

الجمهورية الجزائرية الديمقراطية الشعبية
République Algérienne démocratique et populaire

وزارة التعليم العالي و البحث العلمي
Ministère de l'enseignement supérieur et de la recherche scientifique

جامعة سعد دحلب البليدة
Université SAAD DAHLAB de BLIDA

كلية التكنولوجيا
Faculté de Technologie

قسم الإلكترونيك
Département d'Électronique



Mémoire de Projet de Fin d'Études

présenté par

Djebbari abd elkader

&

Bellache okba

MÉMOIRE DE Projet de fin d'étude Pour l'obtention du diplôme de master en électronique

option : Signaux en Ingénierie des Systèmes et Informatique Industrielle

Thème

Réalisation d'une station météo à basé d'une carte ARDUINO-UNO

Proposé par : Mr .Benselama Zoubir

Année Universitaire 2013-2014

Remerciements

Nous tenons à remercier tout d'abord DIEU tout puissant qui nous a donné durant toutes ces années la santé, le courage et la foi en nous-mêmes pour arriver à ce niveau là.

Nous exprimons nos sincères remerciements à l'égard de **Mr .BENSELAMA** pour avoir accepté d'encadrer ce travail pour leur aide et leurs précieux encouragements pour la mise en œuvre de ce modeste travail.

Notre infinie gratitude va à tous nos enseignants qui ont contribué à notre travail et à tous les enseignants de département d'électronique et en précisant les jurys...

Un grand merci à tous les amis.

Finalement, nous remercions tous ceux qui ont participé de près ou de loin à l'achèvement de ce travail.

ملخص:

في مشروعنا هذا قمنا بـ جازِ محطة للاحوال الجوية بالاعتماد على بطاقة

مكرومعالج " أردوينو".

حيث تسمح بمتابعة و تخزين تغيرات الاحوال الجوية(حرارة- رطوبة – اضاءة- سرعة الرياح واتجاهها)

ومن جهة اخرى اظهر هذه المعطيات على كاشف.

كما إن هذا المشروع موجه إلى تسيير نظام ذكي للتكيف الا صناعي لبيت بلاستيكي.

كلمات المفاتيح: " أردوينو" – حرارة- رطوبة – اضاءة- سرعة الرياح واتجاهها

Résumé :

Dans notre projet nous avons conçu une station météo a base d'une carte arduino , afin de suivre et d'acquérir le changement des données climatiques(température, luminosité, humidité, vitesse de vent ,direction de vent) d'une part, d'afficher ces données a l'aide d'un afficheur LCD et sauvegarder ces variation dans autre logiciel d'autre part. Ce travail sera orienté pour la gestion du climat d'une serre agricole afin de diminuer le cout de son contrôle en énergie.

Mots clés : arduino ; température ; luminosité ; humidité ; vitesse de vent ; direction de vent

Abstract :

In our project we conceived a station weather report has basis of a card "arduino" in order to follow and to acquire the climatic data change (temperature, brightness, humidity, speed of wind, direction of wind) on the one hand, to display these data with the help of an afficheur LCD and to protect this variation on the other hand on our software. this work will be oriented for the management of the climate of an agricultural greenhouse in order to decrease the cost of his control in energy

Keywords : arduino ; temperature; brightness; humidity; speed of wind; direction of wind

Table des matières

Introduction générale.....	1
Chapitre I : généralités sur les stations météo et les serres	
I.1 introduction.....	3
I.2 météorologie.....	3
I.2.1 historique.....	3
I.2.2 les outils de la météorologie.....	4
I.2.3 processeur de mesure.....	5
I.2.3.1 CAPTEUR.....	5
I.2.3.2 conditionnement	6
I.2.4 les capteurs utilise dans les stations météo.....	6
I.2.4.1 capteur de vitesse de vent.....	6
I.2.4.1.1 l'anémomètre numérique à coupelle ou à hélice.....	7
I.2.4.1.2 le tachymètre.....	8
I.2.4.2 capteur de direction de vent.....	9
I.2.4.2.1 comment déterminé la direction de vent.....	9
I.2.4.2.2 Codeurs absolus	9
I.2.4.3 Thermomètre	10
I.2.4.4 Baromètre électronique	12
I.2.4.5 pluviomètre numérique	13
I.2.4.6 capteur de l'humidité	13
I.2.4.6.1 Définition d'humidité	14
I.2.4.6.2 principales définition relatives a l'air humide	14
I.3 Généralité sur les serres	15

I.3.1 l'emplacement des serres	15
I.3.2 types de serres	15
Conclusion	17

Chapitre II : la carte arduino - uno

II.1 Introduction	18
II.2 Choix de la carte électronique « arduino »	19
II.3. Description de la carte « arduino-uno »	20
II.3.1 Synthèse des caractéristiques de la carte arduino uno	22
II.3.2 Brochage de la carte arduino-Uno	22
II.3.3 Alimentation	23
II.3.4 Mémoire de la carte arduino-uno	24
II.3.5 Entrées et sorties numériques	25
II.3.6 Broches analogiques	26
II.3.7 Autres broches de la carte arduino-uno	28
II.3.8 Communication de la carte arduino	28
II.3.9 Programmation	29
II.3.10 Réinitialisation (logicielle) automatique	30
II .3.11 Schéma structurel de la carte ARDUINO UNO	31
II.4 le Microcontrôleur ATMEL ATmega328p	32
Conclusion	33

Chapitre III : Etude et Réalisation du système

III.1 Introduction.....	34
III.2 Présentation Générale de l'ensemble.....	34
III.3 Schéma synoptique du système	35
III.4 Etude des capteurs	36
III.4.1 Capteur de la vitesse du vent.....	36
III.4.1.1 circuit de mise en forme	37
III.4.2 Capteur de la direction du vent	38
III.4.3 Capteur de température LM35DZ	41
III.4.3.1 caractéristiques du capteur de température LM 35 DZ	41
III.4.3.2 Calcul des caractéristiques du montage pour une variation pleine échelle du CAN	41
III.4.3.3 Conversion la tension Vout du LM35 en température avec l'arduino	43
III.4.4 capteur de la lumière	44
III.4.4.1 caractéristique	45
III.4.4.2 Linéarisation	45
III.4.5 Capteur de l'humidité	47
III .4.5.1 Caractéristiques métrologique	47
III.5. La carte arduino-uno.....	48
III.6 Afficheur a cristaux liquides (LCD).....	50
III.6.1 principe de fonctionnement	50
III.6.2 Brochage des afficheurs LCD	51
III .7 Le branchement de l'écran LCD avec l'arduino	54
III .7.1 Le démarrage de l'écran LCD avec Arduino	55
III .7.2 Ecriture de texte et Gérer le curseur sur l'écran LCD avec l'arduino	56
III .6.2.1 Organisé le repère de l'écran LCD avec l'arduino	56

III.8 La communication entre le PC et la cartes Arduino	57
III.8.1 Principe du montage	57
III.7.2 Programmation du montage	58
III.9 Carte de puissance	60
Conclusion	61

Chapitre IV : partie logiciel et simulation

IV.1 Introduction	62
IV.2 organigramme général du programme principal	62
IV.2.2 organigramme de sous programme de la vitesse	65
IV.3 le langage de programmation utilisée	66
IV.3.1 Présentation de l'Espace de développement Intégré (EDI) Arduino	66
IV.3.1.1 Description de l'interface	66
IV.3.1.2 Description de la barre des boutons	68
IV.3.1.3 Description de la structure d'un programme	69
IV.3.1.3.2 Description détaillée des parties	70
IV.3.1.3.3 Compilation et programmation de l'ARDUINO	71
IV.3.2 Guide matlab	73
IV.3.2.1 Ajout de composants à la zone Mise en page	74
IV.3.3.1 le programme de l'affichage	75
IV.3.3.2 organigramme général de gestion (logiciel)	76
IV.3.3.3 Environnement de logiciel	77
IV.3.3.4 Présentation du logiciel	77
Conclusion	79
Conclusion générale	80

Annexes

Bibliographie

Conclusion générale

L'objectif de notre projet qui consiste à capter, afficher et sauvegarder sous forme des tableaux les données climatiques (température, luminosité, vitesse de vent, direction de vent et l'humidité), et de commander automatiquement la gestion climatique d'une serre agricole a été atteint.

De plus grâce à l'outil de base de données du logiciel réalisé permet l'archivage des variations de ces paramètres et de suivre l'état du climat de l'environnement. Le logiciel réalisé, est souple, pratique et conviviale vue qu'il offre en premier lieu la possibilité de changer les consignes de la commande pour chaque position ou type de la serre, en second lieu, n'importe qu'elle operateur n'ayant aucune connaissance de l'outil informatique pourra consulter les données climatiques ou configurer la commande automatiquement.

Ce modeste travail que nous avons élaboré, nous a permis d'une part d'avoir une idée sur le résultat que peut donner l'association de l'outil informatique et de l'électronique, et d'autre part la possibilité offerte à un ordinateur de communiquer avec des périphérique extérieur via une liaison série qui est le USB que nous avons choisie pour notre réalisation.

Grâce à ce projet on a compris la philosophie d'une commande ou autrement dit l'automatisation d'un processus. Il nous a permis également de nos familiariser avec l'utilisation de multiples langages de programmation, tels que le MATLAB GUIDE et logiciel Arduino, ainsi que l'utilisation d'une communication série entre le PC et une carte Arduino à base d'un microcontrôleur. La carte que utilisé, représenté un facteur très important pour l'adapter à d'autres applications, pour cela, il suffit de la chargée avec un programme adéquat correspond a l'application désiré.

Enfin, nous espérons que ce travail que nous a permis d'élargir nos connaissances, servira de base pour mieux la développée et l'enrichir dans de prochains projet.

Liste des figures

Fig. I.2.3.1 Anémomètre	7
Fig. I.2.4.1.2 Principe du tachymètre optique.....	8
Fig.-1.2.4.2 La girouette.....	9
Fig. II.2 carte arduino –uno.....	19
Fig. II.3 Description de la carte « arduino-uno »	21
Fig. II.3.3 Alimentation	23
Fig. II.3.5 broches numériques de la carte arduino-uno.....	25
Fig. II.3.6 les Broches analogiques de la carte arduino-uno	26
Fig. II.3.6.B Le principe de la conversion CAN 3 bits.....	27
Fig. II.3.11 : schéma structurel de la carte ARDUINO UNO	31
Fig. II.4 Schéma bloc de description du microcontrôleur AVR.....	32
FIG III.2 Présentation Générale de l'ensemble	34
Fig.III.3 Schéma synoptique du système	35
Fig.III.4.1 Anémomètre à 4 coupelles.....	36
Fig-III.4.1.1 montage de la mise en forme du signal de sortie du capteur.....	37
Fig.III.4.2.a disque de codage.....	38
Fig. III.3.2.b Capteur de couleur CNY70.....	39
Fig.III.4.2.c Fonctionnement du capteur CNY70.....	40
Fig.III.4.2.c Circuit de conditionnement de la girouette.....	40
fig.III.4.3.2 Le montage du capteur de température LM35DZ avec l'arduino.....	42
Fig.III.4.4 LDR	44
Fig.-III.4.4.1 Circuit de conditionnement du LDR.	45
Fig.III.4.4.2 variation de la tension de sortie du capteur en fonction de luminosité.....	46
Fig. III.4.5. à le composant RTC 6919001.....	47
Fig.III.4.5.b : caractéristique de transfert du capteur d'humidité.....	48

Fig.III.6.1 schéma fonctionnel d'un afficheur LCD	50
Fig.III.7 Le montage à 4 broches de données	54
fig.III.9 la chaine de commande de carte de puissance.....	60
Org.IV.2 organigramme général du programme principal	65
Fig. IV.3.1.1.a: présentation des éléments de l'ARDUINO software.....	67
FIG. IV.3.2.a Fenêtre principale du GUIDE.....	73
Org.IV.3.3.2 organigramme général de la gestion de logiciel	76
Fig. IV.3.3.4.A interface principale.....	77

Liste des tableaux

Tab. II.1 caractéristiques des différentes carte « arduino ».....	19
Tab.II.3.1 Synthèse des caractéristiques de la carte arduino uno.....	22

Introduction Générale

L'introduction de la microinformatique dans le domaine industrielle s'avère indispensable cela est dus d'une part à la disponibilité de moyen informatique très performant a un prix relativement très réduit et la nécessité d'améliorer les méthodes de mesure artisanale pour les remplacer par d'autres techniques plus modernes et plus précises. A l'époque , la météorologie utilisait des capteurs chimiques ou mécaniques ayant une mauvaise précision avec un cout très élevé, mais aujourd'hui l'utilisation des capteurs purement électronique associé à des carte de traitement nous donne des mesures ayant une très grande précision .

