA à 5Hz, 60µA à 20Hz, 110µA à 40Hz
- Détection d'orientation et de signalisation
- Détection de robinet
- Interruptions programmables par l'utilisateur
- Interruption G élevée
- Autotest utilisateur

b) Broches de connexion

Le module GY 521 possède huit broches:

- VCC (Le module est équipé d'un régulateur de tension, ce qui nous permet de le connecter à des sources de 3,3 V et 5 V).
- GND
- SCL (Ligne d'horloge série du protocole I2C.)
- SDA (Ligne de données série du protocole I2C.)
- XDA (données auxiliaires => données série I2C maître pour la connexion du module à des capteurs externes.)
- XCL (Horloge auxiliaire => Horloge série I2C maître pour la connexion du module aux capteurs externes.)
- AD0 (Si cette broche est BASSE, l'adresse I2C de la carte sera 0x68, sinon, si la broche est HAUTE, l'adresse sera 0x69.)
- INT (Sortie numérique d'interruption)

c) Connexion à la carte annexe

Le module GY 521 est connecté à l'Arduino Mega (figure 43) par les ports logiques 02, 20, 21 et les ports d'alimentation 5V et 0V. Les ports logiques 20, 21 seront configurés dans le programme pour qu'il fonctionne en mode I2C, ainsi 21 sera un SCL

et 20 sera un SDA, le port 02 sert pour changer l'adresse du module pour éliminer le conflit d'adresse avec l'horloge RTC.

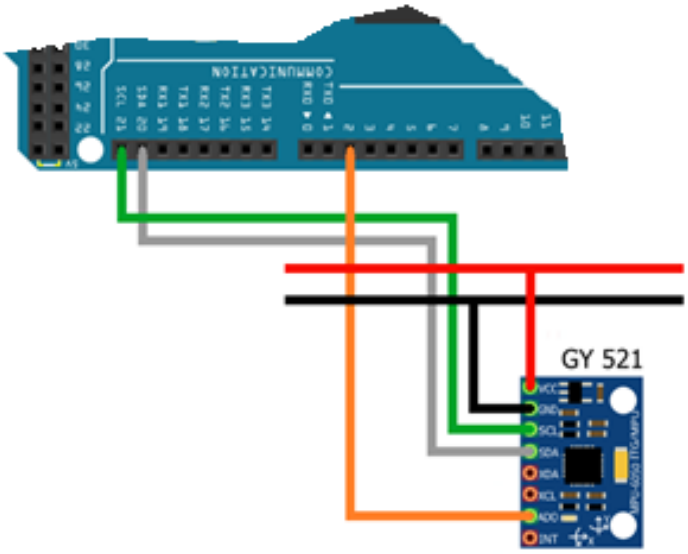


Figure 43 : Connexion du module GY 521 a l'Arduino Mega

d) Utilisation dans le système

Le GY521 est utilisé pour déterminer l'angle de pente sur laquelle se déplace le tracteur. Pour pouvoir utiliser le module GY521, on fait appel à la librairie « Wire.h », et on définit les constantes et variables suivantes, en écrivant :

```
const int MPU_addr=0x68;          // définition de l'adresse I2C du GY521
int16_t AcX,AcY,AcZ;              // Variable de stockage pour accéléromètre
int16_t Tmp;                      // Variable de stockage pour température
int16_t GyX,GyY,GyZ;              // Variable de stockage pour gyroscope
int minVal=265;                   //
int maxVal=402;                   //
double x;                          // Variable de stockage de l'angle/l'abscisse
double y;                          // Variable de stockage de l'angle/l'ordonnée
double z;                          // Variable de stockage de l'angle/la cote
```

Et dans la fonction d'initialisation on initialise par.

```
Wire.begin();                      // démarrage de I2C
Wire.beginTransmission(MPU_addr);  // transmettre à l'adresse du GY521
Wire.write(0x6B);
Wire.write(0);                     // mettre à zéro (réveille le MPU-6050)
Wire.endTransmission(true);
```

II.2 Les unites annexes

Les unités annexes (figures 44, 45) sont à base de l'Arduino nano (figures 46, 47), une carte basée sur le microcontrôleur Atmel ATmega328. L'Arduino Nano est essentiellement un Arduino Uno conçu pour une utilisation directe sur breadboard. Il embarque son propre contrôleur USB permettant une communication série très facile.

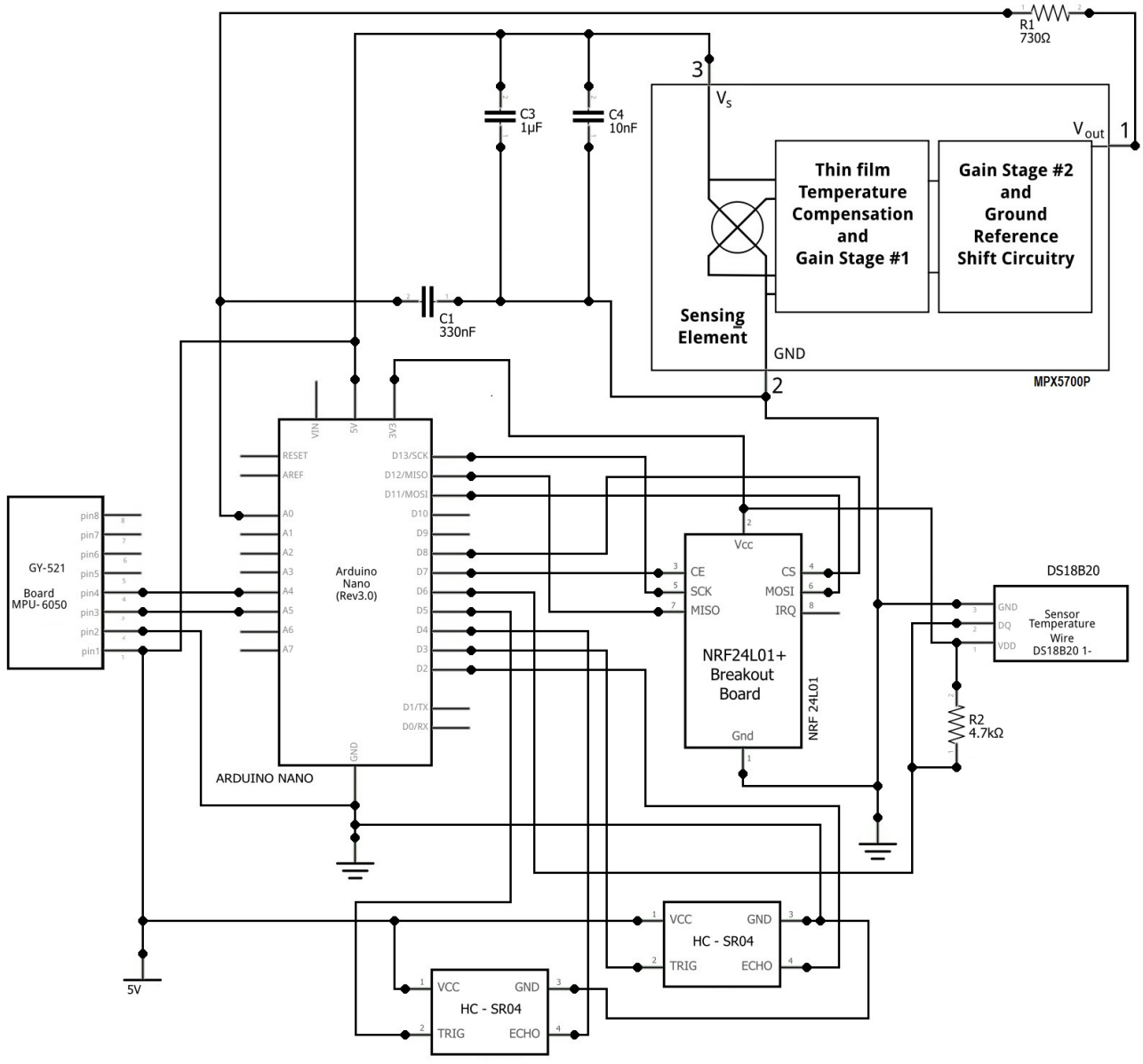


Figure 44 : Schéma électrique d'une unité annexe

Les modules connectés à l'unité annexe sont ; un module NRF25L01 RF pour communiquer avec l'unité principale, un gy521 module gyroscopique pour calculer le nombre de tours de la roue motrice, un MPX 5500 capteur de pression pour mesurer la pression à l'intérieur du pneu et, enfin, deux capteurs à ultrasons pour mesurer à la fois

le rayon statique et le rayon dynamique des roues motrices. L'unité annexe est placée à l'intérieur de la roue (figure 18) elle contient un firmware qui gère les différents modules de mesure, et la communication avec l'unité principale. L'unité annexe est alimentée par une pile de 9V directement après une régulation de la tension à 5V.

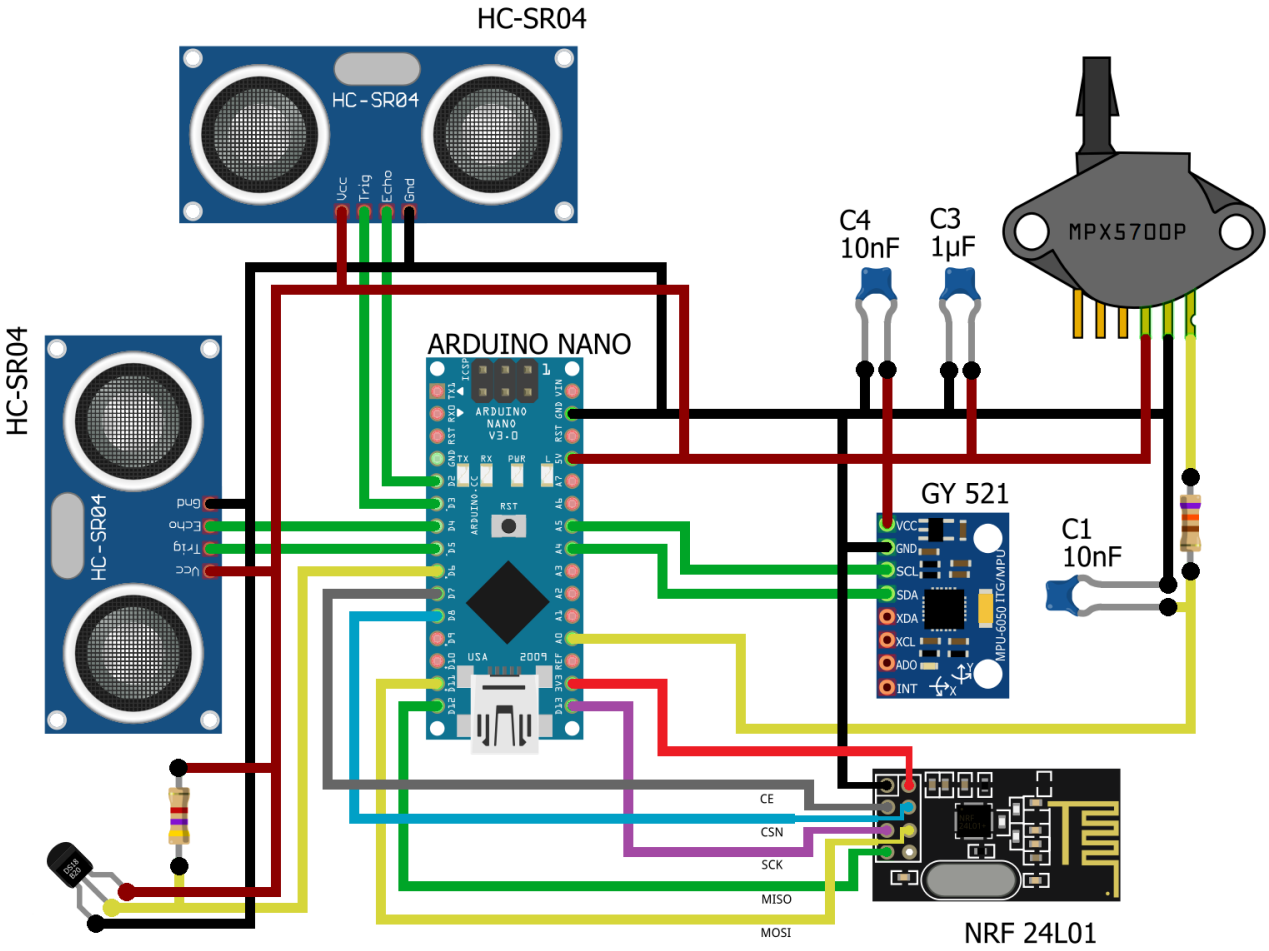


Figure 45 : Schéma de connexion d'une unité annexe

II.2.1 La carte Arduino Nano

La carte Arduino Nano propose plus ou moins les mêmes fonctionnalités que la carte Arduino Uno mais dans un design plus compact. Par rapport à la carte Arduino Uno, la carte Arduino Nano ne propose pas de prise jack pour l'alimentation et la prise USB est une prise mini-USB plutôt qu'une prise USB standard. La carte Arduino Nano 3.0 est basée sur un ATMega328 cadencé à 16 MHz. Sa mémoire de 32 kB et son grand nombre d'E/S font de ce circuit compatible DIL30 un élément idéal pour les systèmes embarqués ou pour des applications robotiques nécessitant du multitâches.

La Nano 3.0 peut se programmer avec le logiciel Arduino. Le contrôleur ATMega328 contient un bootloader qui permet de modifier le programme sans passer par un programmateur.