C'est avec ces avantages très intéressante, que s'insère notre travail qui s'intitule <<Réalisation d'une station météo à Base d'une carte Arduino >>.Il consiste surtout à utiliser un microcontrôleur pour traiter les données venants de capteurs en utilisant son convertisseur A/N, ces bibliothèques associée à ces fonctions d'affichage sur un afficheurs à cristaux liquide , et la transmission des données vers un pc, et ceci pour rendre l'application pratique et convivial.

Ce projet aura comme tache la visualisation de données (vitesse de vent ,direction de vent ,température, humidité, luminosité) sur un afficheur LCD, à travers une interface sur un PC et de les sauvegarder dans une base des données sous formes de tableaux et graphes. Cette acquisition de l'état du climat sera utilisé pour différend situation que ce soit pour la gestion d'une serre agricole pour diminuer aux maximum le cout en énergie et sa protection, pour la prise de décision de l'autorisation des vol aérien etc.....

Pour présenter au mieux et facilite la compréhension de notre travail on a suivi la structure suivante :

.Le premier chapitre est consacré pour de Généralités sur les station météo et les serres

.Le deuxième chapitre réalise un rappel sur la carte Arduino se basant sur le convertisseur A/N ,compteur d'impulsion et la transmission série.

.Le troisième chapitre et l'étude et réalisation du système réalisation pratique de la maquette à Arduino avec un afficheur « LCD ».

Introduction Générale

.Le quatrième chapitre est la partie logicielle donnant les différents organigrammes qui nous permettra ensuite la réalisation de notre application, avec une explication de la manière d'utilisation du logiciel.

Chapitre I Généralité sur les stations météo et les serres

I.1 Introduction :

Les observations attentives des plants, des animaux ou plus simplement de la couleur d'un ciel couchant ont longtemps fait office de prévisions *météorologiques*. Pendant des siècles, même à l'échelle locale, la prévision du temps fut aléatoire. On ne pouvait que prier ou s'en remettre à des dictionnaires rassurants, fondés sur l'observation. De nos jours encore, les gens proches de la terre savent, leur solide bon sens prédire une pluie prochaine ou un beau temps persistant. Il n'y a qu'à songer à la multitude des dictons qui ont traversé les régions et les temps, jusqu'à devenir parfois véritables proverbes.

I.2 Météorologie :

La météorologie est une science qui étudie l'atmosphère terrestre. Elle a pour objet d'en connaître les états pour comprendre les phénomènes qui s'y déroulent afin de décrire le temps qu'il fait et de le prédire. La météorologie observe et étudie les 30 premiers kilomètres de l'atmosphère en contact avec la surface de la terre : la troposphère et la stratosphère inférieure.

I.2.1 Historique [15]:

La météorologie est une science très récente. Certes, les savants de la Grèce antique se montrent très intéressés par l'atmosphère et, au **IV^e** siècle av. J-C., Aristote rédige un traité intitulé « Etude des éléments de l'air », Environ un tiers de l'ouvrage est consacré aux phénomènes atmosphériques et c'est d'après cette œuvre que le terme moderne de météorologie a été forgé, cependant la météorologie progresse peu jusqu'à l'époque moderne.

Les premières observations scientifiques (au sol) ont lieu à partir du milieu du **XVII^e siècle** quand sont inventés les instruments de mesure indispensables : le thermomètre de Galilée (1641), le pluviomètre de Castelli (1639), le baromètre de Torricelli (1643), l'anémomètre et le premier hygromètre de Hooke (1667) en même temps que progresse la connaissance

Chapitre I Généralité sur les stations météo et les serres

des lois physiques des gaz et de la mécanique des fluides. En 1646, pascal répète l'expérience de Torricelli, qu'il

Complète en 1648 en confirmant la pesanteur de l'air. Au XVIIIe siècle, Hadley démontre l'effet de la rotation de la terre sur la direction des vents, Lavoisier découvre la composition de l'air et, le 1^{er} décembre 1783, le premier ballon explore l'atmosphère jusqu'à 3 400 m d'altitude. C'est à la veille du XXe siècle (1899) que trois ballons-sondes lancés depuis Trappes atteignant 1300 m et

Permettant d'identifier la stratosphère. Un trentaine d'années plus tard, les ballons sont équipés d'un émetteur transmettant les mesures (température, pression, humidité de l'air) au fur et à mesure de l'ascension de la radiosonde. Depuis les années cinquante, le perfectionnement des instruments de base et L'invention de nouveaux moyens d'investigation (radar, avions, fusée, satellites artificiels)

Ont permis d'acquérir une connaissance de plus précise et étendue des phénomènes atmosphériques.

I.2.2 Les outils de la météorologie :

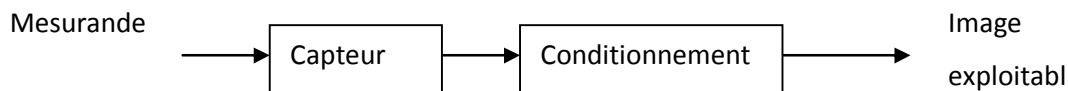
Les observations les plus répandues s'effectuent dans des stations météorologiques. L'équipement de base est l'abri météorologique installé dans un lieu dégagé. Placé à 2 m au-dessus du sol, il renferme des instruments de mesure homologués (capteur), protégés des rayons solaires et de l'agitation du vent ; Ce sont les **thermomètres à minima et à maxima**, qui marquent la température la plus basse et la plus élevée de la journée ; le **thermographe**, que enregistre les variations de la température au fil des heures, des jours, des semaines, **l'évaporomètre**, qui mesure la hauteur d'eau évaporée ; l'hygromètre ou psychromètre qui, par la température de l'air sec et de l'air humidité, indique l'humidité de l'air, tandis que **l'hygrographe enregistre les variations du taux d'humidité**; le **barographe** ou **baromètre** enregistreur signale les variations de la pression. A proximité de l'abri météorologique se trouvent un **pluviomètre** et un **pluviographe** qui mesurent les hauteurs d'eau précipitées, une girouette qui indique la direction du vent, un anémomètre qui en mesure la vitesse et un héliographe qui enregistre la durée d'insolation quotidienne. La nébulosité, exprimée en octas ou huitièmes de ciel couvert, doit être appréciée par l'observateur au moment des relevés. Sur les océans des navires

Chapitre I Généralité sur les stations météo et les serres

Météorologiques et des bouées équipées de station automatiques fournissent des mesures équivalentes à celles des abris à terre. Il existe dans le monde 9 000 stations météorologiques terrestres et 5 000 sur les océans.

I.2.3 Processeur de mesure [13]:

Un processus physique de mesure est généralement représenté par le schéma Qui suit :



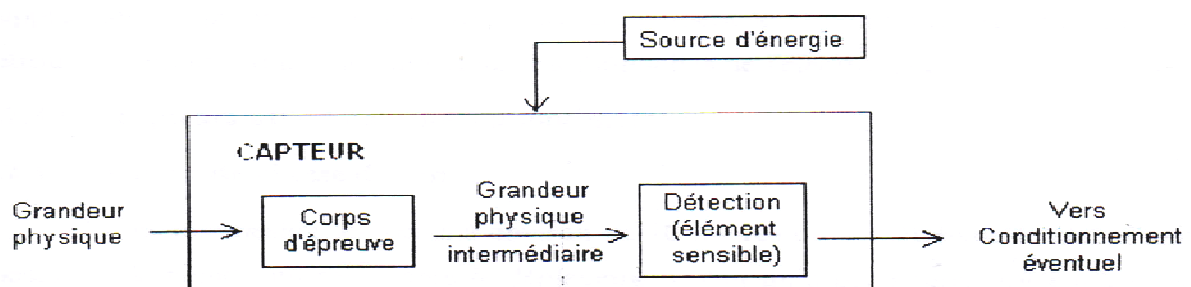
Les éléments essentiels de ce processus sont, le capteur, le circuit de conditionnement suivi d'un montage de traitement.

I.2.3.1 CAPTEUR :

Le capteur est un sous-ensemble (électronique, mécanique, optique...) qui traduit la mesurande (la grandeur objet de la mesure) en un signal électrique « brut ».son schéma fonctionnel est présenté comme suit :



La plupart des capteurs simples peuvent se décomposer selon le schéma suivant :



- **Corps d'épreuve** : c'est l'élément influencé par la mesurande qui est transformé en une grandeur physique intermédiaire interprétable par l'élément sensible du capteur : par exemple pour un anémomètre, il s'agit d'un disque entraîné par hélice et dont les trous placés en périphérie « défilent » sous une source de lumière éclairant un capteur optique....

Chapitre I Généralité sur les stations météo et les serres

- **DETECTEUR OU ELEMENT SENSIBLE** : il convertit la grandeur physique intermédiaire en une grandeur physique exploitable électriquement que nous appelons « SIGNAL » l'élément sensible est directement dépendant de la technologie de fabrication du capteur . A lui seul il peut constituer un « capteur »
- **SOURCE D'ENERGIE** : selon leur origine en source d'énergie électrique, on distingue :
 - a- **LES CAPTEUR ACTIFS** : ils délivrent un signal électrique (tension, courant, champ électrique..) sans nécessiter de source d'énergie externe. Dans ce cas c'est l'énergie propre au mesurande à prélever (énergie thermique, mécanique ou de rayonnement) qui est convertie en énergie électrique : le capteur se comporte donc comme un générateur.
 - b- **LES CAPTEURS PASSIFS** : il s'agit généralement d'impédance (résistance, condensateur, inductance...) dont l'un des paramètres déterminants est sensible à la grandeur mesurée.

I.2.3.2 CONDITIONNEMENT :

Le signal précédent n'est pas toujours exploitable directement par le système de traitement auquel il est destiné. L'ensemble des opérations d'adaptation (compatibilité, suppression de parasites ou amplification) constituent le conditionnement du signal. On y retrouve à peu près toutes les fonctions traditionnelles de l'électronique.

I.2.4 Les capteur utilisé dans les stations météo :

I.2.4.1 capteur de vitesse de vent :

La météo mesure les vents à environ de 10 mètres du sol. habituellement, l'unité de mesure est le nœud, qui représente le déploiement d'un mile par heure, soit environ 1.852 km/h . Cette unité reste en vigueur de nos jours, car c'est exactement la distance qui sépare deux points de même longitude et éloignés par une minute d'arc en latitude. Les marins apprécient la force du vent à l'aide de l'échelle BEAUFORT, du nom d'un amiral du siècle dernier. Cette mesure s'échelonne de 0 pour une mer calme à 12 pour un ouragan. Sur terre, il suffit de regarder les feuilles des arbres pour estimer cette valeur. Il nous sera bien utile pour mener à bien l'étalonnage des capteurs (anémomètres), appareils de mesure de la vitesse.

Chapitre I Généralité sur les stations météo et les serres

Le choix d'un type de capteur adapté est lié à la nature de la pièce tournante et des repères qu'elle porte, on utilise selon les cas soit l'un parmi les divers types :

I.2.4.1.1 l'anémomètre numérique à coupelles ou à hélice :

Ce type d'anémomètre désigné aussi comme moulinet, comprend un corps d'épreuve formé des coupelles ou d'une hélice qui est mise en rotation par des vents en mouvement (Fig. I.2.3.1).

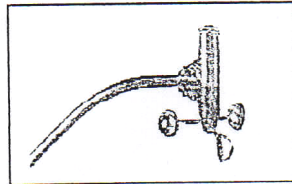


Fig-I.2.3.1 Anémomètre

La vitesse de rotation, mesurée par un dispositif « tachymétrique » approprié est proportionnelle à la vitesse du vent.

a)-LES COUPELLES :

Les coupelles, demi-sphères creuses, sont au nombre de 3 ou 4 selon les modèles, chacune est fixée, à l'extrémité d'un bras porté par un axe. L'ensemble peut être ou non caréné, le carénage diminuant la sensibilité de l'anémomètre aux inhomogénéités spatiales de l'écoulement.

b)-L'HELICE :

On peut remplacer les coupelles par des hélices (anémomètre Gill), utilisé en météorologie, comporte 3 axes perpendiculaires portant chacun un ensemble de coupelles il permet la détermination vectorielle de la vitesse de vent.