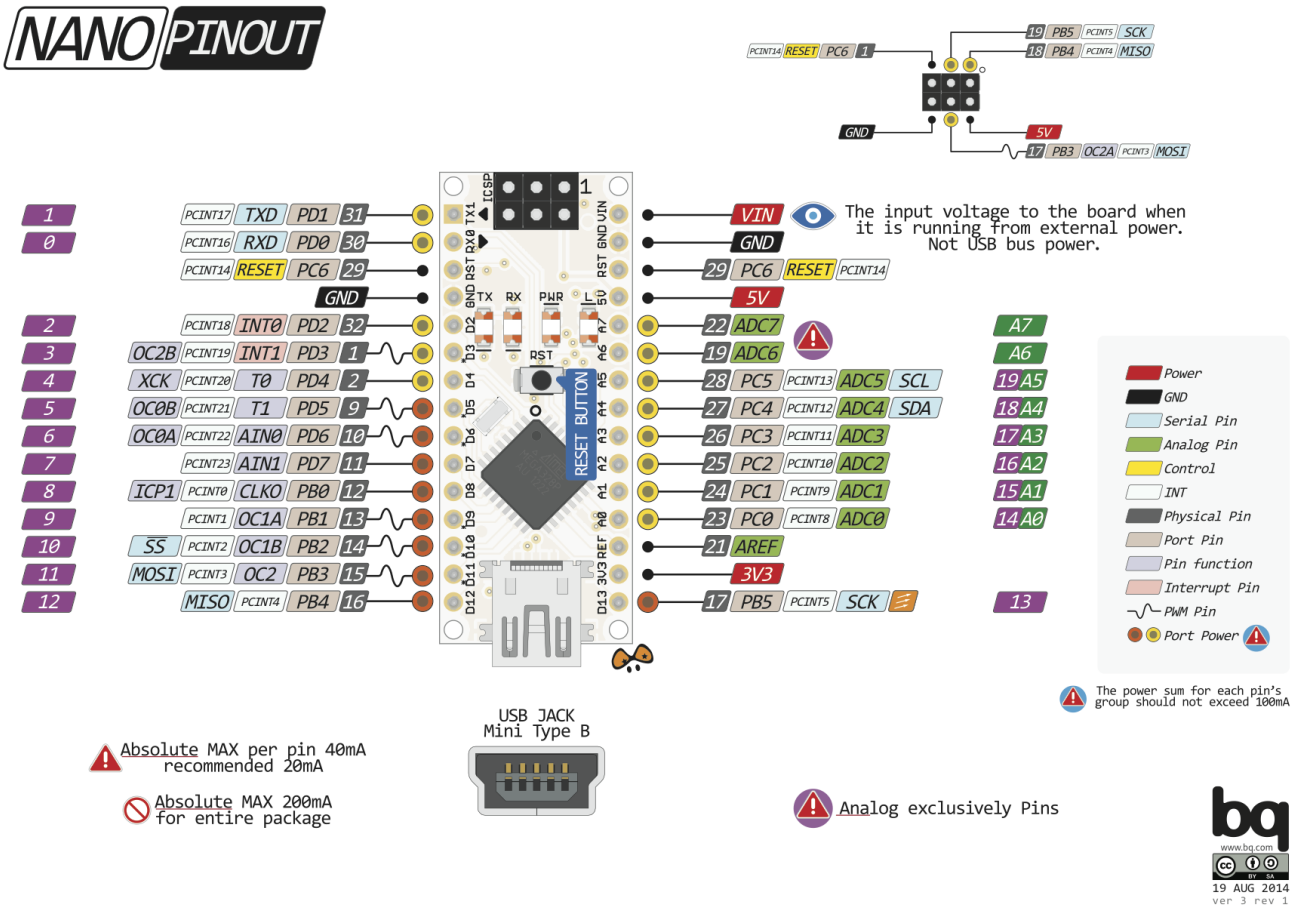


Figure 46 : Les connexions de la carte Arduino Nano

a) Spécifications techniques de la carte Arduino Nano 3.0

Les spécifications techniques de cette carte Arduino Nano sont les suivantes:

- Microcontrôleur Atmel ATmega328
- Voltage opérationnel (au niveau logique) : 5 V
- Voltage d'entrée recommandé : de 7 à 12 V
- Limite de voltage d'entrée : de 6 à 20 V
- Pins d'entrées/sorties digitales : 14 (dont 6 proposent une sortie PWM)
- Pins d'entrée analogique : 8
- Courant direct par pin d'entrée/sortie : 40 mA
- Mémoire Flash : 32 KO (ATmega328) dont 2 KO sont utilisés par le bootloader
- SRAM : 2 KO (ATmega328)

- EEPROM : 1 KO (ATmega328)
- Vitesse d'horloge : 16 MHz
- Dimensions 0.73" x 1.70"

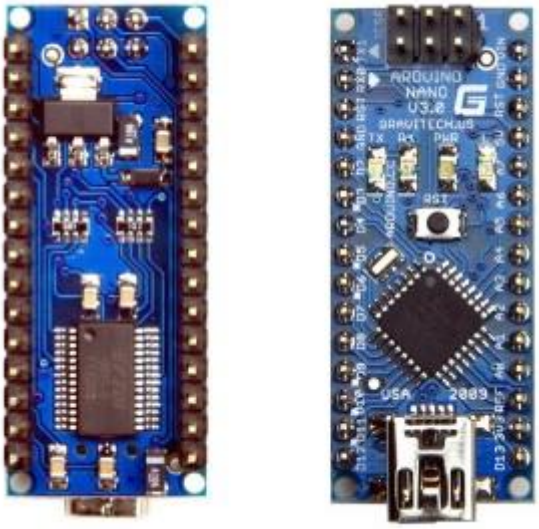


Figure 47 : La carte Arduino Nano 3.0

b) Alimentation de la carte Arduino Nano

La carte Arduino Nano peut être alimentée via le connecteur USB mini-B, de 7 à 12V via la pin 30 (Vin) ou via une alimentation 5V régulée sur la pin 27 (5v). Le Nano sélectionne automatiquement son alimentation sur la source au voltage le plus élevé. La puce FTDI FT232RL n'est alimentée que si la carte Arduino Nano est alimentée par le connecteur USB. La sortie 3.3v ne sera donc disponible que dans ce mode d'alimentation.

c) Entrées – Sorties

Les E/S digitales du Nano peuvent être utilisées en entrée ou en sortie avec un niveau logique de 5v. Chaque pin peut fournir 40mA au maximum et comporte une résistance de pull-up interne (désactivée par défaut) de 20-50 KOhms. Certains pins peuvent avoir des fonctions spéciales :

- Série : 0 (Rx) et 1 (Tx)
- Interruptions externes : 2 et 3. Ces pins peuvent être configurés pour déclencher une interruption sur n'importe quel changement d'état
- PWM : 3, 5, 6, 9, 10 et 11. Ces pins produisent une sortie PWM sur 8-bit

- SPI : 10 (SS), 11 (MOSI), 12 (MISO), 13 (SCK). Ces pins peuvent servir pour la communication avec d'autres périphériques SPI.
- LED : 13. C'est une LED sans fonction spéciale, connectée à la sortie digitale 13.

La carte Arduino Nano a 8 entrées analogiques, chacune ayant une résolution de 10 bits (1024 valeurs). Les pins analogiques 6 et 7 sont les seules qui ne peuvent pas être reconfigurées en pins digitales.

Certains pins ont également des fonctions spéciales :

- I²C : 4 (SDA) et 5 (SCL). Supporte la communication I²C (TWI) avec des composants externes, en utilisant la librairie Wire.

Il existe quelques autres pins spéciaux sur le Nano :

- AREF : Voltage de référence pour les entrées analogiques.
- RESET : passez cette ligne à LOW pour redémarrer le microcontrôleur. Habituellement utilisé pour rajouter une fonction ou un bouton de reset directement sur les shields/modules.

d) Programmation de la carte Arduino Nano

La carte Arduino Nano peut être programmée à l'aide du logiciel de programmation gratuit fourni par Arduino. Le microcontrôleur ATmega328 de la carte Arduino Nano possède un bootloader qui permet le téléchargement de nouveau code sans l'aide d'un programmateur matériel supplémentaire. Il est cependant possible de contourner le bootloader et de programmer le microcontrôleur directement grâce au connecteur ICSP (In-Circuit Serial Programming).

Le programme introduit dans la mémoire flash de la carte Arduino Nano a pour rôle les fonctions suivantes :

- Echange d'informations entre la carte principale et la carte annexe
- Gestion du capteur de température et récolte de la température mesurée
- Gestion du capteur de pression et récolte de la pression mesurée
- Gestion des deux capteurs a ultra son et récolte de la mesure des distances
- Gestion du module GY 521 et récolte de la vitesse angulaire de la roue motrice

II.2.2 Les modules connectés à la carte annexe

II.2.2.1 Le module GY 521

C'est le même que celui utiliser pour la carte principale (voir page 89).

a) Connexion à la carte annexe

Le module GY 521 est connecté à l'Arduino nano (figure 48) par les ports analogiques A4, A5 et les ports d'alimentation 5V et 0V.

Les ports analogiques A4, A5 seront configurés dans le programme pour qu'il fonctionne en mode I2C, ainsi A5 sera un SCL et A4 sera un SDA.

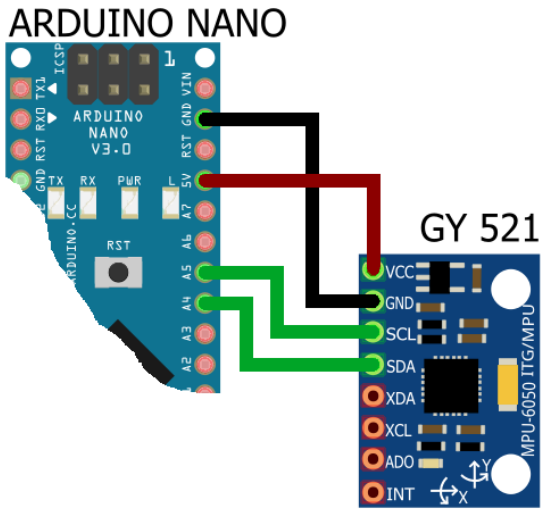


Figure 48 : Connexion du module GY 521 a l'Arduino nano

b) Utilisation dans le système

Le GY521 est utilisé pour déterminer l'angle par rapport à l'axe ox du pneu en rotation. Pour pouvoir utiliser le module GY521, on fait appel à la librairie « Wire.h », et on définit les constantes et variables suivantes, en écrivant :

```
const int MPU_addr=0x68;          // définition de l'adresse I2C du GY521
int16_t AcX,AcY,AcZ;              // Variable de stockage pour accéléromètre
int16_t Tmp;                      // Variable de stockage pour température
int16_t GyX,GyY,GyZ;              // Variable de stockage pour gyroscope
int minVal=265;                   //
int maxVal=402;                   //
double x;                          // Variable de stockage de l'angle/l'abscisse
double y;                          // Variable de stockage de l'angle/l'ordonnée
double z;                          // Variable de stockage de l'angle/la cote
```

Et dans la fonction d'initialisation on initialise par.

```
Wire.begin();                          // démarrage de I2C
Wire.beginTransmission(MPU_addr);      // transmettre à l'adresse du GY521
Wire.write(0x6B);
Wire.write(0);                         // mettre à zéro (réveille le MPU-6050)
Wire.endTransmission(true);
```

II.2.2.2 HC-SR04

Le capteur HC-SR04 est un capteur à ultrason (figure 54), utilise les ultrasons pour déterminer la distance d'un objet. Ce capteur fonctionne avec une tension d'alimentation de 5 volts, et dispose d'un angle de mesure d'environ 15° et permet de faire des mesures de distance entre 2 centimètres et 4 mètres avec une précision de 3mm théorique. Il offre une excellente plage de détection sans contact, avec des mesures de haute précision et stables. Son fonctionnement n'est pas influencé par la lumière ou des matériaux sombres, bien que des matériaux comme les vêtements puissent être difficiles à détecter.

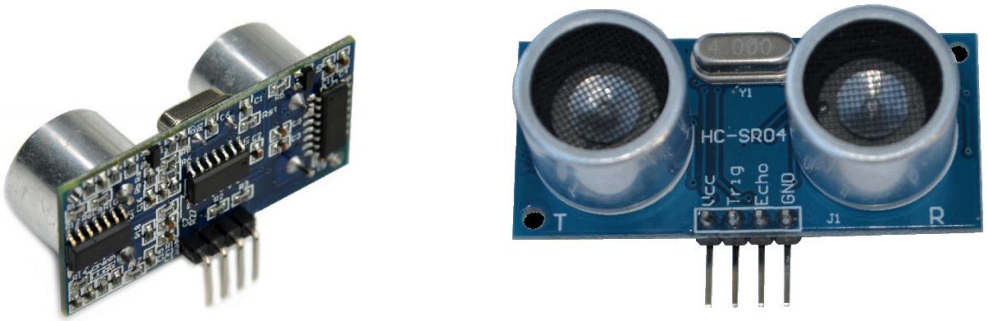


Figure 49 : Le capteur HC-SR04

a) Caractéristiques

- Dimensions : 45 mm x 20 mm x 15 mm
- Tension de fonctionnement : 5V DC
- Courant de fonctionnement : 15mA
- Plage de mesure : 2 cm à 400 cm
- Résolution de la mesure : 0.3 cm
- Angle de mesure efficace : 15 °
- Fréquence de fonctionnement : 40kHZ
- Largeur d'impulsion sur l'entrée de déclenchement : 10 µs (Trigger Input Pulse width)

b) Broches de connexion

- Vcc = Alimentation +5 V DC
- Trig = Entrée de déclenchement de la mesure (Trigger input) D3
- Echo = Sortie de mesure donnée en écho (Echo output) D2
- GND = Masse de l'alimentation

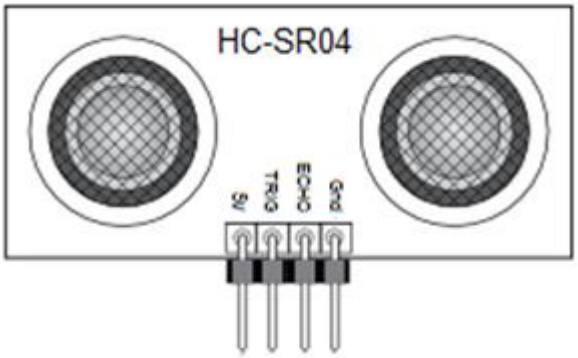


Figure 50 : Broches de connexion

c) Connexion à la carte annexe

Pour le premier HC-SR04, Trig est connectée à D5 et Echo à D4 de carte Arduino Nano. Pour le deuxième HC-SR04, Trig est connectée à D3 et Echo à D2 de carte Arduino Nano (figure 56).

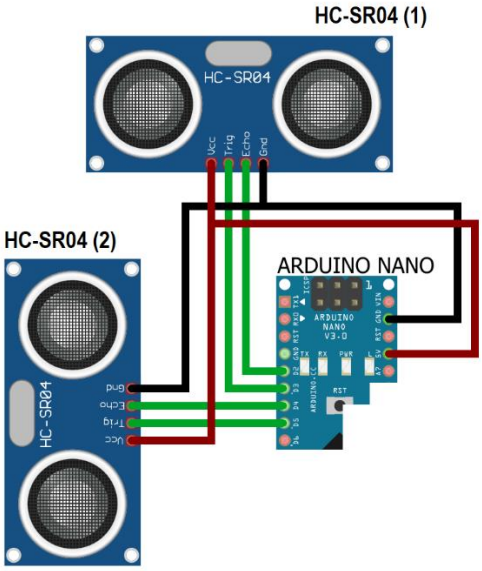


Figure 51 : Connexion du HC-SR04 a l'Arduino nano

d) Principe de fonctionnement du capteur

Le principe de fonctionnement du capteur est entièrement basé sur la vitesse du son. Pour déclencher une mesure, il faut envoyer une impulsion "high" (5V) d'au moins 10µs sur l'entrée TRIGGER du capteur. Le capteur émet alors une série de 8 impulsions ultrasoniques de 40kHz à travers le transducteur émetteur et attend le signal réfléchi à travers le transducteur récepteur. Lorsque celui-ci est détecté, il envoie un signal "high" sur la sortie "Echo", dont la durée est proportionnelle à la distance mesurée (figure 57).

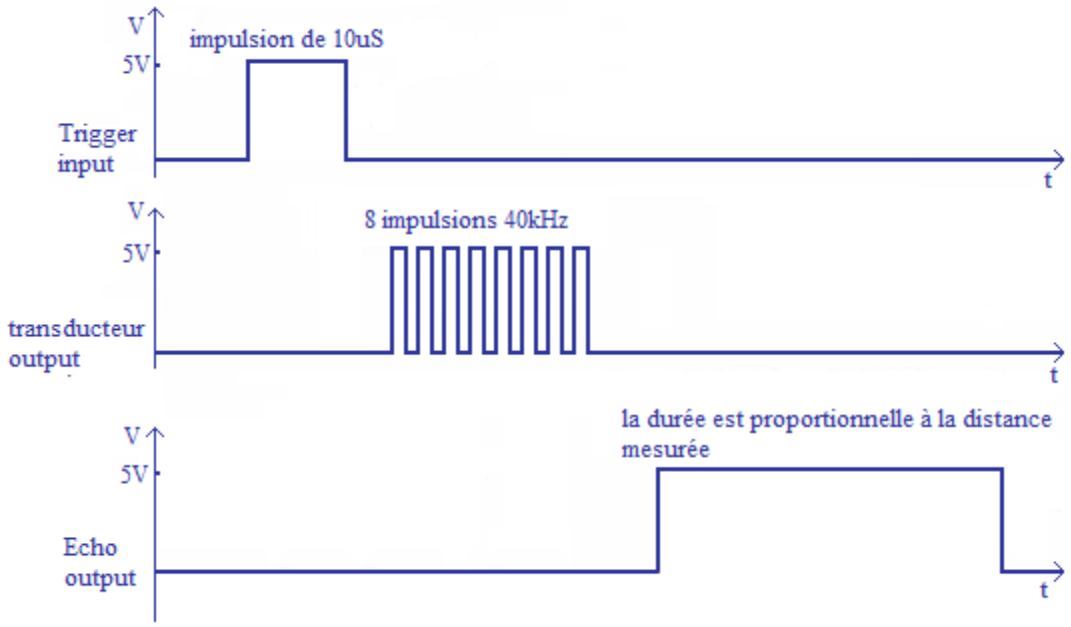


Figure 52 : Illustration du signal TRIGGER et ECHO

La distance parcourue par un son se calcule en multipliant la vitesse du son, par le temps de propagation, soit :

$$d = v \cdot t \quad (1)$$

d : distance en cm

v : vitesse en cm/s

t : temps en s

Le HC-SR04 donne une durée d'impulsion proportionnelle à la distance mesurée multiplié par deux car le son fait un aller-retour, cette durée d'impulsion est mesurée en µs. La distance vaut donc la moitié (figure 58).

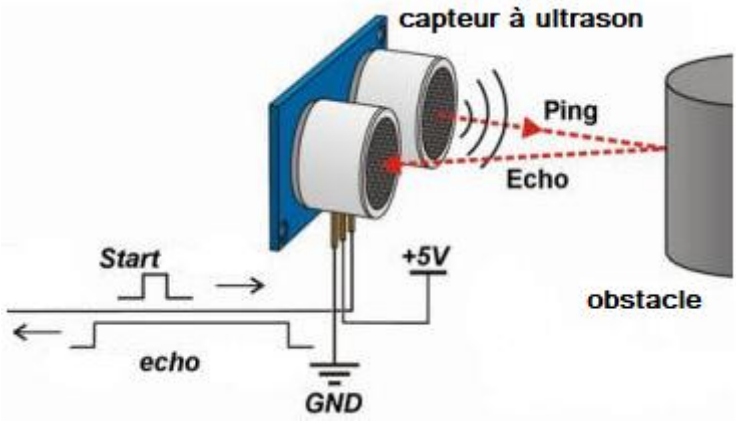


Figure 53 : Illustration de fonctionnement du capteur

$$d = v \cdot \frac{t}{2} \cdot \frac{1}{1000} = v \cdot \frac{t}{2000} \qquad (2)$$

d : distance en mm

v : vitesse en mm/µs

t : temps en µs

Dans l'air, à pression atmosphérique normale, la vitesse du son est donnée par la formule :

$$v = 331{,}35 + 0{,}607 \cdot T \qquad (3)$$

Où v est la vitesse du son exprimée en mètres par seconde, et T est la température, exprimée en degrés Celsius, l'hygrométrie influe peu.

Les équations (2) et (3) donnent :

$$d = (331{,}35 + 0{,}607 \cdot T) \cdot \frac{t}{2000} \qquad (4)$$

d : distance en mm

v : vitesse en mm/µs

t : temps en µs

T : température en °C

e) Utilisation dans le système

Le système au niveau des cartes annexes utilise deux capteurs à ultrason type HC-SR04 pour la mesure et le calcul des rayons dynamique et statique des roues motrices.

Pour pouvoir utiliser le capteur HC-SR04, On définit les ports de connexion des deux capteurs à l'Arduino nano en écrivant :

```
#define trigPin_1 D5
#define echoPin_1 D4
#define trigPin_2 D3
#define echoPin_2 D2
```

La fonction setup() initialise les ports séries, met la broche TRIGGER du capteur en sortie et à LOW, et met la broche ECHO du capteur en entrée.

```
pinMode(trigPin_1, OUTPUT);
pinMode(echoPin_1, INPUT);
pinMode(trigPin_2, OUTPUT);
pinMode(echoPin_2, INPUT);
```

Pour la gestion des capteurs HC-SR04 on utilise la fonction pulseIn() qui accepte au maximum trois paramètres et retourne un nombre entier long (unsigned long) correspondant à la durée de l'impulsion mesurée en microsecondes.

II.2.2.3 Capteurs de température

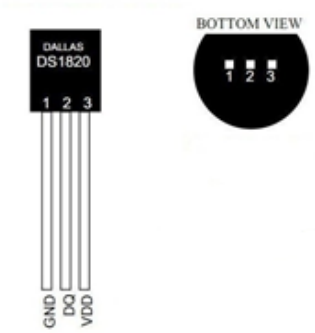


Figure 54 : Capteur de température numérique

Le capteur DS18B20 du fabricant Maxim (figure 57) est un capteur de température numérique intégrant dans une seul puce tout le nécessaire requis pour réaliser la mesure de température, il intègre un capteur analogique, un convertisseur analogique / numérique, une électronique de communication et d'alimentation.

Il possède une résolution numérique de 12 bits (programmable) avec une plage de mesure de -55°C à +125°C et communique via un bus 1-Wire. La précision analogique du capteur est de 0,5°C entre -10°C et +85°C, ce qui le rend très intéressant pour une utilisation "normale".

Le capteur DS18B20 existe dans le commerce en deux versions : en boitier TO-92 (format transistor, voir figure 59) pour des utilisations standards en intérieur, ou en format "sonde étanche" pour des applications en milieu humide / extérieur. *Les deux formats sont strictement identiques d'un point de vue technique.*

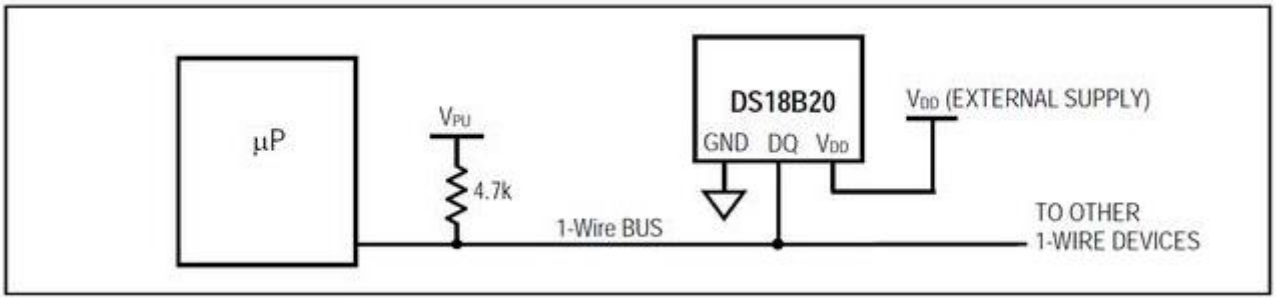


Figure 55 : Brochage des capteurs DS18B20

Le capteur DS18B20 est un capteur One-Wire (figure 60), cela signifie qu'il communique avec une carte maître au moyen d'un bus One -Wire. Plusieurs capteurs peuvent être reliés sur un même bus One -Wire. De plus, chaque capteur possède une adresse unique programmée lors de sa fabrication, il n'y a donc pas de risque de conflit.

Un bus One-Wire est composé classiquement de trois fils : un fil de masse, un fil d'alimentation (5 volts) et un fil de données. Un seul composant externe est nécessaire pour faire fonctionner un bus One-Wire : une simple résistance de tirage de 4.7kΩ sur la broche de données reliée à l'alimentation suffit.

Comme tout périphérique 1-Wire, le DS18B20 contient un "scratchpad" qui est une sorte de mémoire tampon sécurisée ou l'on peut venir lire et / ou écrire des données. C'est dans cette mémoire qu'on vient lire les données de mesures et écrire les informations de configuration du capteur.

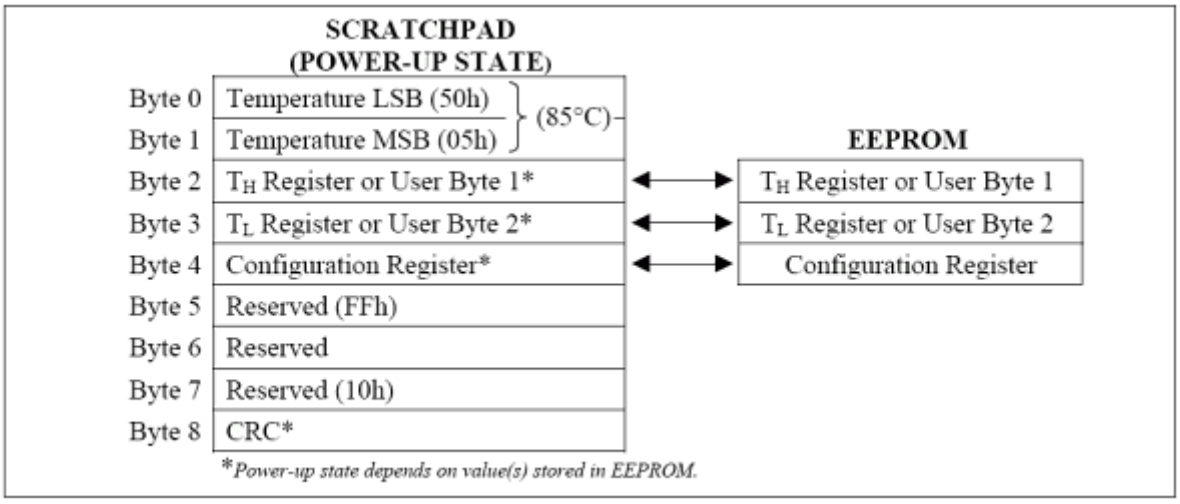


Figure 56 : Mémoire interne du capteur DS18B20

Le scratchpad (figure 59) du capteur DS12B20 est divisé en quatre parties :

- Le résultat de la dernière mesure de température (deux octets),
- Deux octets à usages divers (le capteur dispose d'un mode "alarme", mais cela ne sera pas traité dans ce tutoriel),
- Le registre de configuration du capteur,
- Une somme de contrôle.

Ce qui nous intéresse, ce sont les deux premiers octets qui contiennent le résultat de la mesure de température. Le reste ne nous intéresse pas.

a) Caractéristiques

- Plage de mesure : -55°C à +125°C
- Tension de fonctionnement : 5V DC
- Précision analogique du capteur : 0,5°C entre -10°C et +85°C

b) Connexion à la carte annexe

Le capteur DS12B20 est connecté à l'Arduino nano (figure 60) par le port logique D6 et les ports d'alimentation 5V et 0V. Le port logique D6 sera configure dans le programme pour qu'il fonctionne en onewire.

c) Utilisation dans le système

Le système utilise le capteur DS12B20 (figure 60) au niveau des cartes annexes pour connaitre la température à l'intérieure des roues motrices pour le calcul de la

vitesse du son (voir équation 3 page 108).

Pour pouvoir utiliser le capteur DS12B20, la librairie « DallasTemperature » et la librairie « OneWire » sont nécessaires.

On définit le port de connexion du capteur DS12B20 à l'Arduino nano en créant en même temps une instance à partir de l'objet OneWire appelé oneWire en écrivant :

OneWire oneWire(D6);

Pour la gestion des capteurs DS12B20 on utilise les fonctions de la librairie DallasTemperature.

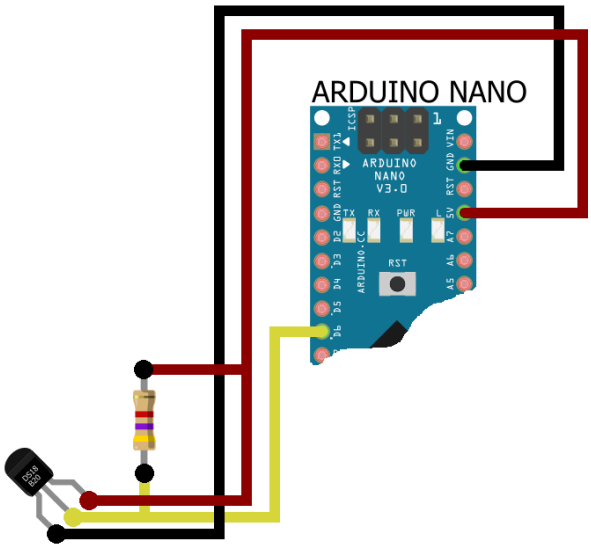


Figure 57 : Connexion du DS12B20 a l'Arduino nano

II.2.2.4 Le Capteur de pression MPX 5700AP

C'est un capteur de pression en silicium monolithique de pointe, reposant sur l'effet piézorésistive du Silicium (figures 61, 62, 63), c'est à dire le changement de conductibilité d'un matériau dû à une contrainte mécanique. Il est conçu pour un large éventail d'applications, en particulier celles qui utilisent un microcontrôleur ou un microprocesseur avec entrées A / N. Ce capteur est prévu pour mesurer des pressions absolues variant de 150mb à 7000mb. Ce transducteur mono-élément breveté combine des techniques avancées de micro-usinage, de métallisation en couches minces et de traitement bipolaire pour fournir un signal de sortie analogique précis de haut niveau proportionnel à la pression appliquée.

Figure 58 : Capteur de pression MPX 5700AP

a) Caractéristiques

- Conditionnement de signal sur puce, calibré et compensé en température
- Erreur maximum de 2.5% entre 0°C et 85°C
- Il est parfaitement adapté pour les systèmes basés sur un microprocesseur ou un microcontrôleur.
- Jauge de contrainte à cisaillement silicium breveté
- Gamme de pression de -15KPa à 700KPa
- Gamme de tension d'alimentation simple de 4.75VDC à 5.25VDC
- Sensibilité de 6,4mV/KPa
- Temps de réponse 1ms
- Elément unicorps en époxy durable
- Gamme de température d'utilisation de -40°C à 125°C

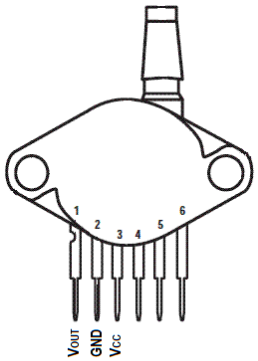


Figure 59 : Brochage du MPX 5700AP

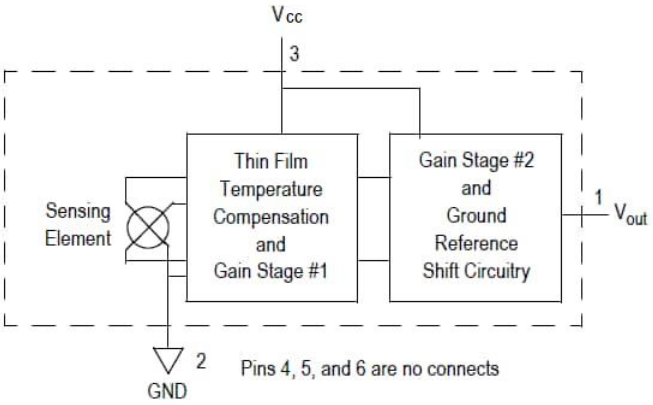


Figure 60 : Schémas électrique du MPX 5700AP

b) Connexion à la carte annexe

Le MPX 5700AP est connecté à l'Arduino nano par le port analogique A0 et les ports d'alimentation 5V et 0V.