L'axe de l'anémomètre à hélice est placé parallèlement à la vitesse de l'écoulement. Dans le cas où on néglige les frottements, on peut estimer que l'hélice se visse, en quelque sorte, dans l'écoulement ; on aurait alors une relation de la forme :

$$U = h \cdot N \quad ((\text{EQUATION 1}))$$

U : étant la vitesse de l'écoulement, **N** : le nombre de tours par seconde de l'hélice et **h** : une constante.

I.2.4.1.2 le tachymètre :

Sous sa forme la plus simple, il comprend une source lumineuse et un détecteur optique : photodiode ou phototransistor. La pièce en rotation est munie de repères réfléchissants régulièrement espacés sur une circonférence et vers lesquels est dirigé le faisceau lumineux, est associée à un disque, à secteurs alternativement translucides et opaque, placé entre source

Et détecteur (FIG- I.2.4.1.2).

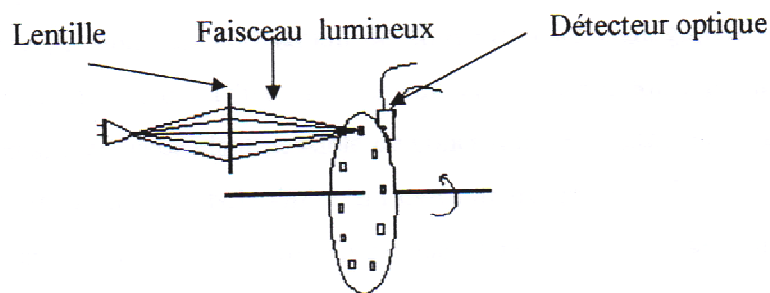


FIG- I.2.4.1.2 Principe du tachymètre optique

Le détecteur recevant un flux modulé la discontinuité de réflexion ou de transmission, délivre un signal électrique de fréquence proportionnelle à la vitesse de rotation et d'amplitude indépendante de cette vitesse.

La gamme de vitesses mesurables dépend, d'un part, du nombre de discontinuités optiques (repères, fentes, secteurs translucides) portées par le disque ou la pièce en rotation, d'autre part. De la bande passant du détecteur et des circuits électriques associés.

Pour la mesure de faibles vitesses, par exemple 0.1 tr/mn, on utilise de disque à très grand nombre de fentes (de 500 à quelques 10^3), pour la mesure de vitesses élevées, par exemple 10^5

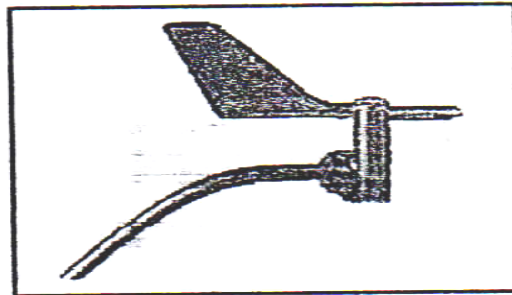
à 10^6 tr/mn dans le cas d'ultracentrifugeuses, le disque ne porte qu'une seule fente et c'est la fréquence de coupure du circuit électrique qui détermine la vitesse maximale mesurable .

L'utilisation d'un disque à deux pistes décalées d'un quart de période spatial (générateur incrémental optique), permet la détermination du sens de rotation.

I.2.4.2 Capteur de direction de vent :

La direction du vent est traditionnellement Représentée sur la rose des vents, selon 4,8 ou même 16 points cardinaux, (l'ouest et souvent noté W).

La girouette (**Fig.-I.2.4.2**). Nommée devine-vent dans le sud, indique par une flèche ou le bec du coq sur un clocher.



(Fig.-1.2.4.2) La girouette

I.2.4.2.1 Comment déterminer la direction de vent [3] :

Les informations, acquises par les capteurs et délivrées sous forme analogique, doivent très fréquemment être traitées par des calculateurs digitaux qui en assurent l'exploitation rapide et précise.

Ceci est rendu possible par l'utilisation de convertisseurs analogique_ numérique qui recevant le signal de capteur, délivrent au calculateur l'information digitalisée. Dans le cas des déplacements, il est cependant possible de concevoir des capteurs qui assurent de façon immédiate la traduction d'une position en un mot binaire qui la définit. Cependant, comme dans tout dispositif de Conversion numérique d'une grandeur, celle-ci se trouve quantifiée. Un nombre limité de position peut être distingué et la résolution est donc toujours finie

I.2.4.2.2 Codeurs absolus :

C'est, un disque qui est divisé en N secteur égales à l'intérieur desquelles se trouve Matérialisé le mot binaire associé à la position à traduire, selon un code et une technologie déterminé.

Le nombre N de secteur fixe la résolution soit de $360^\circ/N$.

Chapitre I Généralité sur les stations météo et les serres

Les N bits constituant chacun des mots sont matérialisés sur N pistes. En utilisant des états physiques complémentaires pour distinguer la valeur 0 ou 1 :

- secteurs magnétique ou ferromagnétique (lecteur magnétique).
- secteurs isolants ou conducteurs (lecture par courant).
- secteurs opaque ou translucide (lecture optique).

La lecture optique est actuellement le procédé le plus employé : il y a pour chacune des pistes une source qui est une diode électroluminescent et récepteur qui est un phototransistor.

Si les dispositifs de lecture ne sont pas parfaitement alignés. Les changements de bits ne sont pas lus simultanément et il y a un risque d'erreur lorsque la lecture s'effectue pendant la transition, ou si le codeur s'arrête dans une position à la limite de deux valeurs. Ce risque de lecture erronée, qui existe aussi pour le code « BCD », peut être évité soit en

Utilisant : -un code dont un seul bit change lors de chaque déplacement élémentaire, en général un code réfléchi.

-un code binaire naturel ou « BCD » mais en employant un dispositif de lecture supplémentaire permettant d'éviter les lectures ambiguës dans les zones de transition (codeurs à lecture commandée, à double balayage).

Les codes les plus couramment utilisés sont le code GRAY et le code « BCD » réfléchi à partir de ceux-ci, on peut définir de nouveaux codes réfléchis, dits par excès 3, en décalant de 3 unités vers les valeurs supérieures les représentations primitives, ces codes facilitent certaines opérations arithmétique.

I.2.4.3 Thermomètre [1] :

Il est intéressant de connaître les températures minimales et maximales observées au cours de la journée. Le minimum est souvent mesuré au petit matin, alors que le maximum se produit généralement à la fin de l'après-midi. Pour cela on utilise un capteur à base d'une résistance métallique.

a-sensibilité thermique : d'une façon générale la valeur d'une résistance dépend de sa température t : $r(t) = R_0 * f(t-t_0)$

R_0 étant la résistance à la température t_0 et la fonction f une caractéristique des matériaux, égale à 1 pour $t=t_0$. C'est ainsi que l'on a :

Chapitre I Généralité sur les stations météo et les serres

Pour les métaux : $R(t) = R_0(1 + At + Bt^2 + Ct^3)$

« t » étant exprimé en °C, $t_0 = 0$ °C ;

-pour les thermistances, mélanges d'oxydes semi-conducteurs:

$R(t) = R_0 \exp [B (1/t - 1/t_0)]$; t étant la température absolue.

Les coefficients de la loi de variation de R ayant été préalablement précisés par un ensemble de mesures à température connues, la détermination de la valeur de R permet d'en déduire sa température.

Pour de petites variations ΔT de la température autour d'une valeur T, la loi générale de variation de résistance peut être linéarisé :

$R(T - \Delta T) = R(T) \cdot (1 + \Delta r \cdot -\Delta T)$

Δr : est le coefficient de température de la résistance ou sensibilité thermique à la température T ; Δr dépende évidemment de la température et du matériau. Ainsi par exemple à °C : pour platine : $\Delta r = 3.9 \cdot 10^{-3} / ^\circ C$

Pour un certain type de thermistance : $\Delta r = 5.2 \cdot 10^{-2} / ^\circ C$. Si l'on mesure la température au voisinage de 0°C à l'aide d'un pont de « WHEATSTONE » dont l'une de branches est constituée par la résistance thermométrique et les trois autres branches sont formées par trois résistance fixes et égale à R_0 , résistance thermométrique à 0°C,

La tension de déséquilibre est : $V_m = \frac{E_s}{4} * \frac{\Delta R}{R} = \frac{E_s}{4} * \Delta r \cdot \Delta T$ Pour $E_s = 2V$ et $\Delta T = 1^\circ C$,

$V_m = 1.9mV$ pour la résistance de platine, $V_m = 26mV$ pour la thermistance considérée. Ces valeurs sont notablement supérieures à celles fournies par un thermocouple : Fer / constantan : $V_m = 0.05 mV$. OU (pt -Rh (10%))/pt : $V_m = 0.005 mV$. La qualité de l'appareillage mesure fixe une valeur minimale mesurable $(\Delta R / R_0)_{min}$; il en résulte alors la minimale de variation de température mesurable soit : $\Delta T_{min} = (1 / \Delta r \cdot \Delta R / R_0)_{min}$.

Pour $(\frac{\Delta R}{R_0})_{min} = 10^{-6}$ et pour des mesures faites autour de 0°C on a : avec résistance de

platine : $\Delta T_{min} = 2.6 * 10^{-4} ^\circ C$

Et avec la thermistance considérée : $\Delta T_{min} = 2 * 10^{-5} ^\circ C$

La variation thermique de la résistance est en principe, liée à la fois aux modifications de sa résistivité Δ et de ses dimensions géométrique ; pour un fil cylindrique (longueur L ; sections) :

Chapitre I Généralité sur les stations météo et les serres

$$\Delta_r = 1/R \cdot \Delta R / \Delta T = 1/\rho \cdot d\rho/dt + 1/L \cdot dL/dt - 1/s \cdot ds/dt.$$

Ou :

$1/\rho \cdot d\rho/dt = \Delta_\rho$, coefficient de température de la résistivité de la résistivité de matériau

$1/L \cdot dL/dt = \Delta_L$ et $1/s \cdot ds/dt = 2\Delta_l$, Δ_l étant le coefficient de dilatation linéaire du matériau ; il en résulte : $\Delta_r = \Delta_\rho - \Delta_l$ Mais pratiquement on trouvera : $\Delta_r = \Delta_\rho$

Remarque : L'exploitation du signal de mesure est facile lorsque sa variation est linéaire en fonction de la température, pour cela une plus simple méthode de linéarisation consiste à associer un montage Adéquat au capteur, en série ou en parallèle selon le cas.

I.2.4.4 Baromètre électronique :

L'unité de pression et dans le système S-I. le pascal(pa), qui représente un newton par mètre carré . cette valeur, peu pratique , est souvent remplacée par le bar qui vaut 100 000 pa. Une autre unité, l'atmosphère (atm) est une valeur moyenne de la pression atmosphérique Moyenne, soit $1.013 \times 10^5 = 1.013$ bar ou 1013 millibars, unité plus familière puisque figurant sur le cadran des baromètres domestique. On admet aussi que cette valeur normal est de 760 mm de mercure au niveau de mer . on trouvera aisément sur le baromètre les valeurs 1013 et 76 (en centimètre cette fois – ci), quasiment alignées. Sachez enfin que le millibar a laissé la place à son équivalent hecto pascal (hpa), de nos jours pour mesuré cette grandeur d'une manière totalement électronique, il suffit de "peser" en quelque sorte le poids de la colonne d'aire . pour se faire , on exploite les propriétés piezo-résistif d'une jauge de contrainte minuscule pastille de silicium, un peut à la manière d' un jauge de contrainte minuscule capable de distinguer les infimes variations de masse sur sa surface sensible. La société **Motorola** propose un composant intéressant , compensé température et cplibré précisément en usine : le capteur MPX2200AP , ce circuit est un capteur de pression faisant appel à l'effet piezo-résistif induit par quatre résistances, montées sur une fine membrane de silicium.

Le composant est étalonné très précisément en usine et compensé en température, ses applications le destinent à la robotique, à la commande de pompes diverses, à la construction de mesureurs barométriques, altimétriques, médicaux, etc.

I.2.4.5 pluviomètre numérique :

Le pluviomètre classique à lecture directe comporte un récipient surmonté d'un entonnoir, dont la section réceptrice est connue à intervalles réguliers, on vide l'eau accumulée en mesurant préalablement la hauteur atteinte, qui représente au $1/10^{\circ}$ de Millimètre près la quantité d'eau tombée, c'est-à-dire le volume en litres par mètre carré de Terrain.