Puisqu'il s'agit d'un capteur analogique, il doit posséder des filtres adéquats pour que la mesure soit la plus précise possible. Trois solutions sont proposées : le filtrage RC (figure 65), le filtrage par ampli-op et le filtrage logiciel. Pour la simplicité du montage, on utilise un filtrage RC et un filtrage logiciel. Le filtrage logiciel se fera par échantillonnage (64 échantillons), qui permet d'améliorer la précision à 1mV (pic à pic).

Afin d'améliorer encore la précision, on n'utilise l'alimentation régulée de 5V de l'Arduino nano.

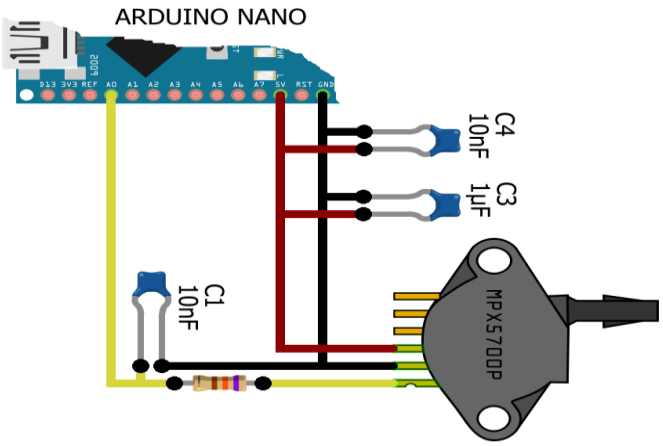


Figure 61 : Connexion du mpx5700 a l'Arduino nano

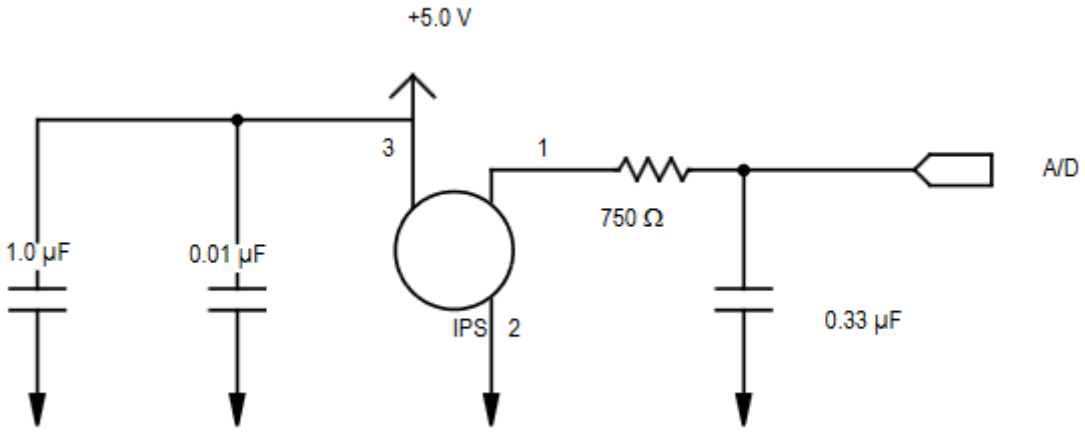


Figure 62 : Capteur de pression intégré avec filtre RC (Reodique, Schultz, 2005)

c) Utilisation dans le système

Le capteur mpx5700 mesure une pression entre 0.15 et 7bar, et c'est cette plage qui va servir à la conversion A/D. Le capteur est connecté à l'Arduino nano au port analogique A0 (figure 64).

Pour pouvoir utiliser le capteur mpx5700, On définit le port de connexion du capteur à l'Arduino nano et la variable de mesure en écrivant :

```
int sensorPin = A0;
int sensorValue = 0;
```

Pour la gestion mpx5700 on utilise la fonction analogRead () qui lit la valeur de la tension présente sur la broche spécifiée, ici A0.

II.2.2.5 Le module nRF24L01

C'est le même que celui utiliser pour la carte principale (voir page 79).

a) Connexion à la carte annexe

Le tableau 8 présente les connexions entre Arduino nano et nrf24L1

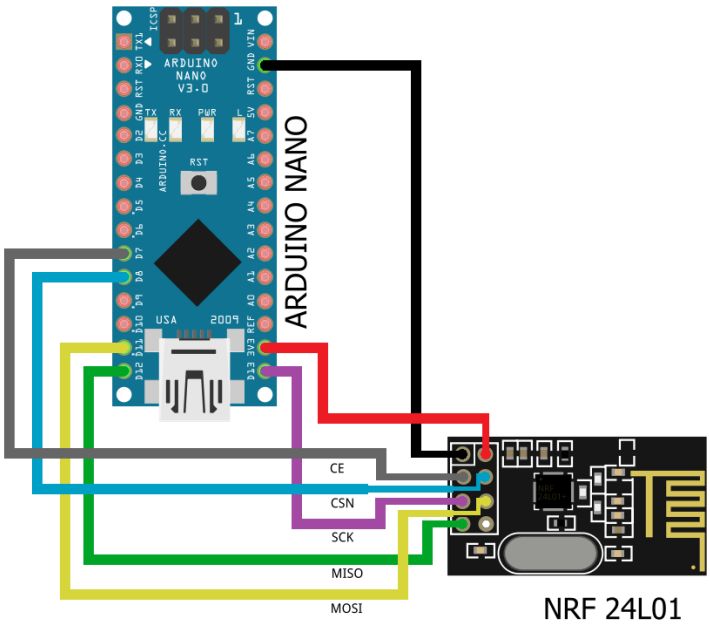


Figure 63 : Connexion du NRF24L01 a l'Arduino nano

Tableau 8: Connexion NRF24L01 avec Arduino nano

Broche	NRF24L01+	Arduino nano
1	GND	GND
2	VCC	3.3V
3	CE	D7
4	CSN	D8
5	SCK	D13
6	MOSI	D11
7	MISO	D12
8	IRQ	-

b) Utilisation dans le système

Pour pouvoir utiliser le module NRF24L01, les librairies suivantes « SPI.h, Mirf.h, nRF24L01.h, et MirfHardwareSpiDriver.h » doivent être ajoutées au sketch par l'ajout des lignes suivantes.

```
#include <SPI.h>
#include <Mirf.h>
#include <nRF24L01.h>
#include <MirfHardwareSpiDriver.h>
```

Et dans la fonction d'initialisation on initialise par.

```
Mirf.cePin = 7;
Mirf.csnPin = 8;
Mirf.spi = &MirfHardwareSpi;
Mirf.init();
Mirf.channel = 1;
Mirf.payload = 32;
Mirf.config();
Mirf.setTADDR((byte *) "nrf02");
Mirf.setRADDR((byte *) "nrf01");
```

Pour la gestion du module NRF24L01on utilise les fonctions de la bibliothèque Mirf disponible sur GitHub.

III.Structure du firmware de la carte mère

La réalisation du firmware a été faite en utilisant l'open source Arduino IDE 1.8.5. Le firmware (figure 64, annexe C) de la carte mère recueille des mesures de données (r rayon dynamique de la roue motrice, α position angulaire de la roue motrice, p la pression de la roue motrice) à partir des cartes annexes (Arduino nano) à travers le module RF. Il effectue des mesures de la force de traction P, de l'angle de pente du sol et les stocke dans la mémoire, affiche les valeurs actuelles des mesures et exécute les requêtes du clavier relier à l'Arduino Mega et les requêtes de l'application Android (pour le programme voir annexe).

Au lancement de la carte mère Arduino Mega, le firmware commence par la déclaration des libraires nécessaires, la déclaration des variables et des constantes ainsi que des instances, puis lance l'opération d'initialisation (setup) et termine par le lancement de la boucle de traitement (loop).

La fonction d'initialisation (setup), initialise la transmission série, l'horloge temps réel, le module NRF24L01, le module gy521, le Module Bluetooth HC-05, l'afficheur LCD. Après que la fonction d'initialisation se termine la fonction boucle (loop) se lance automatiquement, cette fonction commence par une demande de lecture des données de mesures des cartes annexes, quand la transmission des données se termine elle passe à la récupération des données de mesures du hx711 pour les convertir en daN, elle récupère aussi les données de mesures du module GY521 et calcule l'angle de pente du terrain sur lequel se déplace le tracteur, puis elle traite les demandes du clavier et les demandes de l'application Android et affiche les résultats des mesures.

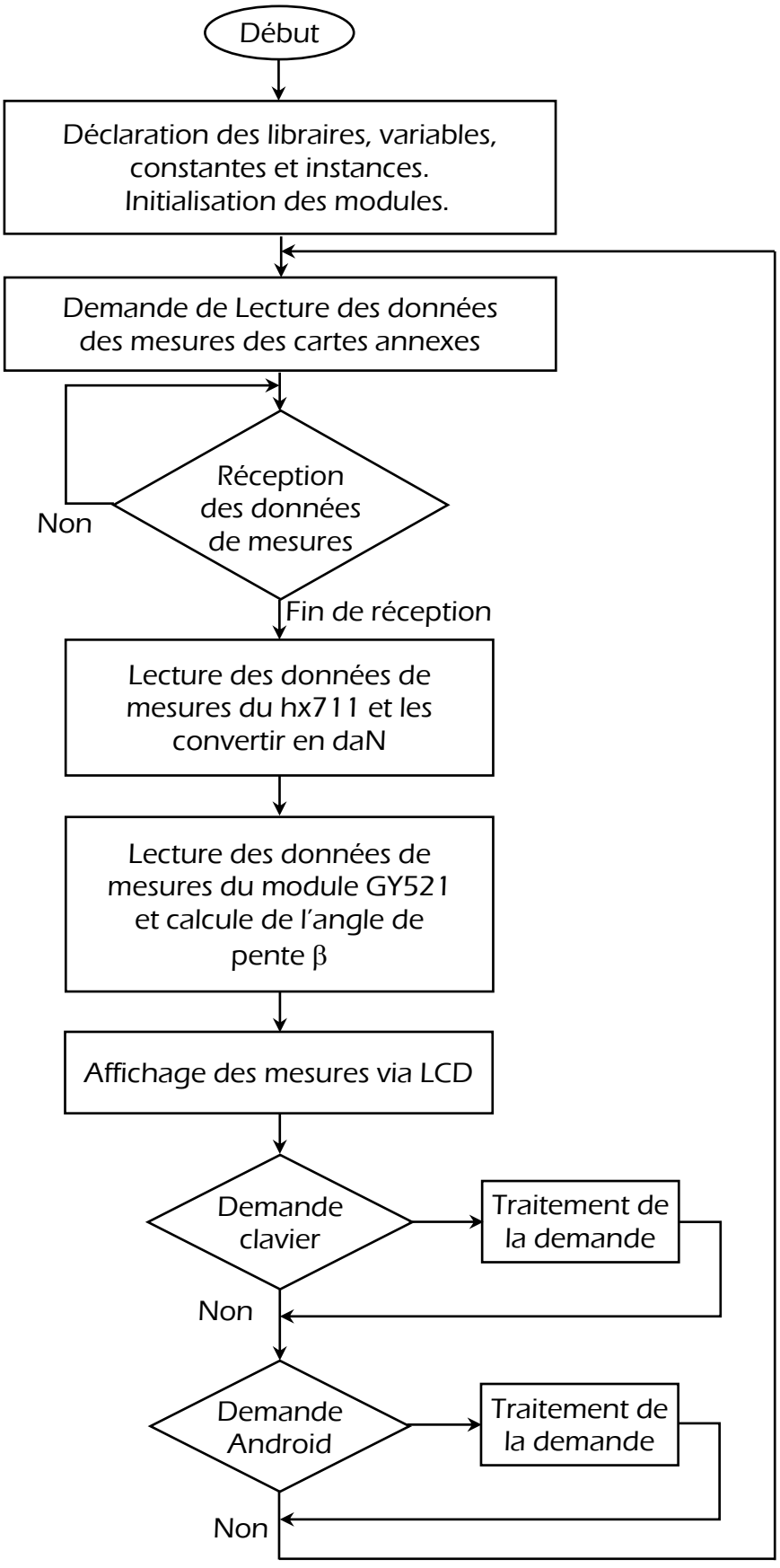


Figure 64 : Organigramme du firmware de la carte principale

IV. Structure du firmware des cartes annexes

La réalisation du firmware a été faite en utilisant l'open source Arduino IDE 1.8.5. Le firmware de chaque carte annexe (figure 65, annexe B) effectue des mesures au niveau de la roue motrice, il mesure le rayon dynamique et statique, α position angulaire, p pression et T température de la roue motrice. Les mesures sont stockées dans la mémoire et envoyer à la carte mère par l'intermédiaire du module radio fréquence nRF24L01(pour le programme voir annexe).

Au lancement des cartes annexes, le firmware commence par la déclaration des libraires nécessaires, la déclaration des variables et des constantes ainsi que des instances, puis lance l'opération d'initialisation (setup) et termine par le lancement de la boucle de traitement (loop).

La fonction d'initialisation (setup), initialise la transmission série, module NRF24L01, le capteur de température Dallas, le module gy521.

Après que la fonction d'initialisation se termine la fonction boucle (loop) se lance automatiquement, elle commence par la récupération des données de mesures du module GY521 pour le comptage du nombre de tour de la roue motrice, la lecture de la pression et de la température interne de la roue motrice et enfin la variation du rayon dynamique à l'aide des ultrasons, puis Transmission des données vers la carte principale.

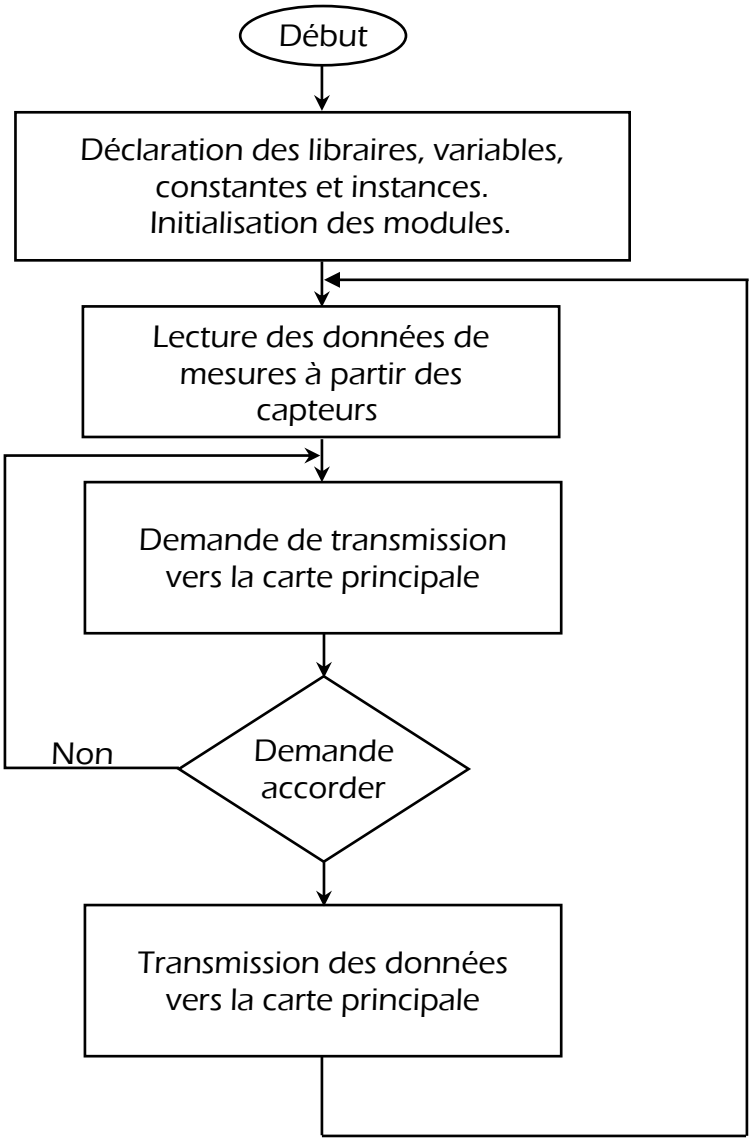


Figure 65 : organigramme du firmware de la carte annexe

V. Structure de l'application Android

L'application Android (Tractor performance and soil characteristics) est le dernier maillon du système, son rôle est la récolte des données de la carte principale Arduino Mega 2560 qui se trouve au niveau du tracteur et cela à travers le module Bluetooth. Ces données qui sont (vitesse réelle, vitesse apparente, rayon dynamique, force de traction, rayon de pente, pression et température à l'intérieur de la roue motrice) servirent pour le calcul des paramètres pour la prédiction des performances de traction des tracteurs, le calcule et l'appréciation des caractéristiques mécaniques et physiques du sol.