Il existe également le pluviomètre à pesée ou le modèle à flotteur, ces appareils recueille bien entendu toute l'eau atmosphérique, sous ses divers états (pluies, neige, bouillards). Pour résoudre le problème du capteur, la conception d'un simple entonnoir en plastique a été mis au point avec un Bouchon poreux, constitué d'un morceau de "scotch bride vert" dont l'orifice est obturé vers le bas,

Afin de favoriser la constitution de goutte d'eau régulières, tout comme une éponge ou un chiffon imbibé le feraient. Le goutte à goutte régulier doit tomber exactement sur les deux électrodes métalliques très proches l'un de l'autre, et inclinées vers le bas pour faciliter l'évacuation rapide du

Liquide détecté. Le temps de contact, il se produira une série d'impulsions rapides qui seront traitées,

Comptées et finalement affichées. Le réglage de l'écartement des électrodes reste la seule opération quelque peu délicate avec l'étalonnage du pluviomètre, la fréquence exacte de l'oscillateur de base n'a guère d'importance et sera modifiée aisément par une ajustable, avec un facteur de division faible

Qui sera d'abord programmé. Un contact entre les deux sondes sera obtenu, ou mieux encore à l'aide d'un liquide d'un liquide déversé dans le capteur. Une bascule bistable sera auparavant validé, une quantité d'eau connue pourra être versé dans l'entonnoir, on essaiera par tous les réglages prévus d'obtenir la valeur en millimètre correspondant à cette quantité, répartie bien entendu sur un mètre carré.

I.2.4.6 capteur de l'humidité :

L'humidité de l'air en générale est susceptible d'avoir des répercussions importantes sur un certain nombre de processus physico- chimiques ou biologiques. C'est pourquoi sa mesure c'est imposée pour notre system à savoir la gestion d'une serre.

I.2.4.6.1 Définition d'humidité :

L'humidité. C'est la quantité de vapeur d'eau contenue dans l'air, elle dépend principalement de la température de l'air. Ainsi, l'air chaud contient plus de vapeur d'eau que l'air froid. Un climat tropical sera humide, c'est-à-dire chaud et moite. L'air est saturé lorsqu'il ne peut plus contenir de vapeur d'eau, on parle alors d'humidité relative maximale. La quantité de vapeur réelle contenue dans l'air est exprimée sous le nom l'humidité relative ; elle représente en pourcentage la quantité maximale que cette masse d'air pourrait contenir à une même température naturellement. Une sensation de bien-être est admise entre 40% et 70% de H .R. à la condition que la température ambiante soit également satisfaisante (entre 18 et 20°C).

I.2.4.6.2 principales définition relatives a l'air humide : En considèrent

- Un volume **V** d'air d'humide à une température **T**.
- La masse **M** d'air humide contenue dans ce volume est la somme d'une masse M_a d'air sec et la masse M_v De vapeur d'eau.
- La pression totale P, ou pression barométrique, est la somme des pressions partielles **P_a** de l'air sec et **P_v** de la vapeur d'eau.

On définit les paramètres suivant :

➤ Rapport de mélange **r (kg/kg)**

C'est le rapport entre la masse M_v de vapeur d'eau et la masse M_a d'air sec à laquelle cette vapeur d'eau est mélangée : $r = M_v/M_a$ c'est la grandeur de référence en humide

➤ Pression de vapeur saturante **P_s (T, Pa)**

La Pression de vapeur saturante à une température T, est valeur maximum que peut atteindre la pression partielle **P_v** de la vapeur, à la température T ; au-delà il y a condensation.

➤ Humidité relative **U(%)** : rapport de pression partielle de vapeur d'eau à la pression de saturante à la température T : $U = P_v/P_s(T).100$ c'est la grandeur la plus couramment utilisée.

➤ Température de rosée **Tr (°C)** : Température à laquelle il faut refroidir l'air humide pour atteindre la saturation, le rapport de mélange « r » restant constant pendant le refroidissement. Cette température Td est t'elle que : **$P_v = P_s(Tr)$**

I.3 Généralité sur les serres [2] :

Par souci d'économiser le coût d'une gestion d'une serre agricole nous allons associer la station météo à la commande d'une serre afin de bénéficier des états du climat externe, sans activer les actionneurs tel que le ventilateur pour le refroidissement, le brûleur pour le chauffage, les lampes pour la luminosité et l'humidificateur pour l'humidité et cela pour minimiser les dépenses du point de vue énergétique. Avant de passer au contrôle de la serre il faut bien choisir l'emplacement et le type de la serre tels que

I.3.1 l'emplacement des serres :

Un bon éclairage et une protection contre les vents violents sont les deux facteurs fondamentaux à prendre en considération. –

Pour profiter au maximum de lumière du bas soleil d'hiver, la serre doit être montée avec son grand axe dans la direction est-ouest, ou le plus près possible de cette orientation.

-la protection contre le vent qui consiste à planter une haie ou à ériger une palissade comme brise-vent. Placées à une distance de la serre pour lutter contre les vents dominants.

-prévoir un accès facile, et une alimentation en énergie électrique d'autres considérations, telle que l'aspect et l'ombre, peuvent intervenir.

I.2.3 types de serres :

Les serres sont de taille, de forme et de types très divers pour répondre à tous les besoins. Les facteurs fondamentaux dont on doit tenir compte son l'usage auquel la serre est destinée, la somme d'argent dont on dispose et l'emplacement où cette serre doit être érigée. Les serres peuvent avoir des murs droits ou inclinés vers l'intérieur. Quant à leur toit, il peut être simplement à deux versants ou bien en croupé avec un arêtier ou encore curvilinéaire à double croupé. Le

modèle de la serre qu'on va commander doit posséder, l'équipement suivant :

- moteurs d'ouvrants 2 cotes.
- moteurs d'écran thermique ou d'ombrage.
- moteurs de ventilation.
- commande d'éclairage artificiel.
- commande d'humification.-mélangeur sur le circuit de chauffage à basse température.

Chapitre I Généralité sur les stations météo et les serres

Ce choix va nous permettre de contrôler les cinq fonctions suivantes :

1) fonction aération :

L'objectif principal de la régulation de l'aération est d'empêcher la température de l'air dans la serre de dépasser la température maximum admise. L'aération peut être également régulée à des fins d'humidification de l'atmosphère, et pour favoriser l'entrée du CO₂ dans la serre, les moteurs des ouvrants ou des extracteurs sont mis en route en fonction de mesures de température ou d'hygrométrie de l'air, en ce qui concerne les ouvrants, leur ouverture doit pouvoir être réduite, limitée ou interdite en cas de vent fort, L'aération par ouvrants est donc généralement asservie à des mesures de vitesse du vent et de sa direction, et de la température externe.

2) fonction ombrage : la régulation de l'ombrage a pour objectif presque unique la limitation de la quantité d'énergie solaire entrant dans la serre qui est comme on l'a vu à l'origine d'une augmentation souvent exagérée des températures à l'intérieur de la serre, le fonctionnement des enrouleurs de câbles ou des moteurs à crémaillères entraînant les écrans d'ombrage dépend donc de mesures du temps, de la température d'air intérieur ou du niveau de rayonnement reçu.

3) fonction éclairage : On peut employer des installations d'éclairage pour accélérer la croissance des plants et pour modifier leur vitesse de développement dans le but d'obtenir des effets spéciaux. De nombreuses plantes sont très sensibles à la « longueur du jour », la période pendant laquelle la lumière est assez forte pour qu'elles se développent. Dans les régions septentrionales et dans celles à forte pollution atmosphérique, ce niveau est souvent loin d'être atteint en hiver. Si on envisage de placer des installations d'éclairage, ils doivent comporter des appareillages résistants et étanches à l'humidité. L'employer des lampes à vapeur de mercure est préférable, car le type de lumière qu'elles produisent est meilleur pour la croissance des végétaux.

4) fonction chauffage :

Dans les régions à température les plus fraîches ou il se produit régulièrement des gelées en hiver, la chaleur du soleil seule est trop faible et trop aléatoire pour que l'on réussisse à faire pousser correctement sous abri les plantes très délicates. Aussi pour tirer la meilleure partie possible d'une serre, on doit y installer une source de chaleur artificielle. On cite les

Chapitre I Généralité sur les stations météo et les serres

Types de chauffages suivant :

- Tuyaux d'eau chaude chauffée par : un combustible solide, fuel ou gaz.
- chauffage par radiateur au gaz ou pétrole.
- chauffage électrique : le chauffage électrique est le plus efficace et celui qui donne le Meilleur rendement, il est facile à régler et à nettoyer et c'est celui qui présente le moins de danger pour les plantes car il ne dégage pas de fumées.

5) Fonction Humidification de l'air : la régulation de humidification de l'air à pour objectif le maintien de l'hygromètre des serres au dessus de la valeur fixée par l'utilisation, la brumisation ou la nébulisation fine sont déclenchées en fonction de mesure de l'hygrométrie de l'air.

Conclusion :

On s'est intéressé dans ce chapitre aux généralités sur les stations météorologiques ainsi que les serres agricoles. Le but de notre projet est de réaliser une station météo qui nous permettra d'acquérir les données du climat externe, ces dernières pourront être transmises à un système qui gère une serre agricole. Les paramètres climatiques transmis sont :

- Rayonnement global.
- humidité relative de l'air extérieure
- Vitesse du vent
- Orientation du vent
- Température de l'air extérieur.

Ces données seront reportées à un système de contrôle intelligent qui prendra la décision d'utiliser ces paramètres en ouvrant les volets et utilise des stores pour l'ombrage ou d'activer les actionneurs existant dans la serre, afin d'avoir :

- une économie d'énergie pour l'activation des chaudières, des ventilateurs et de l'éclairage artificiel, par la comparaison du climat interne de la serre avec le climat externe.
- une amélioration de la sécurité de la serre et de l'installation grâce au contrôle des vents.
- un gain de temps lié à l'automatisation de certaines tâches et à la centralisation des informations et des commandes.
- une amélioration du rendement des cultures, obtenue à une meilleure maîtrise des conditions optimales de culture.

II.1 Introduction :

De nos jours pratiquement tous les systèmes de commande et de contrôle sont basés sur un C.P.U, ces dernières peuvent être trouvées sous plusieurs formes c'est-à-dire à base microprocesseur, d'un microcontrôleur, ou d'un circuit programmable comme la carte électronique « Arduino ». Le développement des cartes électroniques modernes, associé à leur faible coût, a permis leur utilisation dans des applications les plus diverses. Pour cela, notre étude et réalisation est basée sur une carte électronique programmable « Arduino ».

Ce dernier est composé d'un nombre important d'éléments, on s'intéressera surtout aux éléments qui toucheront l'aspect pratique de notre composant, avec des exemples, des schémas, des chronogrammes....etc. La carte électronique « Arduino ». Est équipée de plusieurs outils dont les principaux sont d'un « **microcontrôleur AtmelAVR** » et de composants complémentaires qui facilitent la programmation et l'interfaçage avec d'autres circuits. Chaque module possède au moins un « régulateur linéaire » 5 V et un oscillateur à « quartz » 16 MHz

Le microcontrôleur est préprogrammé avec un « boot loader » de façon à ce qu'un programmeur dédié ne soit pas nécessaire.

Les modules sont programmés au travers d'une connexion série « RS-232 », mais les connexions permettant cette programmation diffèrent selon les modèles. Les premiers Arduino possédaient un port série, puis « l'USB » est apparu sur les modèles Décimal, tandis que certains modules destinés à une utilisation portable se sont affranchis de l'interface de programmation, relocalisée sur un module USB-série dédié (sous forme de carte ou de câble). L'Arduino utilise la plupart des « entrées/sorties » du microcontrôleur pour l'interfaçage avec les autres circuits. Le modèle Décimal par exemple, possède 14 entrées/sorties numériques, dont 6 peuvent produire des signaux « PWM », et 6 entrées analogiques. Qui ne consomme que peu de courant (40mA à 500mA). Il existe une grande variété de la carte « arduino », le tableau ci-dessous montre un petit récapitulatif des différentes versions :

II.2 Choix de la carte électronique « arduino » :

Le choix de la carte électronique « arduino » se fera suivant leur caractéristiques, le microcontrôleur à réaliser et le cout, et d'après ces paramètres on a constaté que la carte « **Arduino** » le plus adéquat pour notre travail est la carte « **arduino-uno** » (**fig.-II.2**), qui a les caractéristiques suivantes :

Arduino	Processeur	Flash ko	EEPROM ko	SRAM ko	Broches d'E/S numériques	...avec PWM	Broches d'entrée analogique	Type d'interface USB
Diecimila	ATmega168	16	0,5	1	14	6	6	FTDI
Duemilanove	ATmega168/328P	16/32	0,5/1	1/2	14	6	6	FTDI
Uno	ATmega328P	32	1	2	14	6	6	ATmega8U2
Leonardo	ATmega32U4	32	1	2,5	20	7	12	ATmega32U4
Mega	ATmega1280	128	4	8	54	15	16	FTDI
Mega2560	ATmega2560	256	4	8	54	15	16	ATmega8U2

Tab. II.1 caractéristiques des différentes carte « arduino »

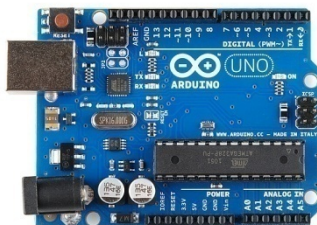


Fig. II.2 carte arduino -uno

II.3 Description de la carte « Arduino-Uno » [4]:

La carte **Arduino -Uno** (fig. II.3) est une carte à microcontrôleur basée sur l'**ATmega328**

Elle dispose :

- de 14 broches numériques d'entrées/sorties (dont 6 peuvent être utilisées en sorties **PWM** (largeur d'impulsion modulée)),
- de 6 entrées analogiques (qui peuvent également être utilisées en broches entrées/sorties numériques),
- d'un quartz **16Mhz**,
- d'une connexion **USB**,
- d'un connecteur d'alimentation jack,
- d'un connecteur **ICSP** (programmation "in-circuit"),
- et d'un bouton de réinitialisation (**reset**).

Elle contient tout ce qui est nécessaire pour le fonctionnement du microcontrôleur; Pour pouvoir l'utiliser et se lancer, il suffit simplement de la connecter à un ordinateur à l'aide d'un câble USB (ou de l'alimenter avec un adaptateur secteur ou une pile, mais ceci n'est pas indispensable, l'alimentation étant fournie par le port USB).

La carte **Arduino-Uno** diffère de toutes les cartes précédentes car elle n'utilise pas le circuit intégré FTDI USB-vers-série. A la place, elle utilise un Atmega8U2 programmé en convertisseur USB-vers-série.

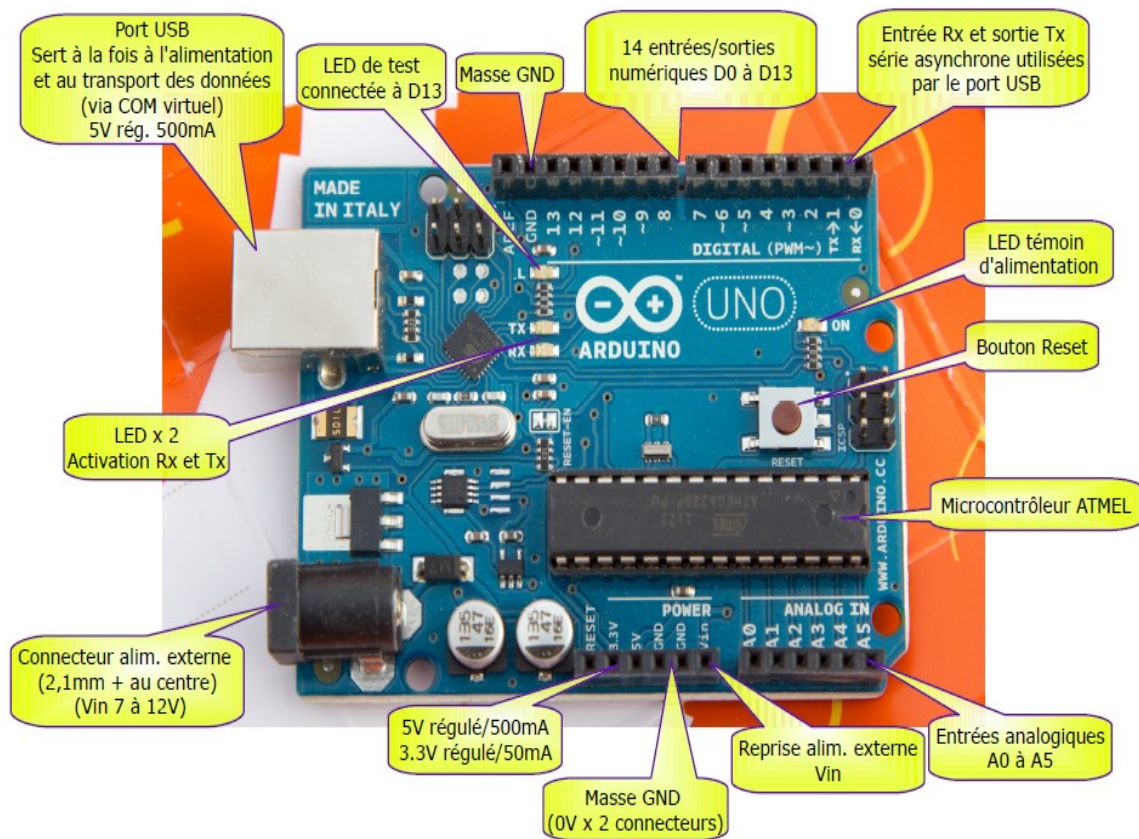


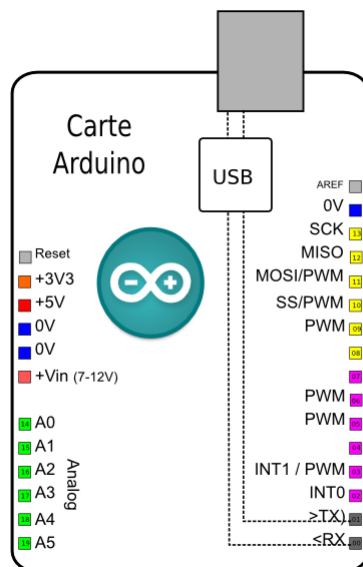
Fig. II.3 Description de la carte « arduino-uno »

II.3.1 Synthèse des caractéristiques de la carte Arduino - uno :(tab. II.3.1)

Microcontrôleur	ATmega328
Tension de fonctionnement	5V
Tension d'alimentation (recommandée)	7-12V
Tension d'alimentation (limites)	6-20V
Broches E/S numériques	14 (dont 6 disposent d'une sortie PWM)
Broches d'entrées analogiques	6 (utilisables en broches E/S numériques)
Intensité maxi disponible par broche E/S (5V)	40 mA : (200mA cumulé pour l'ensemble des broches E/S)
Intensité maxi disponible pour la sortie 3.3V	50 mA
Intensité maxi disponible pour la sortie 5V	Fonction de l'alimentation utilisée - 500 mA max si port USB utilisé seul
Mémoire Programme Flash	32 KB (ATmega328) dont 0.5 KB sont utilisés par le bootloader
Mémoire SRAM (mémoire volatile)	2 KB (ATmega328)
Mémoire EEPROM (mémoire non volatile)	1 KB (ATmega328)
Vitesse d'horloge	16 MHz

Tab.II.3.1 Synthèse des caractéristiques de la carte arduino uno

II.3.2 Brochage de la carte arduino-Uno :



II.3.3 Alimentation de la carte arduino :

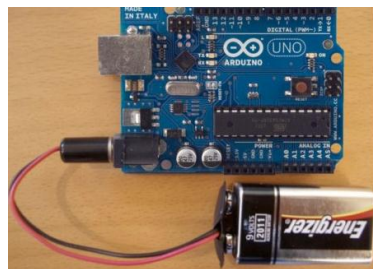


Fig. II.3.3 Alimentation

La carte Arduino Uno peut-être alimentée soit via la connexion USB (qui fournit 5V jusqu'à 500mA) ou à l'aide d'une alimentation externe. La source d'alimentation est sélectionnée automatiquement par la carte.

L'alimentation externe (non-USB) peut être soit un adaptateur secteur (pouvant fournir typiquement de 3V à 12V sous 500mA) ou des piles (ou des accus). L'adaptateur secteur Peut être connecté en branchant une prise 2.1mm positif au centre dans le connecteur jack de la carte. Les fils en provenance d'un bloc de piles ou d'accus peuvent être insérés dans les connecteurs des broches de la carte appelées **Gnd** (masse ou 0V) et Vin (Tension positive en entrée) du connecteur d'alimentation.

La carte peut fonctionner avec une alimentation externe de 6 à 20 volts. Cependant, si la carte est alimentée avec moins de 7V, la broche 5V pourrait fournir moins de 5V et la carte pourrait être instable. Si on utilise plus de 12V, le régulateur de tension de la carte pourrait chauffer et endommager la carte. **Aussi, la plage idéale recommandée pour alimenter la carte Uno est entre 7V et 12V.**

Les broches d'alimentation sont les suivantes :

- **VIN.** La tension d'entrée positive lorsque la carte Arduino est utilisée avec une source de tension externe (à distinguer du 5V de la connexion USB ou autre source 5V régulée). Vous pouvez alimenter la carte à l'aide de cette broche, ou, si l'alimentation est fournie par le jack d'alimentation, accéder à la tension d'alimentation sur cette broche.

- **5V**. La tension régulée utilisée pour faire fonctionner le microcontrôleur et les autres composants de la carte (pour info : les circuits électroniques numériques nécessitent une tension d'alimentation parfaitement stable dite "tension régulée" obtenue à l'aide d'un composant appelé un régulateur et qui est intégré à la carte Arduino). Le 5V régulé fourni par cette broche peut donc provenir soit de la tension d'alimentation VIN via le régulateur de la carte, ou bien de la connexion USB (qui fournit du 5V régulé) ou de toute autre source d'alimentation régulée.
- **3V3**. Une alimentation de **3.3V** fournie par le circuit intégré **FTDI** (circuit intégré faisant l'adaptation du signal entre le port USB de votre ordinateur et le port série de l'**ATmega**) de la carte est disponible : ceci est intéressant pour certains circuits Externes nécessitant cette tension au lieu du 5V). L'intensité maximale disponible sur cette broche est de 50mA.
- **GND**. Broche de masse (**ou 0V**).

II.3.4 Mémoire de la carte arduino -uno :

L'ATmega 328 a 32Ko de mémoire FLASH pour stocker le programme (dont 0.5Ko également utilisés par le bootloader). L'ATmega 328 a également 2ko de mémoire SRAM (volatile) et 1Ko d'EEPROM (non volatile - mémoire qui peut être lue à l'aide de la [librairie EEPROM](#)).

Pour info : Le bootloader est un programme préprogrammé une fois pour toute dans l'ATmega et qui permet la communication entre l'ATmega et le logiciel Arduino via le port USB, notamment lors de chaque programmation de la carte.

II.3.5 Entrées et sorties numériques :

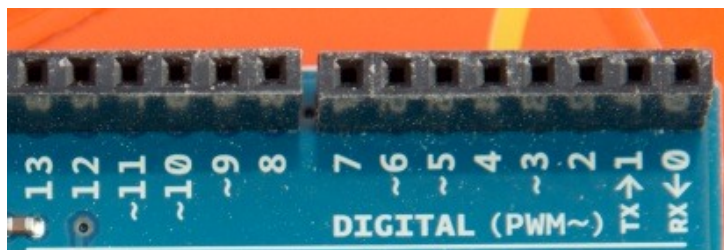


Fig. II.3.5 broches numériques de la carte arduino-uno

Chacune des 14 broches numériques de la carte UNO (numérotées des 0 à 13) peut être utilisée soit comme une entrée numérique, soit comme une sortie numérique, en utilisant les instructions [pinMode\(\)](#), [digitalWrite\(\)](#) et [digitalRead\(\)](#) du langage Arduino. Ces broches fonctionnent en 5V. Les signaux véhiculés par ces connecteurs sont des signaux logiques compatibles TTL, c'est-à-dire qu'ils ne peuvent prendre que deux états HAUT (5 Volts) ou BAS (0 Volt).

Chaque broche peut fournir ou recevoir un maximum de 40mA d'intensité et dispose d'une résistance interne de "rappel au plus" (pull-up) (déconnectée par défaut) de 20-50 KOhms. Cette résistance interne s'active sur une broche en entrée à l'aide de l'instruction [digitalWrite](#) (broche, [HIGH](#)).