Le programme de l'application Android (voir organigramme figure 68, programme annexe D) a été écrit en java dans (Android Studio IDE 3.0.1).

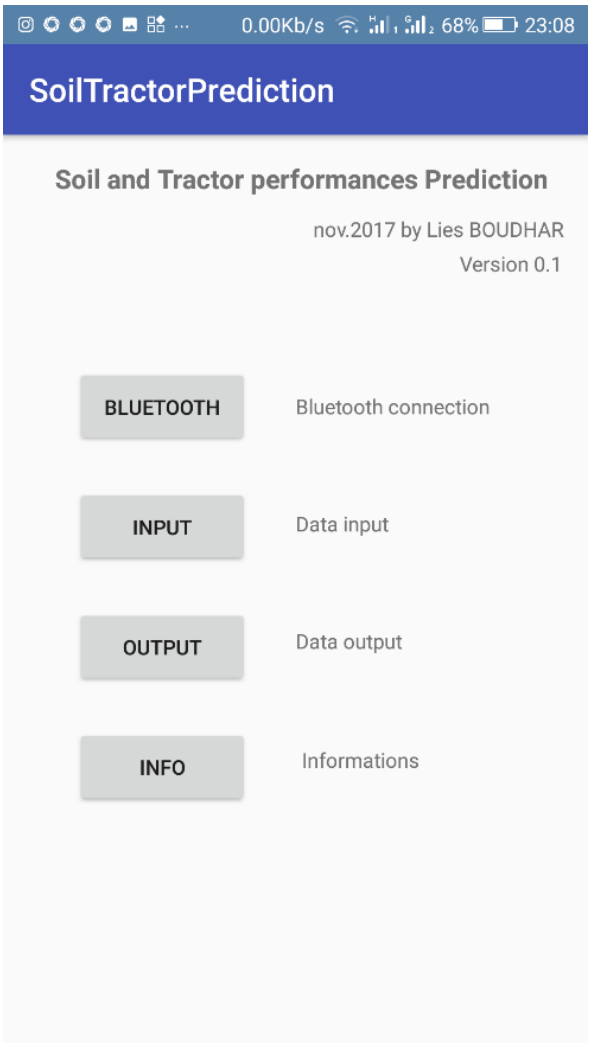


Figure 66 : activité principale de l'application Android

L'application Android (Tractor performance and soil characteristics) possède cinq activités (l'activé principale, l'activé Bluetooth, l'activé input, l'activé output, l'activé info).

On trouve dans l'activé principale (Figure 66) quatre boutons cliquables nommées selon l'activité ciblée.

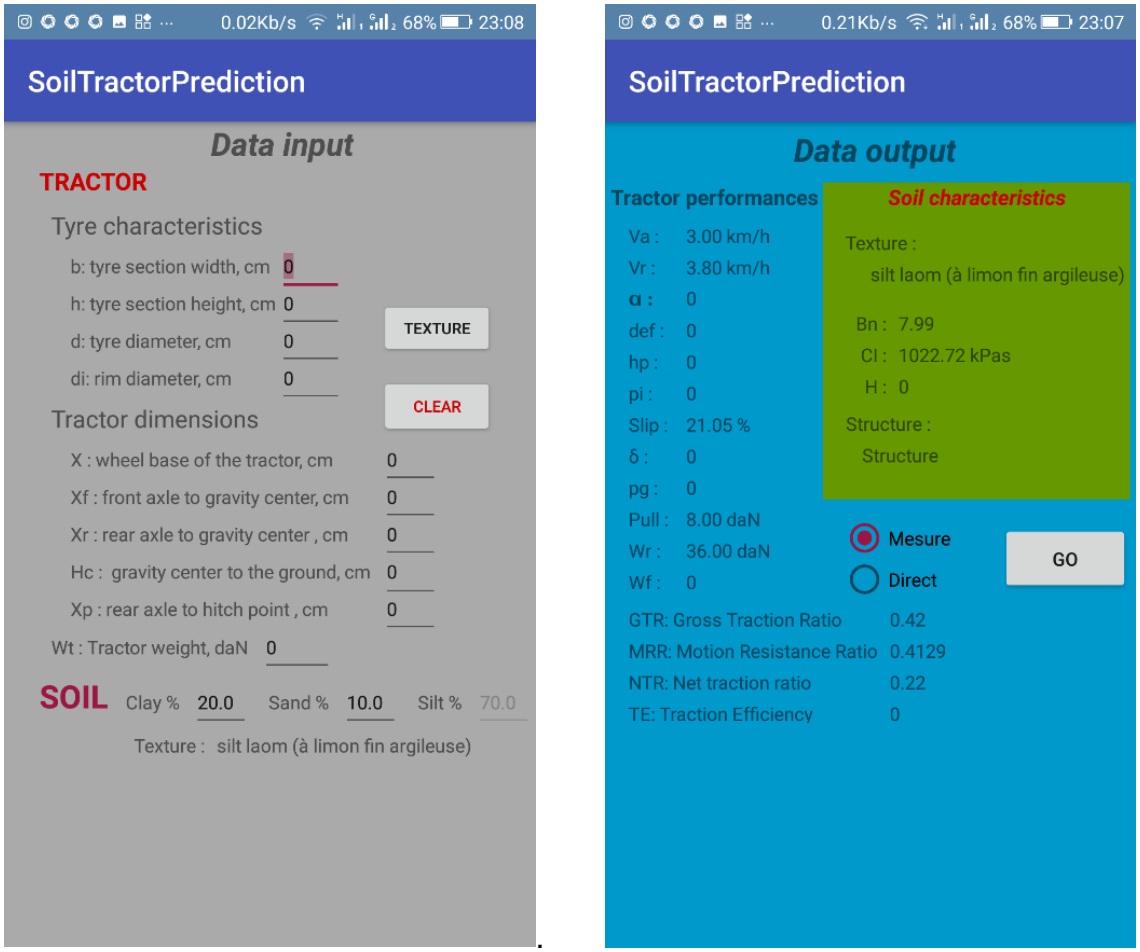


Figure 67 : activités input et output de l'application Android

L'activé data input (figure 67 de gauche) sert à l'introduction des constantes et la détermination de la texture du sol sur lequel se déplace le tracteur.

L'activé data output (figure 67 de droite) sert à l'affichage des résultats de mesures et de calculs.

L'activé Bluetooth sert pour la connexion a un récepteur émetteur Bluetooth.

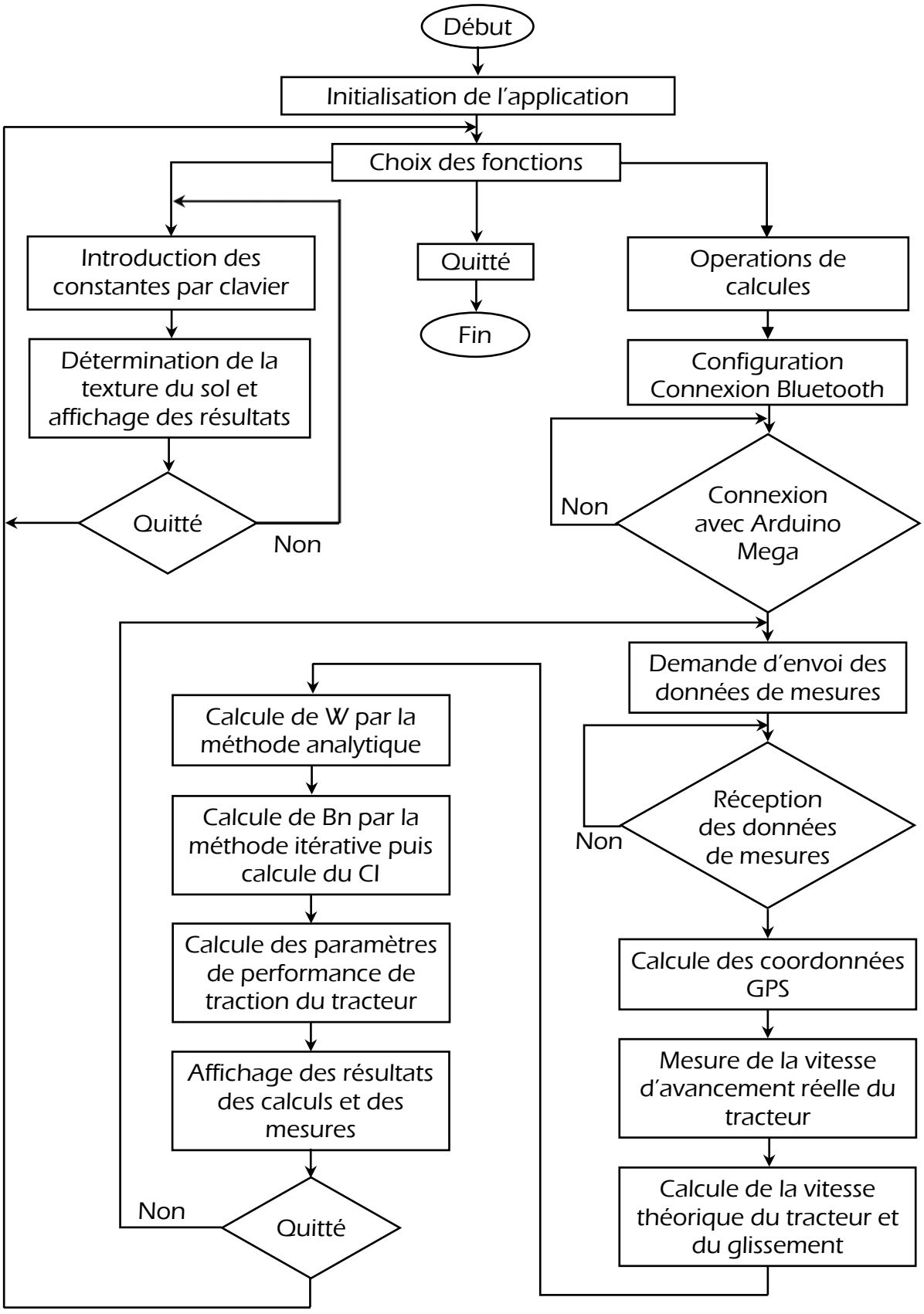


Figure 68 : Organigramme de la structure de l'application Android

VI. Conclusion

Durant la réalisation du projet certains modules ont été changés par d'autres modules plus récents ou plus efficaces ou par leurs manques sur le marché. Le module d'affichage graphique a été remplacé par un autre plus récent et plus efficace un (0.96" 128×64 White OLED). Le module GPS, est retiré de la carte principale pour être remplacé par celui du smartphone, même chose pour la carte mémoire sd pour le stockage des données de mesures.

Par manque de matériel nécessaire et disponible, le système en entier n'a pas pu être testé sur le terrain en grandeur réelle. La grandeur réelle ne concerne que le mécanisme de mesure de la force de traction, par contre le capteur de force a été testé seul. Les différentes parties du système à savoir la carte principale, les cartes annexes, et l'application Android ont été testées séparément.

Le firmware de la carte principale fonctionne normalement et contrôle bien les modules connectés à cette dernière, le transfert des données de la carte annexe ce fait sans perte, même chose pour le firmware de la carte annexe. Mais il n'en est pas moins que ces firmwares nécessitent des améliorations dans les fonctions de mesures.

L'application Android quant à elle, est en version de test, et nécessite beaucoup d'amélioration dans la facilité d'utilisation. Pour le moment, l'application réalise les fonctions voulues à savoir, mesure et calcule l'indice de cône, la force de traction, les performances de traction, la texture du sol, brixius number.

Pour l'humidité du sol, il existe des modèles pour la prédire à partir de l'indice de cône, comme celui de (Collins, 1971; Wells and Treesuwan, 1977) qui disent que

$$ln(CI) = C_1 + C_2 \times ln(MC)$$

Ou, CI est de l'indice de cône, MC est l'humidité du sol, C1 et C2 des constantes qui dépendent du type de sol.

Comme dans notre cas on utilise un capteur pour mesurer la conductivité électrique du sol, par laquelle on peut estimer l'humide du sol après étalonnage.

CONCLUSION GENERALE :

Si une seule conclusion était à garder de cette thèse, c'est que l'utilisation des nouvelles technologies à savoir l'électronique, la robotique, l'informatique est une chose inévitable et incontournable en agriculture de futur et en sciences agronomiques, et ce serait aveugle d'esprit de dire l'inverse.

Le système de capteurs pour l'analyse des propriétés physico-mécaniques du sol en temps réel et en continu, est un système conçu spécialement pour être évolutif de côté hardware que de côté software. Côté hardware, il y a possibilité de lui ajouter des capteurs ou de lui changer ses capteurs avec des capteurs plus performants de moment que, la méthode de dialogue des capteurs avec la carte Arduino est du type I2C ou one wire. Côté software (application Android) on peut modifier les équations de prédiction et de calculs ou ajouté d'autres pour augmenter les paramètres d'analyse.

Ce système entre dans le cadre du développement de l'agriculture numérique et de l'agriculture de précision afin de mettre à la disposition des chercheurs, des industriels, et agriculteurs, un outil performant capable de rependre aux exigences de ces derniers chacun dans son domaine.

Pour les chercheurs, il servira pour le développement de la recherche sur les performances de traction des engins qui roulent sur des sols accidentés comme les sols agricoles, afin de vérifier les performances des nouveaux modèles ou de les comparer avec des modèles existants, et de connaitre en temps réel les caractéristiques mécaniques d'un sol donné.

Pour les agriculteurs ; le système ainsi réalisé, suit et calcule le glissement à partir de la vitesse de rotation des roues motrices et de la vitesse d'avancement réelle du tracteur, donc si on lui ajoute un mécanisme qui modifie la profondeur du travail du sol, on peut contrôler et maintenir le glissement des roues motrices dans une plage près définie en ajustant automatiquement la profondeur de labour si c'est un labour, alors on agira directement sur la consommation du carburant.

Ce travail de recherche a fait l'objet d'une expérience très intéressante, qui nous a permis d'apprendre et d'améliorer nos connaissances et nos compétences dans le domaine du sol agricole, du tracteur, de la programmation, de l'électronique numérique, et des capteurs.

Cependant l'amélioration de ce système reste très envisageable et ouverte, à savoir la prédiction de la densité et de la structure du sol par exemple.

Grâce à ce travail de recherche on a pu avoir une idée plus claire et plus approfondie des applications de l'électronique, des capteurs, et de l'informatique sur l'agriculture et sur la science agronomique en général. On peut conclure avec fermeté qu'il ne peut y avoir une avancée en science agronomique sans l'utilisation des capteurs, de l'électronique, et de l'informatique.

REFERENCES BIBLIOGRAPHIQUES

[1] Abbaspour-Gilandeh Y., 2009. On-the-go soil mechanical strength measurement at different soil depths. Journal of Food, Agriculture & Environment. Vol.7 (3&4): 696-699.

[2] Adamchuk, V. I., M. T. Morgan and J. Lowenberg-DeBoer. 2004. A Model for Agro-Economic Analysis of Soil pH Mapping. Precision agriculture 5:111-129.

[3] Adamchuk, V.I., C.R. Hempleman, and D.G. Jahraus. 2009. On-the-go capacitance sensing of soil water content. Paper No. MC09-201. St. Joseph, Michigan: ASABE.

[4] Agüera J., Carballido J., Gil J., Gliever C. J., Perez-Ruiz M., 2013. Design of a Soil Cutting Resistance Sensor for Application in Site-Specific Tillage. Sensors. 13: 5945-5957.

[5] Amara M., 2007. Contribution à la modélisation interface outils aratoires-sol Optimisation de la forme et de l'effort de résistance à la traction des corps de charrues à socs et des outils à dents. Thèse d'État. Institut national Agronomique Alger.

[6] Andrade-Sanchez, P., S. K. Upadhyaya and B. M. Jenkins. 2007. Development, construction, and field evaluation of a soil compaction profile sensor. Transactions of the ASABE 50(3): 719-725.

[7] ASAE: St. Joseph, MI, USA, 2002. Procedures for Using and Reporting Data Obtained with the Soil Cone Penetrometer; ASAE Standards EP542,

[8] Ayers, P.D. and J.V. Perumpral. 1982. Moisture and density effect on cone index. Transactions of the ASAE, 25: 1169-1172.

[9] Ayers, P.D., Perumpral, J.V., 1982. Moisture and density effect on cone index. ASAE Trans. 25 (5), 1169-1172.

[10] Bédard, Y., S. Tessier, C. Laguë, Y. Chen, and L. Chi. 1997. Soil compaction by manure spreaders equipped with standard and oversized tires and multiple axles. Transactions of the ASAE, 40: 37-43.

[11] BENNIE, A. T. P., 1991. Growth and mechanical impedance. Y. Waisel, A. Eshel ET U. Kafkafi, ed. Plant Roots the Hidden half. 393-414.

[12] BILLOT J. F., 1982. Les applications agronomiques de la pénétromètrie à l'étude de la structure des sols travaillés. Centre National du Machinisme Agricole du Génie Rural des Eaux et Forêts. C.E.M.A.G.R.E.F., 92160 ANTONY

[13] Botta, G.F., D. Jorajuria, R. Balbuena, M. Ressia, C. Ferrero, H. Rosatto, and M. Tourn. 2006. Deep tillage and traffic effects on subsoil compaction and sunflower (Helianthus annus L.) yields. Soil & Tillage Research, 91: 164-172.

[14] Brixius W.W, 1987. Traction prediction equations for bias ply tires, Paper No. 87-1622, Program Manager, Row-Crop Tractors John Deere Product Engineering Center, Waterloo, Iowa U.S.A.

[15] Busscher, W.J., P.J. Bauer, C.R. Camp, and R.E. Sojka. 1997. Correction of cone index for soil water content differences in a coastal plain soil. Soil & Tillage Research, 43: 205-217.

[16] Campbell, D.J., O'Sullivan M.F., 1991. The cone penetrometer in relation to trafficability, compaction, and tillage. In Soil Analysis: Physical Methods, 399–429. K. A. Smith and C. E. Mullins, eds. New York, N.Y.: Marcel Dekker.

[17] Camuzard J.-P., Le sol, un milieu complexe au pouvoir épurateur limité. ENGREF Paris.

[18] Chen, Y., C. Cavers, S. Tessier, and D. Lobb. 2005. Short-term tillage effects on soil cone index and plant development in a poorly drained, heavy clay soil. Soil & Tillage Research 82(2): 161-171.

[19] Chunga S.O., Sudduthb K.A., Motavallic P.P., Kitchenb N.R., 2013. Relating mobile sensor soil strength to penetrometer cone index. Soil and Tillage Research. 129, 9–18.

[20] Ciza Thomas., 2011. SENSOR FUSION - FOUNDATION AND APPLICATIONS. Edited by Ciza Thomas. Published by InTech Janeza Trdine 9, 51000 Rijeka, Croatia. ISBN 978-953-307-446-7

[21] Collins, J.G., 1971. Forecasting trafficability of soils. Technical Memo. 3-331, U.S. Army Corps of Engineers Waterways Experiment Station, Vicksburg, MS.

[22] De Paul M. A., Bailly M., 2005. À propos de la pression exercée par les pneus, chenilles et sabots. La Forêt Wallonne, (78), 21-33.

[23] Dean, T. J., J. P. Bell and A. J. B. Baty. 1987. Soil moisture measurement by an improved capacitance technique, Part I. Sensor design and performance. Journal of Hydrology 93(1-2): 67-78.

[24] Derdour H., Angers D.A. et Loverdiere M.R., 1993. Caractérisation de l'espace poral d'un sol argileux : Effet de ses constituants et du travail du sol. Can. J. soil sci. (73) 299 – 307.

[25] Doan, V., Y. Chen, and B. Irvine. 2005. Effect of residue type on the performance of no-till seeder openers. Canadian Biosystems Engineering, 47: 2.29-2.35.

[26] Drummond, P.E.; Christy, C.D.; Lund, E.D. Using an Automated Penetrometer and Soil EC Probe to Characterize Rooting Zone. In Proceedings of the Fifth International Conference on Precision Agriculture, Bloomington, MN, USA, July 27–30, 2000; pp. 34-54.

[27] Duchaufour Ph., 1997. Pédologie Tome 2 : Constituants ET propriétés du sol. Elsevier Masson.

[28] Ehlers W., Köpke U., Hesse F., Böhm W., 1983. Penetration resistance and root growth of oats in tilled and untilled loess soil. Soil and Tillage Research. 3(3): 261-275.

[29] Francis, G.S., K.C. Cameron, and R.S. Swift. 1987. Soil physical conditions after six years of direct drilling or conventional cultivation on silt loam soil in New Zealand. Journal of Soil Research, 25: 517-520.

[30] Franzen, H., R. Lal, and W. Ehlers. 1994. Tillage and mulching effects on physical properties of a tropical Alfisol. Soil & Tillage Research, 28: 329-346.

[31] Grenier G., 2014. Les Enjeux de l'Agriculture de Précision. Bordeaux Sciences Agro.

[32] Guerif J., 1983. Le compactage. Séminaire CEE Agimed.

[33] Hamza, M. A. and W. K. Anderson. 2005. Soil compaction in cropping systems – A review of nature, causes and possible solutions. Soil & Tillage Research 82(2): 121-145.

[34] Hemmata, A., Rahnamaa T., Vahabib Z., 2014. A horizontal multiple-tip penetrometer for on-the-go soil mechanical resistance and acoustic failure mode detection. Soil and Tillage Research. Vol 138, 17–25

[35] HENIN S., 1976. Cours de Physique du Sol. 2 vol. ORSTOM-Editest.

[36] Hénin S., Gras R., Monnier G., 1969 - Le profil cultural. Principes de physique du sol. Masson et Cie, Paris - 331.

[37] HILLEL, D., 1980. Stress-strain relations and soil strength. Fundamentals of soil physic, Academic Press Inc., University of Massachusetts. 318-354.

[38] Hummel, J.W., Ahmad, I.S., Newman, S.C., 2004. Simultaneous soil moisture and cone index measurement. ASAE Trans. 47 (3): 607-618.

[39] Isavi S., Mahmoudi A., 2013. Design, fabrication and evaluation of a mechanical transducer for real time measurement of tilth aggregate sizes. Agric Eng Int: CIGR Journal. 15(2): 130-137.

[40] Jahns, G., Speckmann, H., 1999. Development and application of an agricultural BUS for data transfer. Computers and Electronics in Agriculture 23(3): 219-237.

[41] Liu, W., S. K. Upadhyaya, T. Katakoa and S. Shibusawa. 1996. Development of a texture/soil compaction sensor. In proceedings of the third international conference on precision agriculture 617-630.

[42] Manuwa, S. and O.C. Ademosun. 2007. Draught and soil disturbance of model tillage tines under varying soil parameters. Agricultural Engineering International: the CIGR Ejournal Vol. IX (March): pp14.

[43] Mari, G.R., C. Ji, J. Zhou, and F.S. Bukhari. 2006. Effect of tillage machinery traffic on soil properties, corn root development and plant growth.Agricultural Engineering International: the CIGR Ejournal Vol. VIII (December): pp12.

[44] Merz, B. and A. Bárdossy. 1998. Effects of spatial variability on the rainfall runoff process in a small loess catchment. Journal of Hydrology 212-213304-317.

[45] Monnier G., Stengel P., Fies J. C., 1973. Une méthode de mesure de la densité apparente de petits agglomérats terreux. Application à l'analyse des systèmes de porosité du sol. Ann. agron., 24 (5), 533-545.

[46] Morrison Jr., J.E., Bartek, L.A., 1987. Design and field evaluation of a hand-pushed digital soil penetrometer with two cone materials. ASAE Trans. 30 (3) : 646-651.

[47] Mouazen, A. M., J. De Baerdemaeker and H. Ramon. 2005. Towards development of online soil moisture content sensor using a fibre-type NIR spectrophotometer. Soil and Tillage Research 80(1-2): 171-183.

[48] Norris, k. H. 1964. Reports on design and development of a new moisture sensor.Agriculture Engineering 45(7): 370-372.

[49] Ohu, J.O., G.S.V. Raghvan, and E. McKyes 1988. Cone index prediction of compacted soils. Transactions of the ASAE, 31: 306-310.

[50] Paetzold, R. F., G. A. Matzkanin and A. De Los Santos. 1985. Surface Soil Water Content Measurement Using Pulsed Nuclear Magnetic Resonance Techniques. soil science society of america journal 49(3): 537-540.

[51] Pauwels, V. R. N., R. Hoeben, N. E. C. Verhoest and F. P. De Troch. 2001. The importance of the spatial patterns of remotely sensed soil moisture in the improvement of discharge predictions for small-scale basins through data assimilation. Journal of Hydrology 251(1-2): 88-102.

[52] Reodique A., Schultz W., 2005. Noise Considerations for Integrated Pressure Sensors. Application Note. Freescale Semiconductor. AN1646, Rev 2.

[53] Rihani Md., 2012. A la découverte du sol vivant. Faculté des Sciences d'El Jadida (Maroc).

[54] Schlesinger, W. H., J. F. Reynolds, G. L. Cunningham, L. F. Huenneke, W. M. Jarrell, R. A. Virginia and W. G. Whitford. 1990. Biological feed-backs in global desertification. Science 247:1043-1048.

[55] Sirjacobs D., Hanquet B., Lebeau F., Destain M.-F. (2002). On-line mechanical resistance mapping and correlation with soil physical properties for precision agriculture. 64, 231-242.

[56] Stengel P., Gelin S., 1998. Sol interface fragile. INRA

[57] Sun Y., Lin J., Ma D., Zeng Q., Schulze Lammers P., 2007. Measurement of penetration force using a Hall-current-sensor. Soil & Tillage Research (92) 264-268.

[58] Sun Y., Ma D., Schulze Lammers P., Schmittmann O., Rose M., 2006. On-the-go measurement of soil water content and mechanical resistance by a combined horizontal penetrometer. Soil and Tillage Research. 86(2): 209-217.

[59] Sun Y., Schulze Lammers P., Ma D., Lin J., Zeng Q., 2008. Determining soil physical properties by multi-sensor technique. Sensors and Actuators 147: 352-357.

[60] Tekeste, M.Z., R.L. Raper, and E. Schwab. 2008. Soil Drying Effects on Soil Strength and Depth of Hardpan Layers as Determined from Cone Index Data. Agricultural Engineering International: the CIGR Ejournal Vol. X (December): pp17.

[61] Tessier, S., B. Lachance, C. Laguë, Y. Chen, L. Chi, and D. Bachand. 1997. Soil compaction reduction with a modified one-way disker. Soil & Tillage Research, 42: 63-77.

[62] Topakci M., Unal I., Canakci M., Celik H. K., Karayel D., 2010. Design of a Horizontal Penetrometer for Measuring On-the-Go Soil Resistance. Sensors. 10, 9337-9348.

[63] Topp, G. C. 1993. Soil water content. Boca Raton,Florida: Lewis publishers.

[64] Weihermüller, L., J. A. Huisman, S. Lambot, M. Herbst and H. Vereecken. 2007. Mapping the spatial variation of soil water content at the field scale with different ground penetrating radar techniques. Journal of Hydrology 340(3-4): 205-216.

[65] Weil Ray R., Brady Nyle C., 2008.The Nature and Properties of soils. Pearson Education.

[66] Wells L.G., Lewis C.O., Distler R.J., 1981. Remote electronic acquisition of soil cone index measurement. Journal of Terramechanics. 18(4): 201-207.

[67] Wells, L.G. and Treesuwan, O., 1977. The response of various soil strength indices to changing water content. ASAE paper No. 77-1055, St. Joseph, MI.

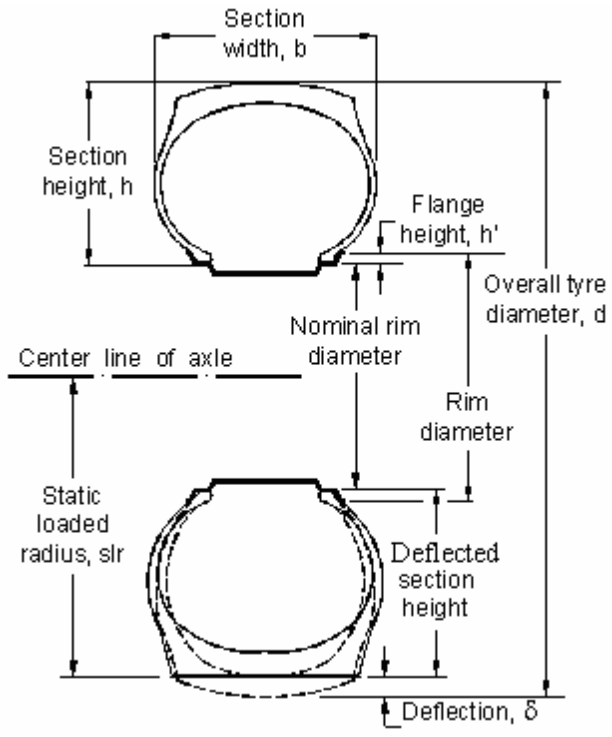
[68] Whalley, W. R., T. J. Dean and P. Izzard. 1992. Evaluation of the capacitance technique as a method for dynamically measuring soil water content. Journal of Agricultural Engineering Research 52:147-155.

[69] YORO G., GODO G., 1990. Les méthodes de mesure de la densité apparente. Cah. ORSTOM, sér. Pédol., 24 (4), 423-429.

[70] Zenga Q., Suna Y., Schulze Lammersb P., Maa D., Linc J., Huegingd H., 2008. Improvement of a dual-sensor horizontal penetrometer by incorporating an EC sensor. Computers and electronics in agriculture (64) 333-337.

[71] Zwaenepoel P., Le Bars J.M., 1997. L'agriculture de précision. Ingénieries – E A T, IRSTEA, 12: 67-79. <hal-00461080>.