De plus, certaines **broches** ont des fonctions **spécialisées** :

- **Communication Série:** Broches **0 (RX)** et **1 (TX)**. Utilisées pour recevoir (**RX**) et transmettre (**TX**) les données séries de niveau TTL. Ces broches sont connectées aux broches correspondantes du circuit intégré ATmega8U2 programmé en convertisseur USB-vers-série de la carte, composant qui assure l'interface entre les niveaux TTL et le port USB de l'ordinateur.
- **Interruptions Externes:** Broches **2** et **3**. Ces broches peuvent être configurées pour déclencher une interruption sur une valeur basse, sur un front montant ou descendant, ou sur un changement de valeur. Voir l'instruction [attachInterrupt\(\)](#) pour plus de détails

- **Impulsion PWM** (largeur d'impulsion modulée): Broches (repérées par un ~) **3, 5, 6, 9, 10, et 11**. Fournissent une impulsion PWM 8-bits à l'aide de l'instruction [analogWrite \(\)](#).
- **SPI** (Interface Série Périphérique): Broches **10 (SS), 11 (MOSI), 12 (MISO), 13 (SCK)**. Ces broches supportent la communication SPI (Interface Série Périphérique) disponible avec la [librairie pour communication SPI](#). Les broches **SPI** sont également connectées sur le connecteur **ICSP** qui est mécaniquement compatible avec les cartes **Mega**.
- **I2C: Broches 4 (SDA) et 5 (SCL)**. Supportent les communications de protocole **I2C** (ou interface TWI (**Two Wire Interface - Interface "2 fils"**)), **disponible en utilisant la librairie Wire/I2C (ou TWI - Two-Wire interface - interface "2 fils")**
- **LED: Broche 13**. Il y a une LED incluse dans la carte connectée à la broche 13. Lorsque la broche est au niveau HAUT, la LED est allumée, lorsque la broche est au niveau BAS, la LED est éteinte.

II.3.6 Broches analogiques :



Fig. II.3.6 les Broches analogiques de la carte arduino-uno

La carte Uno dispose de **6** entrées analogiques (**numérotées de 0 à 5**), chacune pouvant fournir une mesure d'une résolution de 10 bits (càd sur 1024 niveaux soit de 0 à 1023) à l'aide de la très utile fonction [analogRead\(\)](#) du langage Arduino. Par défaut, ces broches mesurent entre le 0V (valeur 0) et le 5V (valeur 1023), mais il est possible de modifier la référence supérieure de la plage de mesure en utilisant la broche AREF et l'instruction [analogReference \(\)](#) du langage Arduino.

Pour pouvoir être traitées par le microcontrôleur, ces entrées analogiques sont prises en charge par un CAN (Convertisseur Analogique Numérique ou ADC pour Analog Digital Converter. **fig.II.3.6.A**) dont le rôle est de convertir l'échantillon de tension V_E en une grandeur numérique binaire sur n bits.

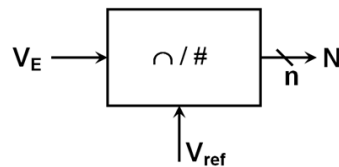


Fig. II.3.6.A Convertisseur Analogique Numérique

Le principe de la conversion Analogique-Numérique est représenté ci-dessous **Fig.II.3.6.B** (avec $n=3$ bits et la tension de référence $V_{ref} = 5$ Volts)

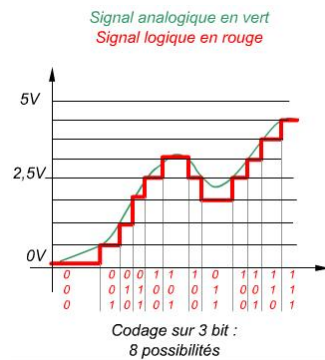


Fig. II.3.6.B Le principe de la conversion CAN 3 bits

Le convertisseur de la carte Arduino Uno possède une résolution de 10 bits, soit $2^{10} = 1024$ possibilités de 0 à 1023.

Ainsi, pour $n=10$ bits et la tension de référence par défaut $V_{ref} = 5$ Volts, si la tension analogique d'entrée échantillonnée est $V_E = 3,8$ Volts, la grandeur numérique N (ici en entier décimal) en sortie du convertisseur peut être calculée grâce aux relations :

$$\text{Quantum } q = V_{ref} / 2^n = 5 / 2^{10} = 5 / 1024$$

$$N = V_E / q = 3,8 \times 1024 / 5 \text{ soit } N = 778$$

Note : les broches analogiques peuvent être utilisées en tant que broches numériques : elles sont numérotées en tant que broches numériques de 14 à 19.

II.3.7 Autres broches de la carte arduino-uno :

Il y a deux autres broches disponibles sur la carte :

- **AREF** : Tension de référence pour les entrées analogiques (si différent du 5V). Utilisée avec l'instruction [analogReference\(\)](#) du langage Arduino.
- **Reset** : Mettre cette broche au niveau BAS entraîne la réinitialisation (= le redémarrage) du microcontrôleur. Typiquement, cette broche est utilisée pour ajouter un bouton de réinitialisation sur le circuit qui bloque celui présent sur la carte

II.3.8 Communication de la carte arduino :

La carte Arduino Uno dispose de toute une série de facilités pour communiquer avec un ordinateur, une autre carte Arduino, ou avec d'autres microcontrôleurs. L'ATmega 328 dispose d'une UART (**U**niversal **A**synchronous **R**eceiver **T**ransmitter ou émetteur-récepteur asynchrone universel en français) pour communication série de niveau TTL (5V) et qui est disponible sur les broches **0 (RX)** et **1 (TX)**. Un circuit intégré **ATmega8U2** sur la carte assure la connexion entre cette communication série vers le port USB de l'ordinateur et apparaît comme un port **COM** virtuel pour les logiciels de l'ordinateur. Le code utilisé pour programmer l'ATmega8U2 utilise le driver standard USB COM, et aucun autre driver externe n'est nécessaire. Cependant, sous [Windows, un fichier .inf est requis](#)

Le logiciel Arduino inclut une fenêtre terminal série (ou moniteur série) sur l'ordinateur et qui permet d'envoyer des textes simples depuis et vers la carte Arduino. Les LEDs RX et TX sur la carte clignote lorsque les données sont transmises via le circuit intégré USB-vers-série et la connexion USB vers l'ordinateur (mais pas pour les communications série sur les broches 0 et 1).

Une [librairie Série Logicielle](#) permet également la communication série (limitée cependant) sur n'importe quelle broche numérique de la carte UNO.

L'ATmega 328 supporte également la communication par protocole I2C (ou interface TWI (Two Wire Interface - Interface "2 fils") et **SPI** :

- Le logiciel Arduino inclut [la librairie Wire](#) qui simplifie l'utilisation du bus I2C.
- Pour utiliser la communication SPI (Interface Série Périphérique), la [librairie pour communication SPI](#) est disponible.

II.3.9 Programmation :

La carte Arduino Uno peut être programmée avec le [logiciel Arduino](#) . Il suffit de sélectionner "Arduino Uno" dans le menu **Tools > Board** (en fonction du microcontrôleur présent sur votre carte)

Le microcontrôleur ATmega328 présent sur la carte Arduino Uno est livré avec un bootloader (petit programme de démarrage) préprogrammé qui vous permet de transférer le nouveau programme dans le microcontrôleur sans avoir à utiliser un matériel de programmation externe. Ce bootloader communique avec le microcontrôleur en utilisant le protocole original STK500 ([reference](#), [fichiers C](#)).

Vous pouvez bien sûr passer outre le bootloader et programmer le microcontrôleur via le connecteur ICSP (In-Circuit Serial Programming - "Programmation Série Dans le circuit" en français)

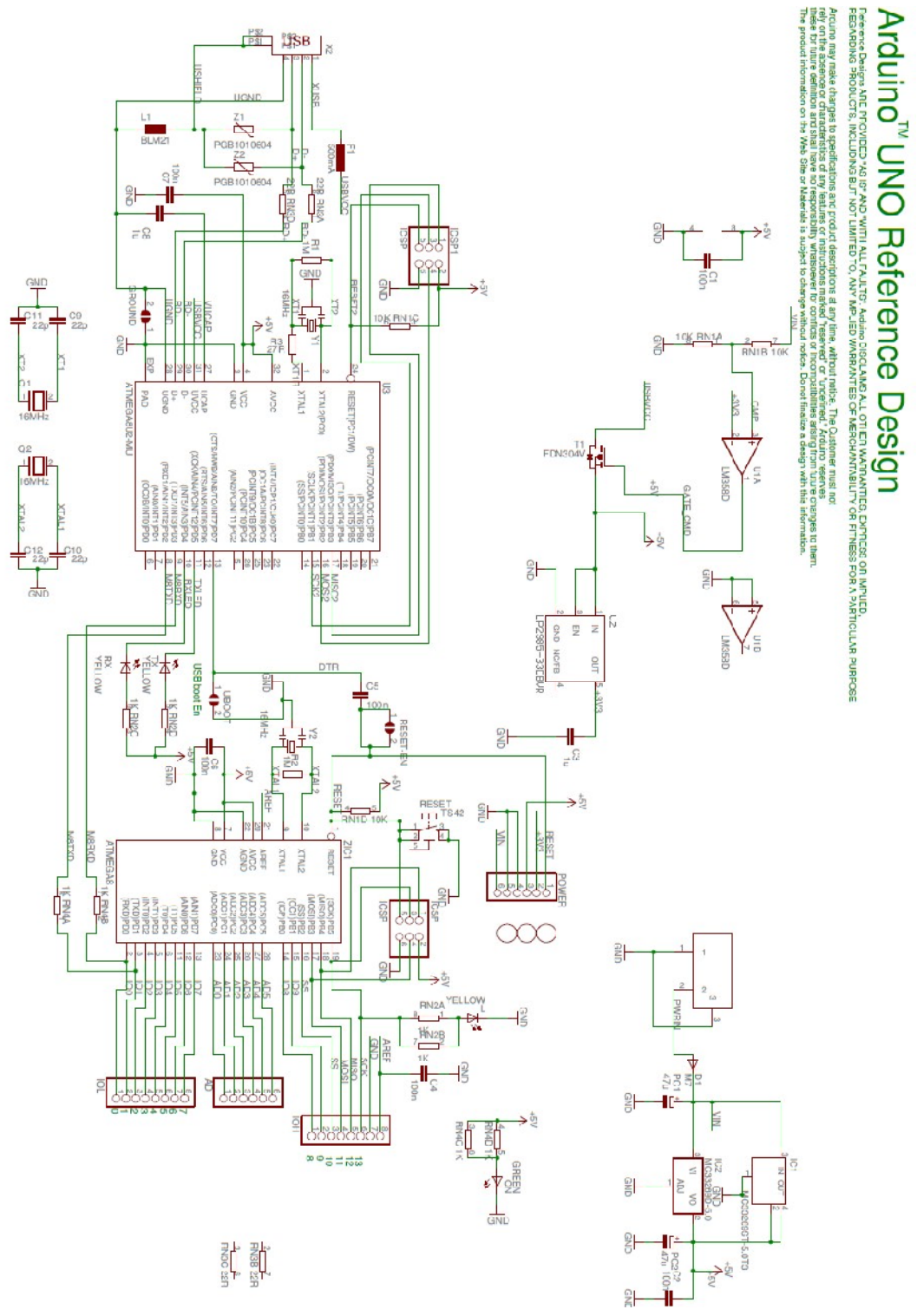
Le source du code pour le circuit intégré ATmega8U2 est disponible. L'ATmega8U2 est chargé avec un bootloader DFU qui peut être activé en connectant le cavalier au dos de la carte (près de la carte de l'Italie) et en réinitialisant le 8U2. Vous pouvez alors utiliser le [logiciel FLIP de chez Atmel](#) (Windows) ou le [programmeur DFU](#) (Mac OS X et Linux) pour charger le nouveau code. Ou bien vous pouvez utiliser le connecteur ICSP avec un programmeur externe (pour réécrire le bootloader DFU).

II.3.10 Réinitialisation (logicielle) automatique :

Plutôt que de nécessiter un appui sur le bouton poussoir de réinitialisation (reset) avant un transfert de programme, la carte Arduino UNO a été conçue de telle façon qu'elle puisse être réinitialisée par un logiciel tournant sur l'ordinateur. Une des broches matérielles de contrôle du flux (DTR) du circuit intégré ATmega8U2 est connecté à la ligne de réinitialisation de l'ATmega328 via un condensateur de 100 nanofarads. Lorsque cette broche est mise au niveau BAS, la broche de réinitialisation s'abaisse

Suffisamment longtemps pour réinitialiser le microcontrôleur. Le logiciel Arduino utilise cette possibilité pour vous permettre de transférer votre programme dans la carte par un simple clic sur le bouton de transfert de la barre de boutons de l'environnement Arduino. Cela signifie que le bootloader peut avoir un temps mort plus court, la mise au niveau bas de la broche DTR étant bien coordonnée avec le début du transfert du programme.