Annexes A



Description of tire parameters (Brixius, 1987)
Description des paramètres d'un pneu

Vue d'une roue motrice pour tracteur agricole

Annexes B

Firmware des cartes annexes

Description du sketch

1. Lecture de la pression,
2. Lecture de la température,
3. Lecture de l'angle de rotation en continu et détection de l'angle 0° et 180° ,
4. Lecture de HC-SR04 (1) et HC-SR04 (2) en continu avec détection de la valeur de mesure des HC-SR04 (1) et HC-SR04 (2) a 0° et 180° ,
5. Envoi des données de mesures vers la carte principale via NRF24L01 et recommence à partir de 1.

Le sketch

```
/*

 Project name:  Soil and Tractor Prediction
 Code Name: doctorat
 Author: Lies BOUDHAR
 Copyright:  (1985-2018).
 Circuit: Carte annexe
 Date: 30/11/2017 at 01:30
 Last Revision:  23/05/2018 at 01:46
 - Initial release;
 Description:
 * Test configuration:
    MCU:           ATmega328P
    Dev. Board:     ARDUINO NANO
    Oscillator:     16 MHz
    Ext. Modules:  nRF24L01, mpx5700, GY 521, HC-SR04, DS18B20
    SW:            C/C++ ARDUINO
 * NOTES:

*/

#include <SoftwareSerial.h>                // Pour la communication serie
#include <Wire.h>                          // Pour la communication Wire
#include <OneWire.h>                       // Pour la communication OneWire
#include <DallasTemperature.h>             // Pour la gestion du capteur de temperature
#include <Mirf.h>                          // Pour la gestion de la communication du nRF24L01
#include <nRF24L01.h>                      // Pour les définitions des registres du nRF24L01
#include <MirfHardwareSpiDriver.h>         // Pour la communication SPI
```

```
#define ONE_WIRE_BUS D6                 // Sélection de l'entrée pour le capteur de température
OneWire oneWire(ONE_WIRE_BUS);
DallasTemperature sensors(&oneWire);    // Création d'une instance DallasTemperature
float temperature = 0;                  // Variable de stockage de la température

int mpx5700Pin = A0;                    // Sélection de l'entrée pour le capteur de pression
float pressureValue = 0;                // Variable de stockage de la pression
const float SensorOffset = 102.0;       //

#define trigPin_1       9               // Sélection de l'entrée TRIGGER pour HC-SR04(1) D5
#define echoPin_1       2               // Sélection de l'entrée ECHO pour HC-SR04(1) D4
#define trigPin_2       1               // Sélection de l'entrée TRIGGER pour HC-SR04(1) D3
#define echoPin_2       32              // Sélection de l'entrée ECHO pour HC-SR04(1) D2

const int MPU_addr=0x68;                // définition de l'adresse I2C du GY521
int16_t AcX,AcY,AcZ;                    // Variable de stockage pour accéléromètre
int16_t Tmp;                            // Variable de stockage pour température
int16_t GyX,GyY,GyZ;                    // Variable de stockage pour gyroscope
int minVal=265;                         //
int maxVal=402;                         //
double x;                               // Variable de stockage de l'abscisse
double y;                               // Variable de stockage de l'ordonnée
double z;                               // Variable de stockage de la cote

int N = 0 ;                             // Variable pour nombre de tour
int W = 0 ;                             // nombre de tour/seconde
int H0 = 0 ;                            //
int H180 = 0 ;                          //

byte Compteur = 0;                      // La variable compteur

void setup() {
   Serial.begin(9600);                  //Définit le débit pour la transmission de données en série
   Wire.begin();
   Wire.beginTransmission(MPU_addr);
   Wire.write(0x6B);
   Wire.write(0);
   Wire.endTransmission(true);

   sensors.begin();                     // Démarrage de la bibliothèque DallasTemperature

   Mirf.cePin = 9;                      // Broche CE sur D7 broche 9
   Mirf.csnPin = 10;                    // Broche CSN sur D8 broche 9
   Mirf.spi = &MirfHardwareSpi;         // On veut utiliser le port SPI hardware
   Mirf.init();                         // Initialise la bibliothèque
   Mirf.channel = 1;                    // Choix du canal de communication
   Mirf.payload = 32;                   // Taille d'un message (maximum 32 octets)
   Mirf.config();                       // Sauvegarde la configuration dans le module radio
```

```
    Mirf.setTADDR((byte *) "nrf02");          // Adresse de transmission
    Mirf.setRADDR((byte *) "nrf01");          // Adresse de réception

}

// Routine d'interruption pour vitesse de rotation
ISR(TIMER2_OVF_vect) {
    TCNT2 = 1024 - 976;                       // 976 x 64 µS = 62,464 ms
    if (Compteur++ > 16) {                    // 62,464 x 16 ms = 999,424 ms
      Compteur = 0;
      W = N;
      N = 0;
    }
}

void loop() {
    unsigned int one = 0;

    sensors.requestTemperatures();
    T = sensors.getTempCByIndex(0);

    pressureValue = (analogRead(mpx5700Pin)-SensorOffset)/100.0;          // en kPa

    read_angle();
    if(x<3){
      H0 = read_distance(T , trigPin_1, echoPin_1);
      H180 = read_distance(T , trigPin_2, echoPin_2);
    }
    if(x<1 | one == 0){
      N++;
      one = 1;
    }
    if(x>2){
      one = 0;
    }
    transmit_data();
}

//fonction de lecture des angles par rapport aux trois axes ox oy oz
void read_angle() {
    Wire.beginTransmission(MPU_addr);
    Wire.write(0x3B);
    Wire.endTransmission(false);
    Wire.requestFrom(MPU_addr,14,true);

    AcX=Wire.read()<<8|Wire.read();
    AcY=Wire.read()<<8|Wire.read();
    AcZ=Wire.read()<<8|Wire.read();
```

```
    int xAng = map(AcX,minVal,maxVal,-90,90);
    int yAng = map(AcY,minVal,maxVal,-90,90);
    int zAng = map(AcZ,minVal,maxVal,-90,90);

    x= RAD_TO_DEG *(atan2(-yAng, -zAng)+PI);
    y= RAD_TO_DEG *(atan2(-xAng, -zAng)+PI);
    z= RAD_TO_DEG *(atan2(-yAng, -xAng)+PI);

    if(x>180)x= RAD_TO_DEG *(atan2(-yAng, -zAng)-PI);
    if(y>180)y= RAD_TO_DEG *(atan2(-xAng, -zAng)-PI);
    if(z>180)z= RAD_TO_DEG *(atan2(-yAng, -xAng)-PI);
}

//fonction de lecture de la hauteur à l'intérieur du pneu
float read_distance(float T , int TRIGGER_PIN, int ECHO_PIN) {

    //Lance une mesure de distance en envoyant une impulsion HIGH de 10µs sur la broche TRIGGER
    digitalWrite(TRIGGER_PIN, HIGH);
    delayMicroseconds(10);
    digitalWrite(TRIGGER_PIN, LOW);

    //Mesure le temps (en µs) entre l'envoi de l'impulsion ultrasonique et son écho (s'il existe)
    long measure = pulseIn(ECHO_PIN, HIGH, 0);

    //Calcul la distance (en mm) à partir du temps mesuré
    float distance_mm = (331,35 + 0,607 * T)*measure/2000;
    return distance_mm;
}

//fonction de transmission des données vers la carte principale
void transmit_data(){

    int message[5];
    message [0] = int (pressureValue);          // pression en kPa
    message [1] = int (T);                      // température en C
    message [2] = int (H0);                     // H0 en mm
    message [3] = int (H180);                   // H180 en mm
    message [4] = int (W);                      // tr/s

    Mirf.send((byte*) &message);                // On envoi le message
    while(Mirf.isSending());                    // On attend la fin de l'envoi

}
```

Annexes C

Firmware de la carte principale

Description du sketch

1. Réception des données de mesures des cartes annexes via NRF24L01,
2. Lecture de l'angle de pente,
3. Lecture de la force de traction,
4. Lecture de la conductivité du sol,
5. Lecture de la date et de l'heure,
6. Affichage des résultats de mesure et de calcul,
7. Traitement des requêtes du clavier,
8. Traitement des requêtes de l'application Android,
9. Envoi des données de mesures et de calcul vers l'application Android via Bluetooth et recommence à partir de 1.

Le sketch

```
/*
 Project name:  Soil and Tractor Prediction
 Code name: doctorat
 Author: Lies BOUDHAR
 Copyright:  (1985-2018).
 Circuit: Carte principale
 Date: 28/11/2017 at 19:30
 Last Revision:  23/05/2018 at 02:46
 - Initial release;
 Description:
 * Test configuration:
    MCU:           ATmega2560
    Dev. Board:      ARDUINO MEGA
    Oscillator:     16 MHz
    Ext. Modules:  nRF24L01, GY 521, HC-05, HX711, ESP_ssd1306_128x64_SPI, keypad 4x4, RTC1307
    SW:           C/C++ ARDUINO
 * NOTES:
*/

#include <SoftwareSerial.h>                // Pour la communication série
#include <SPI.h>                           // Pour la communication via le port SPI
#include <Wire.h>                          // Pour la communication à fil (Wire)
#include <U8glib.h>                        // bibliothèque pour les TFT monochromes et les OLED
```

```
#include <Keypad.h>                    // bibliothèque pour clavier matriciel 4x4
#include <RTClib.h>                    // bibliothèque pour la gestion de l'horloge temps réel
#include <HX711.h>                     // bibliothèque pour la gestion du CAN HX711
#include <Mirf.h>                      // Pour la gestion de la communication
#include <nRF24L01.h>                  // Pour les définitions des registres du nRF24L01
#include <MirfHardwareSpiDriver.h>     // Pour la communication SPI

HX711 TractionCell(31, 30);            // HX711 constructor (dout pin, sck pin)

RTC_DS1307 RTC;                        // Classe RTC_DS1307

#define OLED_MOSI  11
#define OLED_CLK   13
#define OLED_DC    10
#define OLED_CS    12
#define OLED_RESET 9
U8GLIB_SSD1306_ADAFRUIT_128X64 Display(OLED_CLK, OLED_MOSI, OLED_CS, OLED_DC,
OLED_RESET);

long loop_timer;

const byte ROWS = 4; //four rows
const byte COLS = 4; //four columns
//define the symbols on the buttons of the keypads
char hexaKeys[ROWS][COLS] = {
 {'0','4','8','C'},
 {'1','5','9','D'},
 {'2','6','A','E'},
 {'3','7','B','F'}
};
byte rowPins[ROWS] = {A11, A10, A9, A8}; //connect to the row pinouts of the keypad
byte colPins[COLS] = {A15, A14, A13, A12}; //connect to the column pinouts of the keypad
//initialize an instance of class NewKeypad
Keypad Keypad0 = Keypad( makeKeymap(hexaKeys), rowPins, colPins, ROWS, COLS);

byte message[32]; //pour NRF24L01

const int MPU_addr=0x69;               // définition de l'adresse I2C du GY521
int16_t AcX,AcY,AcZ;                   // Variable de stockage pour accéléromètre
int16_t Tmp;                           // Variable de stockage pour température
int16_t GyX,GyY,GyZ;                   // Variable de stockage pour gyroscope
int minVal=265;                        //
int maxVal=402;                        //
double x;                              // Variable de stockage de l'angle/l'abscisse
double y;                              // Variable de stockage de l'angle/l'ordonnée
double z;                              // Variable de stockage de l'angle/la cote

int lcd_loop_counter;
int num, menu, X, Y, L;
```

```
float traction;
float angle;
float conductivity;
int humidite;
int Datas[8];

void setup() {
  Serial.begin(9600);                    //Définit le débit pour la transmission de données en série
  Wire.begin();
  Wire.beginTransmission(MPU_addr);
  Wire.write(0x6B);
  Wire.write(0);
  Wire.endTransmission(true);

  RTC.begin();                                          // Démarrage de la librairie RTClib.h
  RTC.adjust(DateTime(__DATE__, __TIME__));             // adjust to compile time

  Mirf.cePin = 9;                                       // Broche CE sur D7 broche 9
  Mirf.csnPin = 10;                                     // Broche CSN sur D8 broche 10
  Mirf.spi = &MirfHardwareSpi;                          // On veut utiliser le port SPI hardware
  Mirf.init();                                          // Initialise la bibliothèque
  Mirf.channel = 1;                                     // Choix du canal de communication
  Mirf.payload = 32;                                    // Taille d'un message (maximum 32 octets)
  Mirf.config();                                        // Sauvegarde configuration dans le module radio
  Mirf.setTADDR((byte *) "nrf02");                      // Adresse de transmission
  Mirf.setRADDR((byte *) "nrf01");                      // Adresse de réception

  TractionCell.set_scale(2280.f);                       //
  TractionCell.tare();                                  //

  num = 0;
  X = 33;
  Y = 10;
  L = 14;

  draw_accu();

  delay(5000);
  loop_timer = micros();
  lcd_loop_counter = 0;
}

void loop() {

  int message[5];
  if(Mirf.dataReady()){
    Mirf.getData((byte *) &message);                    // Réception du paquet
    Datas[0] = message [0];                             // pression en kPa
    Datas[1] = message [1];                             // temperature en C
```

```
  Datas[2] = message [2];                    // H0 en mm
  Datas[3] = message [3];                    // H180 en mm
  Datas[4] = message [4];                    // tr/s
 }

 read_angle();                               //Lecture de l'angle de pente
 Datas[5] = int (x);

 traction = TractionCell.get_units();        //Lecture de la force de traction
 Datas[6] = int (traction);

 conductivity = analogRead(A0);              // Lit la tension analogique
 humidite = (1023 - conductivity)*0.138;
 Datas[7] = int (humidite);

 DateTime now = RTC.now();
 sendAndroidValues();
 key_traitement();

 if(num > 1000){
  if(menu==0) draw_menu();
  if(menu==1) draw_control();
  if(menu==2) draw_set();
  if(menu==3) draw_accu();
  num = 0;
 }
 num++;
}

//fonction de lecture des angles par rapport aux trois axes ox oy oz
void read_angle() {
 Wire.beginTransmission(MPU_addr);
 Wire.write(0x3B);
 Wire.endTransmission(false);
 Wire.requestFrom(MPU_addr,14,true);
 AcX=Wire.read()<<8|Wire.read();
 AcY=Wire.read()<<8|Wire.read();
 AcZ=Wire.read()<<8|Wire.read();
 int xAng = map(AcX,minVal,maxVal,-90,90);
 int yAng = map(AcY,minVal,maxVal,-90,90);
 int zAng = map(AcZ,minVal,maxVal,-90,90);
 x= RAD_TO_DEG *(atan2(-yAng, -zAng)+PI);
 y= RAD_TO_DEG *(atan2(-xAng, -zAng)+PI);
 z= RAD_TO_DEG *(atan2(-yAng, -xAng)+PI);
 if(x>180)x= RAD_TO_DEG *(atan2(-yAng, -zAng)-PI);
 if(y>180)y= RAD_TO_DEG *(atan2(-xAng, -zAng)-PI);
 if(z>180)z= RAD_TO_DEG *(atan2(-yAng, -xAng)-PI);
}
```

```
void sendAndroidValues(){
   Serial.flush();
   for(int k=0; k<7; k++){
     Serial.print(Datas[k]);
     Serial.print('+');
   }
}

//traitement clavier
void key_traitement(){

  int KeyValue = Keypad0.getKey();
  if(KeyValue){
    if(KeyValue==0x30){   //DOWN
      if(menu==0){
        Y = Y+10;
        if(Y==70)Y = 10;
        draw_menu();
      }
    }
    if(KeyValue==0x31){   //UP
      if(menu==0){
        Y = Y-10;
        if(Y==0)Y = 60;
        draw_menu();
      }
    }
    if(KeyValue==0x32){   //OK
      if(Y==10) menu=1;
      if(Y==20) menu=2;
      if(Y==30) menu=3;
    }
    if(KeyValue==0x33){   //  MENU
      draw_menu();
    }
    if(KeyValue==0x20){   //
    }
    if(KeyValue==0x21){   //
    }
    if(KeyValue==0x22){   //
    }
    if(KeyValue==0x23){   //
    }
   }
}

// Ecran d'accueil
void draw_accu(){
  Display.firstPage();
```

```
  do {
    Display.setFont(u8g_font_unifont);
    Display.drawStr(1, 10, "BOUDHAR LIES");
    Display.drawStr(1, 25, "ENSA 04/2018");
    Display.drawStr(1, 40, "These doctorat");
    Display.setPrintPos(64, 1);
  } while( Display.nextPage() );
}

// Ecran MENU
void draw_menu(void) {
  Display.firstPage();
    do {
       Display.setFont(u8g_font_profont12);
       Display.drawStr( 0, 10, "MENU");
       Display.setFont(u8g_font_profont11);
       Display.drawStr( 35, 10, "Controle");
       Display.drawStr( 35, 20, "Reglages");
       Display.drawStr( 35, 30, "Ecran d'accueil");
       Display.drawHLine(X, Y, L);
       Display.drawHLine(X, Y+1, L);
       Display.drawVLine(X, Y-8, 10);
       Display.drawVLine(X-1, Y-8, 10);
    } while( Display.nextPage() );
    menu=0;
}

// Ecran Controle (affichage des resultats des mesures)
void draw_control(void) {
  Display.firstPage();
    do {

    } while( Display.nextPage() );
    menu=0;
}
// Ecran Reglages
void draw_set(void) {
  Display.firstPage();
    do {

    } while( Display.nextPage() );
    menu=0;
}
```