II .3.11 Schéma structurel de la carte ARDUINO UNO :



Reference Design, NOT PROVIDED WITH ALL PARTS: Arduino 2015, ALL OTHER WARRANTIES, EXPRESS OR IMPLIED, REGARDING PRODUCTS, INCLUDING BUT NOT LIMITED TO, ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Arduino may make changes to specifications and product descriptions at any time, without notice. The Customer must not use the Reference Design for any product without the express written consent of Arduino. Arduino does not warrant, and shall have no responsibility, whatsoever, for omissions or incompatibilities arising from future changes to them. The product information on the Arduino Site or Materials is subject to change without notice. Do not finalize a design with this information.

Fig. II.3.11 : schéma structurel de la carte ARDUINO UNO

II.4 Le Microcontrôleur ATMEL ATmega328p [5] :

Le cœur de la carte Arduino Uno est un microcontrôleur de la famille AVR, un Atmel Atmega 328P.

Ce microcontrôleur renferme dans un seul composant :

- un processeur 8 bits à architecture RISC ;
- de la mémoire avec des espaces physiques séparés :
 - mémoire Flash (32 Ko) pour les programmes,
 - mémoire vive SRAM (2 Ko) pour les données,
 - mémoire EEPROM (2 Ko) pour les données de sauvegarde ;
- toute la logique d'horloge (16 MHz) ;
- des circuits d'interface et des périphériques d'entrée-sortie permettant au processeur d'accéder au monde extérieur :
 - des Timers/Counters (T/C) 8 et 16 bits,
 - génération des signaux PWM,
 - des interfaces de communication série (UART, SPI, TWI compatible I2C...)
 - un convertisseur Analogique-Numérique (A/D Conv.)

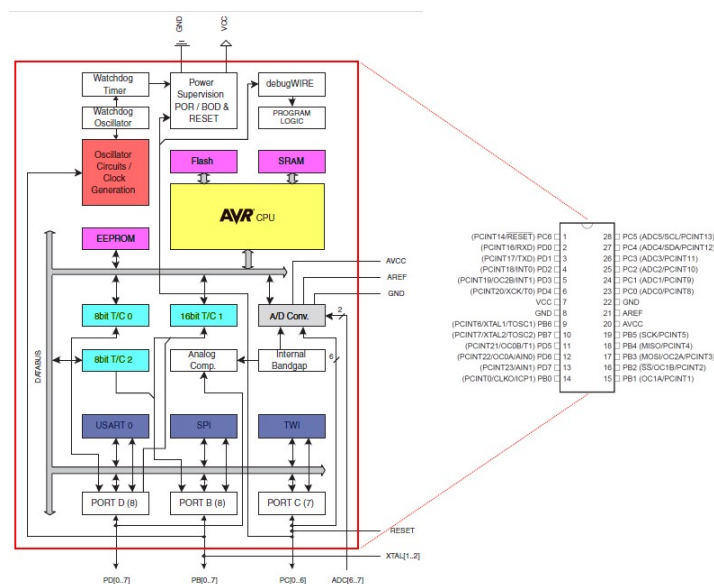


Fig. II.4 Schéma bloc de description du microcontrôleur AVR

Et la figure ci-dessous (**fig.II.4.1**) définir le branchement des broches

De la carte **arduino –uno** Avec les ports de **Microcontrôleur ATMEL ATmega328p**

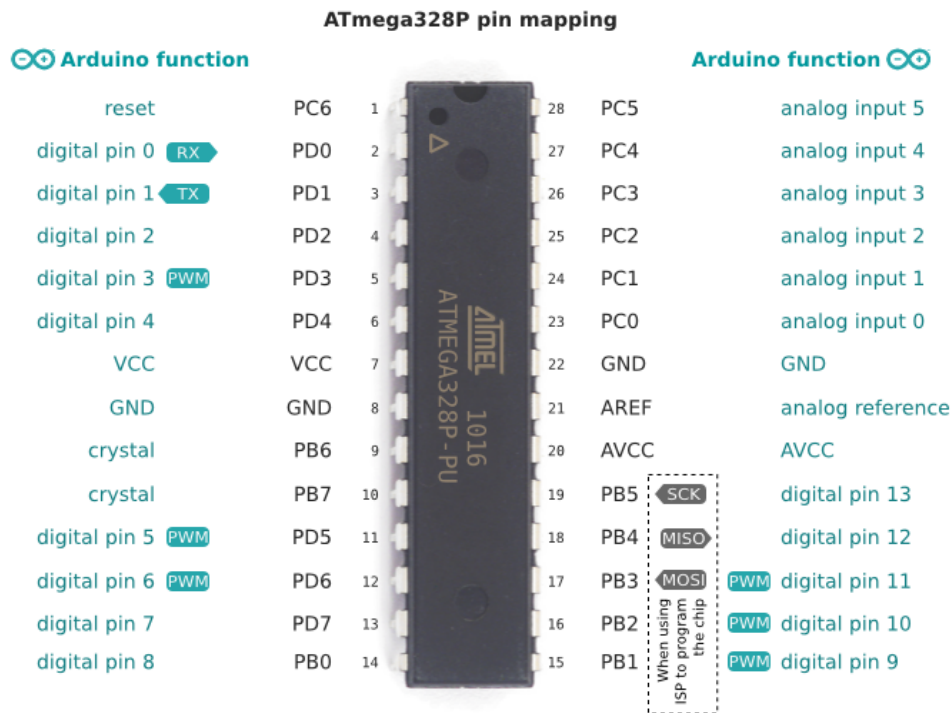


Fig. II.4.1

Conclusion :

On vient de voir dans ce chapitre la description de la carte Arduino Uno en vue d'une application dans une station météo. Cette étude nous a permis de maîtriser presque toutes les fonctions que nous couvrent ce composant à savoir, la conversion analogique - numérique, la Communication série, compteur d'impulsion, ainsi que la différente technique de programmation en langage Arduino afin de réaliser une carte universelle à base de la carte **Arduino Uno**.

Chapitre III Etude et Réalisation du système

III.1.Introduction :

Après avoir, dans les chapitres précédents fait un aperçu sur les données météorologiques pour la réalisation d'une station météo ainsi que l'élément de base du système d'acquisition, on va exposer dans ce chapitre d'une manière claire chacun des blocs constituant le système à réaliser. L'idée de base avec laquelle on a abordé notre réalisation s'articule autour de la représentation et du schéma Synoptique donné par les figures ci-dessous.

III.2 Présentation Générale de l'ensemble

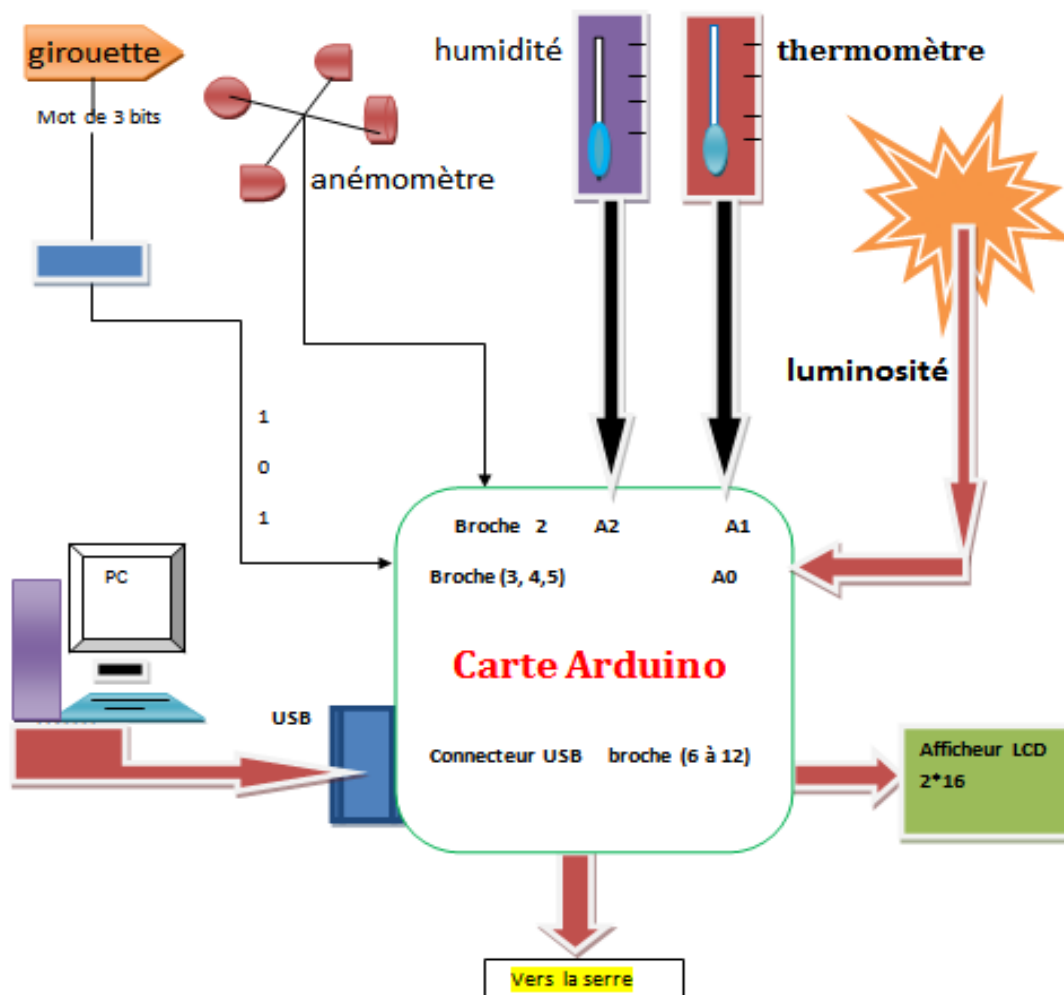


FIG III.2 Présentation Générale de l'ensemble

Chapitre III Etude et Réalisation du système

III.3 Schéma synoptique du système :

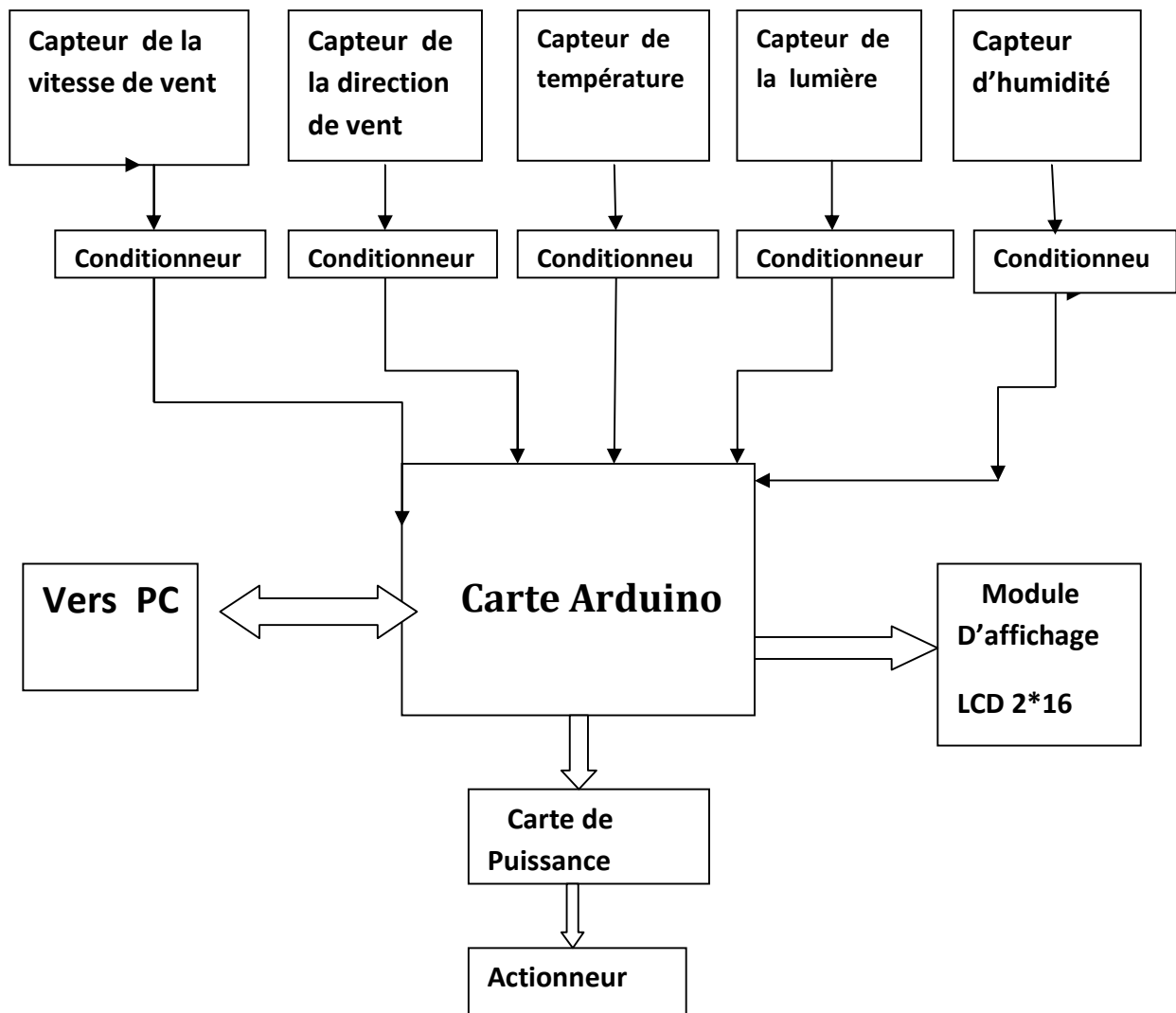


Fig.III.3 Schéma synoptique du système

Comme en voir dans ce schéma synoptique, notre système est constitué de quatre parties essentielles :

- ❖ La station météo, elle est composée de 5 capteurs climatiques et pour chacun leur conditionneur, qui sont :
 - Capteur de la vitesse de vent.
 - Capteur de la direction de vent.
 - Capteur de température.