Annexes D

Programme de l'application Android

Description des programmes java de l'application Android

L'application Android réalise les fonctions suivantes :

1. Calcul du cône index,
2. Détermination de la texture du sol,
3. Prédiction des performances de traction,
4. Calcul du brixius number,
5. Prédiction de l'humidité du sol,
6. Réception des données de mesures de la carte principale,

Voici les programmes java des trois activités (MainActivity, OutputActivity, InputActivity);

Java pour MainActivity

```
package com.example.lboudhar.soiltractorprediction;

import android.support.v7.app.AppCompatActivity;
import android.content.Intent;
import android.view.View;
import android.widget.Button;
import android.os.Bundle;
import android.widget.Toast;

public class MainActivity extends AppCompatActivity {

    // constant to determine which sub-activity returns
    private static final int REQUEST_CODE = 10;

    Button Bluetooth_button, input_button, output_button, info_button;
    public float clay, sand, silt, b, h, d, di, X, Xf, Xr, Hc, Xp,Wt;
    public String texture = "";

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

        Bluetooth_button = findViewById(R.id.Bluetooth_button);
        Bluetooth_button.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                startBluetoothActivity();
            }
```

```
        });

        input_button = findViewById(R.id.input_button);
        input_button.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                startInputActivity();
            }
        });

        output_button = findViewById(R.id.output_button);
        output_button.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                startOutputActivity();
            }
        });

        info_button = findViewById(R.id.info_button);
        info_button.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                startInfoActivity();
            }
        });

    }

    public void startBluetoothActivity(){
        Intent intent = new Intent(MainActivity.this,Bluetooth.class);
        startActivity(intent);
    }

    public void startInputActivity(){
        Intent intent = new Intent(MainActivity.this,InputActivity.class);
        intent.putExtra("clay_init",clay);
        intent.putExtra("sand_init",sand);
        intent.putExtra("silt_init",silt);
        intent.putExtra("texture",texture);
        startActivityForResult(intent, REQUEST_CODE);
    }

    public void startOutputActivity(){
        Intent intent = new Intent(MainActivity.this,OutputActivity.class);
        intent.putExtra("texture",texture);
        startActivity(intent);
    }

    public void startInfoActivity(){
        Intent intent = new Intent(MainActivity.this,InfoActivity.class);
        startActivity(intent);
    }

    @Override
    protected void onActivityResult(int requestCode, int resultCode, Intent data) {
        if (resultCode == RESULT_OK && requestCode == REQUEST_CODE) {

            String result = data.getExtras().getString("clay_init");
            clay = Float.valueOf(result);
            result = data.getExtras().getString("sand_init");
            sand = Float.valueOf(result);
            result = data.getExtras().getString("silt_init");
            silt = Float.valueOf(result);
```

```
            texture = data.getExtras().getString("texture");
                if (result != null && result.length() > 0) {
                    Toast.makeText(this, result, Toast.LENGTH_SHORT).show();
                }
            }
        }
    }
```

Java pour OutputActivity

```
package com.example.lboudhar.soiltractorprediction;

import android.content.Intent;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.TextView;

public class OutputActivity extends AppCompatActivity {

    private TextView GTR_output, MRR_output, NTR_output, TE_output, Bn_output, CI_output, H_output;
    private TextView Texture, Structure_output, Slip_output, Pull_output, Wr_output, Va_output;
    private TextView Vr_output;
    private double  b, h, d, di, X, Xf, Xr, Hc, Xp,Wt;
    private double CI, H, Va, Vr, a, def, hp, pi, S, δ, Bn, Pull, Wr, Wf;
    private double a1, a2, a3, a4, a5, a6, K1, K2;
    private double A, C, U, Z;
    private int TyreType;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_output);

        GTR_output = findViewById(R.id.GTR_output);
        MRR_output = findViewById(R.id.MRR_output);
        NTR_output = findViewById(R.id.NTR_output);
        TE_output = findViewById(R.id.MRR_output);
        Bn_output = findViewById(R.id.Bn_output);
        CI_output = findViewById(R.id.CI_output);
        H_output = findViewById(R.id.H_output);
        Pull_output = findViewById(R.id.Pull_output);
        Wr_output = findViewById(R.id.Wr_output);
        Va_output = findViewById(R.id.Va_output);
        Vr_output = findViewById(R.id.Vr_output);

        Structure_output = findViewById(R.id.Structure_output);
        Slip_output = findViewById(R.id.Slip_output);

        TyreType = 1;
        Pull = 8;
        Wr = 36;
        Va = 3;
        Vr = 3.8;
        K1 = 5.0;
        K2 = 3.0;
        b = 0.25;
        d = 1.5;
        δ = 9.0;
        h = 45.0;
```

```
    Button Go_button = findViewById(R.id.Go_button);
    Texture = findViewById(R.id.Texture_output);
    Intent intent = getIntent();
    Texture.setText(intent.getStringExtra("texture"));

    Go_button.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View v) {
            if (TyreType == 1){     // diagonal
                a1=0.88; a2=0.1; a3=7.5; a4=0.04; a5=1.0; a6=0.5;
            }else{      // radial
                a1=0.88; a2=0.1; a3=9.5; a4=0.032; a5=0.9; a6=0.5;
            }
            S = 1 - Va/Vr;
            Slip_output.setText(String.format ("%.2f", S*100) + " %");
            Pull_output.setText(String.format ("%.2f", Pull) + " daN");
            Wr_output.setText(String.format ("%.2f", Wr) + " daN");
            Va_output.setText(String.format ("%.2f", Va) + " km/h");
            Vr_output.setText(String.format ("%.2f", Vr) + " km/h");
            Get_Bn_CI();
            Tractor_Performance();
        }
    });
}

public double BrixiusNum(double Bn) {
    double Y = Z / Math.exp(a2 * Bn) + (a5 / Bn) + (C / Math.sqrt(Bn)) + U - Z;
    return Y;
}

public double Iteration( double x, double epsilon, int n) {

    while( BrixiusNum(x) > 0) {
        x = x + epsilon;
        n = n + 1;
    }
    return x;
}

public void Get_Bn_CI() {
    double x = 1.0;
    int n = 0;
    C = a6 * S;
    A = 1 - Math.exp(-a3 * S);
    Z = a1 * A;
    U = Pull / Wr;

    x =Iteration(x, 1, 0);
    x =Iteration(x - 1, 0.1, n);
    x =Iteration(x - 0.1, 0.01, n);
    Bn = x - 0.01;
    CI=(Bn*Wr)/(b*d)*((1+K1*(δ/h))/(1+K2*(b/d)));
    Bn_output.setText(String.format ("%.2f", Bn));
    CI_output.setText(String.format ("%.2f", CI) + " kPas");
}

public void Tractor_Performance() {
    double GTR, MRR, NTR, TE;
    GTR = a1 * (1 - Math.exp(-a2 * Bn)) * (1 - Math.exp(-a3 * S)) + a4;
    MRR = a5 / Bn + a4 + (a6 * S) / Math.sqrt(Bn);
    NTR = GTR - MRR;
    TE = (NTR / GTR) * (1.0 - S);
```

```
        GTR_output.setText(String.format("%.2f", GTR));
        MRR_output.setText(String.format("%.2f", MRR));
        NTR_output.setText(String.format("%.2f", NTR));
        TE_output.setText(String.format("%.4f", TE));
    }
}
```

Java pour InputActivity

```
package com.example.lboudhar.soiltractorprediction;

import android.content.Intent;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import android.widget.TextView;

public class InputActivity extends AppCompatActivity {

    private TextView Texture;
    private EditText clay_input, sand_input, silt_input, b_Input, h_input, d_Input, di_Input;
    private EditText X_Input, Xf_Input, Xr_Input, Hc_Input, Xp_Input, Wt_input;

    public double clay, sand, silt, b, h, d, di, X, Xf, Xr, Hc, Xp,Wt;
    public String texture = "";

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_input);

        Texture = findViewById(R.id.Texture_output);
        clay_input = findViewById(R.id.Clay_input);
        sand_input = findViewById(R.id.Sand_input);
        silt_input = findViewById(R.id.Silt_input);
        b_Input = findViewById(R.id.b_Input);
        h_input = findViewById(R.id.h_input);
        d_Input = findViewById(R.id.d_Input);
        di_Input = findViewById(R.id.di_Input);
        X_Input = findViewById(R.id.X_Input);
        Xf_Input = findViewById(R.id.Xf_Input);
        Xr_Input = findViewById(R.id.Xr_Input);
        Hc_Input = findViewById(R.id.Hc_Input);
        Xp_Input = findViewById(R.id.Xp_Input);
        Wt_input = findViewById(R.id.Wt_input);

        Button Texture_button = findViewById(R.id.B_Texture);
        Button Clear_button = findViewById(R.id.B_Clear);

        Intent intent = getIntent();
        clay = intent.getFloatExtra("clay_init",0);
        sand = intent.getFloatExtra("sand_init",0);
        silt = intent.getFloatExtra("silt_init",0);
        clay_input.setText(Double.toString(clay));
        sand_input.setText(Double.toString(sand));
        silt_input.setText(Double.toString(silt));
        Texture.setText(intent.getStringExtra("texture"));

        Texture_button.setOnClickListener(new View.OnClickListener() {
```

```
        @Override
        public void onClick(View v) {
            GetTexture();
        }
    });

    Clear_button.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View v) {
            clay = sand = silt = b = h = d = di = X = Xf = X = Hc = Xp = Wt = 0.0;
            texture ="";
            String string = Float.toString((float)0.0);
            clay_input.setText(string);
            sand_input.setText(string);
            silt_input.setText(string);
            Texture.setText(texture);
            b_Input.setText(string);
            h_input.setText(string);
            d_Input.setText(string);
            di_Input.setText(string);
            X_Input.setText(string);
            Xf_Input.setText(string);
            Xr_Input.setText(string);
            Hc_Input.setText(string);
            Xp_Input.setText(string);
            Wt_input.setText(string);

        }
    });
}


@Override
public void finish() {
    Intent data  = new Intent();
    clay = Double.valueOf(clay_input.getText().toString());
    sand = Double.valueOf(sand_input.getText().toString());
    silt = Double.valueOf(silt_input.getText().toString());
    data .putExtra("clay_init", Double.toString(clay));
    data .putExtra("sand_init", Double.toString(sand));
    data .putExtra("silt_init", Double.toString(silt));
    data .putExtra("texture", Texture.getText().toString());
    setResult(RESULT_OK, data );
    super.finish();
}
}
```

ABSTRACT

The sensor system for the analysis of physicals and mechanicals properties of soil in real time and continuously is a set of electronic circuit made around the Arduino Mega 2560. It features an alphanumeric display, matrix keypad, a conditioning circuit for strain gauges, a micro SD adapter, a Bluetooth module, two RF Module 2.4Ghz, two pressure sensors, four distance sensors, an electrical soil conductivity sensor, two rotation speed sensors, a GPS. It communicates with the application installed in Android. The Android application manages all operations of calculations and decisions. The system embarks on a two-wheel drive tractor. The Android application for calculations and decisions uses mathematical equations developed by researchers that make relationship between the cone index, sliding, and the force of resistance to traction, wet soil, and texture and soil structure.

Key words: Precision agriculture, tractor, Soil sensors, Arduino, microcontroller, firmware, Android, soil strength, wheel slip.

Résumé

Le système de capteur pour l'analyse des propriétés physico-mécanique du sol en temps réel et en continu est un ensemble de circuits électroniques réalisé autour de la carte Arduino Mega 2560. Il comporte un afficheur alphanumérique, un clavier matriciel, un circuit de conditionnement pour jauges de contrainte, un adaptateur pour mémoire micro SD, un module Bluetooth, deux modules HF 2.4Ghz, deux capteurs de pression, quatre capteurs de distance, un capteur de conductivité électrique de sol, deux capteurs de vitesse de rotation, un GPS. Il communique avec un système Android où est installée l'application qui gère l'ensemble des opérations de calculs et de prise de décision. Le système est embarqué sur un tracteur à deux roues motrices. L'application Android pour les calculs et les prises de décision, utilise des équations mathématiques développées par des chercheurs qui mettent en relation l'indice de cône avec le glissement, la force de résistance à la traction, l'humide du sol, la texture et la structure du sol.

Mots clés : agriculture de précision, tracteur, capteurs de force, Arduino, microcontrôleur, firmware, Android, résistance du sol, glissement.

نبذة مختصرة

نظام استشعار لتحليل الخصائص الفيزيائية والميكانيكية للتربة في الوقت الحقيقي وباستمرار عبارة عن مجموعة من الدوائر الإلكترونية التي صنعت حول Arduino Mega 2560 . وهي تتميز بشاشة أبجدية رقمية، لوحة مفاتيح مصفوفة، دائرة تكييف لمقاييس الجهد، و micro SD محول، وحدة بلوتوث، واثنين من الترددات اللاسلكية وحدة 2.4 غيغاهرتز، واثنين من أجهزة استشعار الضغط، وأربعة أجهزة استشعار عن بعد، وجهاز استشعار توصيلية كهربائية كهربائية، واثنين من أجهزة الاستشعار سرعة الدوران، ونظام تحديد المواقع. يتواصل مع التطبيق المثبت في Android . يدير تطبيق Android جميع عمليات الحسابات والقرارات. يبدأ النظام على جرار ذات دفعتين. يستخدم تطبيق Android للحسابات والقرارات المعادلات الرياضية التي طورها الباحثون التي تربط العلاقة بين مؤشر المخروط والانزلاق وقوة مقاومة الجر والتربة الرطبة والملمس وهيكل التربة.

الكلمات المفتاحية: الزراعة الدقيقة، الجرار، أجهزة استشعار التربة، المتحكم الدقيق، البرامج الثابتة، قوة التربة، زلة العجلات Arduino, Android.