Chapitre III Etude et Réalisation du système

- Capteur de lumière.
- Capteur de l'humidité.
- ❖ Carte d'acquisition et de traitement des données à base de carte **Arduino-Uno**
- ❖ Interface d'affichage entre la carte d'acquisition et un afficheur à cristaux liquides 2 lignes 16 caractères.
- ❖ Interface de communication entre la carte d'acquisition et un Ordinateur.
- ❖ Carte de puissance pour commander des actionneurs.

III.4 Etude des capteurs :

Plusieurs paramètres caractérisent un signal électrique : par exemple : sa présence (ou absence), sa forme, son amplitude, sa fréquence, son rapport cyclique, son retard ...etc...pour un capteur, ils représentent autant de moyens pour codifier l'**INFORMATION** physique. Le SIGNAL issu du capteur peut être de type ANALOGIQUE ou NUMERIQUE.

III.4.1 Capteur de la vitesse du vent :

Pour la réalisation de l'anémomètre on a choisi un modèle de 4 coupelles [1], Sorte de demi sphères creuses, formant une roue, comme montre la figure ci-après :

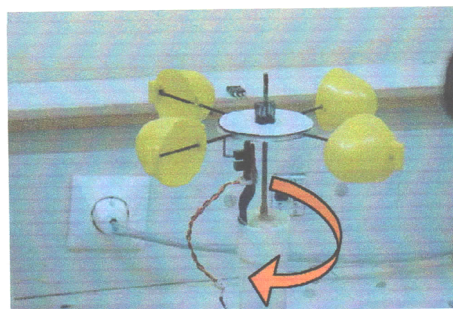


Fig.III.4.1 Anémomètre à 4 coupelles

Lorsque cette roue est placée dans le vent, elle est animée d'un mouvement de rotation et au même temps fait tourner une pièce opaque (secteur non translucide). Cette pièce en rotation fait interrompre à chaque tour un rayonnement infrarouge issu d'une photodiode, de type transitif (émetteur + récepteur).

Chapitre III Etude et Réalisation du système

La vitesse de vent est alors égale au produit de nombre d'impulsions délivré par le capteur sur un intervalle de temps $t=5$ s, par la distance entre l'axe de rotation et le centre d'une coupelle R (exprimée en m).

On a choisi un cercle de rotation de $50\text{ cm} \Rightarrow 0,5\text{ m}$ Donc $2\pi R=50\text{ cm} \Rightarrow R=7,95\text{ cm}$ Ce choix se fait pour faciliter les calculs et pour pouvoir mesurer des faibles vitesses ainsi que des vitesses élevées. C'est l'arduino qui réalise ce processus à l'aide de son accumulateur d'impulsion intégré, il s'agit de compteur 8 bits.

III.4.1.1 circuit de mise en forme :

Le signal qui arrive à la broche **numérique (D 2)** doit avoir une forme carrée que l'arduino puisse le traiter, dans ce signal doit varier entre 0 et 5v. Pour cette raison on a conçu un circuit de mise en forme qui transforme le signal issu du capteur à fourche à un signal équivalent carré (**Fig.III.4.1.1**)

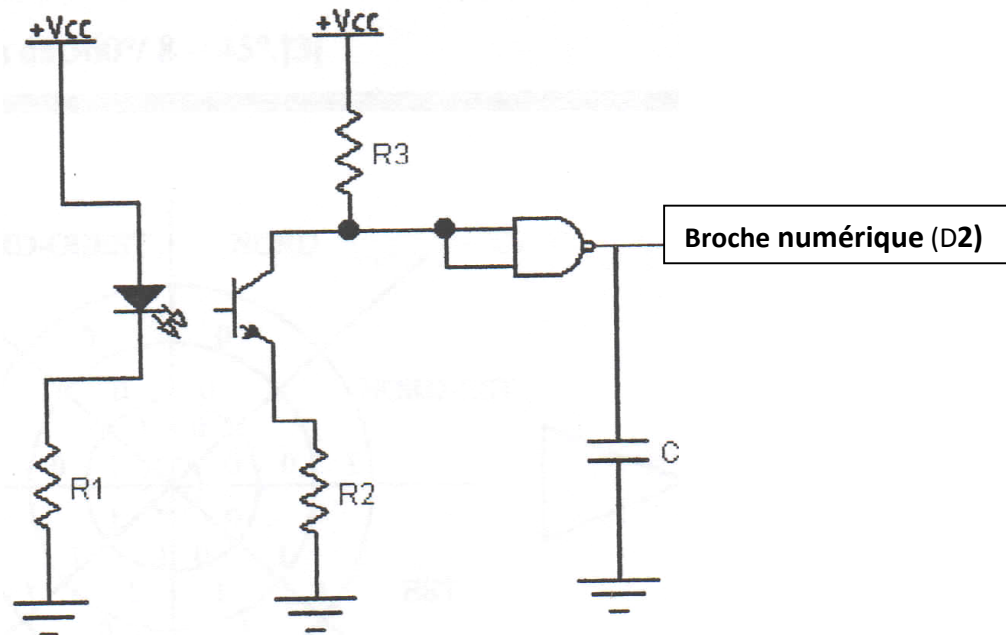
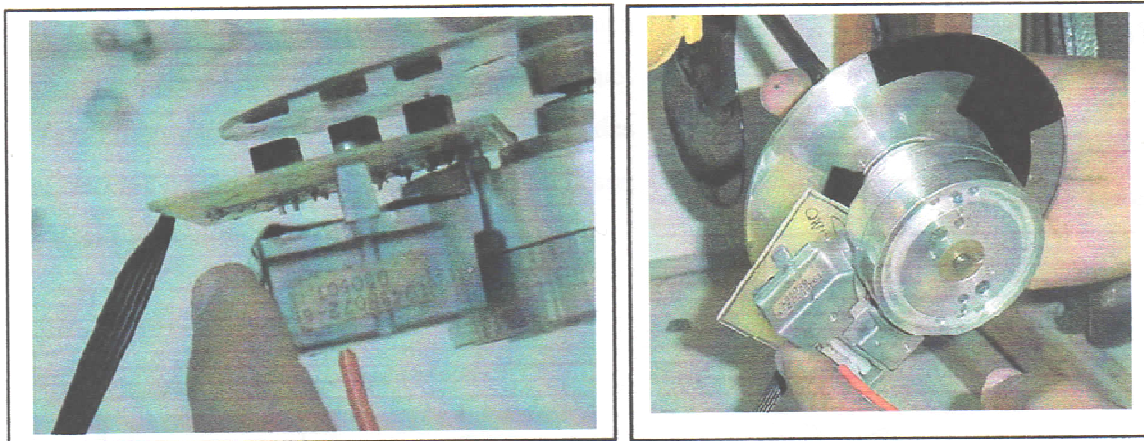
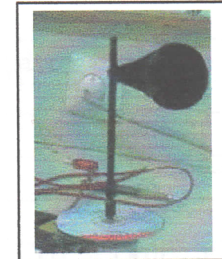


Fig-III.4.1.1 montage de la mise en forme du signal de sortie du capteur

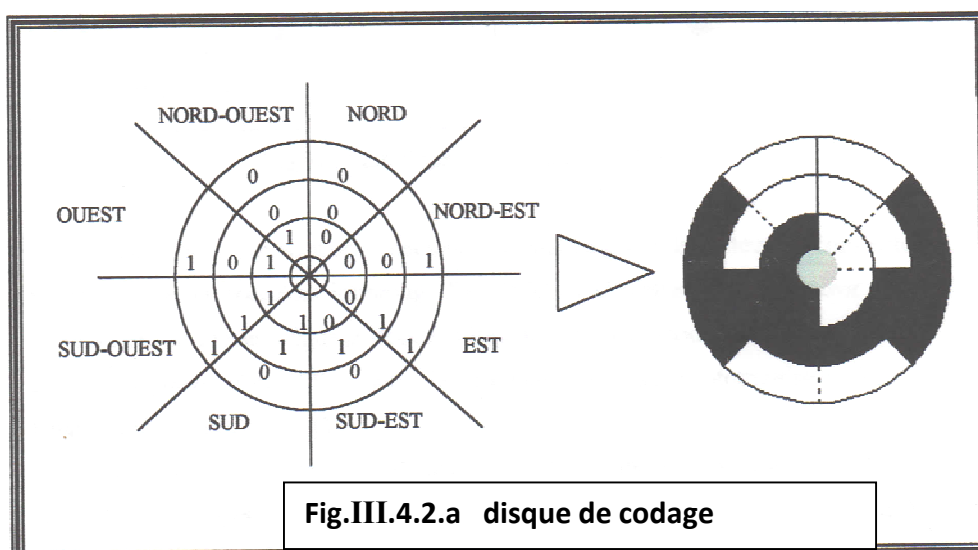
Chapitre III Etude et Réalisation du système

III.4.2 Capteur de la direction du vent :

Après qu'on a vu l'étude de l'anémomètre on vient d'étudier, Maintenant le complément indispensable de l'anémomètre, C'est la girouette, elle est basée sur :



a) Un disque de codage (Fig.III.4.2.a) qu'il est divisé en 8 secteurs égaux, pour avoir 8 directions différentes avec une résolution de $360^\circ/8=45^\circ$. [3]



Chapitre III Etude et Réalisation du système

Chaque secteur un code binaire de 3 variables

A, B, C pour parvient à 2^3 combinaisons. Ces 8 combinaisons Sont ordonnés suivent le code **Gray** dans le tableau Suivent :

b) trois capteurs de couleurs **Fig.III.4.2.b** de type (CNY70) sont utilisé pour généré les codes (A,B,C) Nécessaire.Ce type de capteu permet d'identifier deux type de surface ayant des réflexions différentes : un noir Mat (foncé) et un blanc brillant par exemple.

Capteur de direction			
A	B	C	direction
0	0	0	NORD
1	0	0	NORD-EST
1	1	0	EST
0	1	0	SUD-EST
0	1	1	SUD
1	1	1	SUD- OUEST
1	0	1	OUEST
0	0	1	NORD-OUEST
0=Capteur libre		1=Capteur obturé	

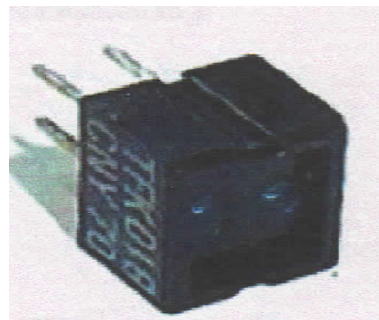
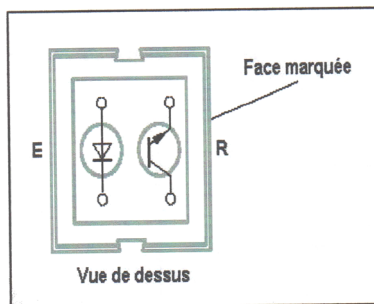


Fig. III.3.2.b Capteur de couleur CNY70

Le **CNY70** est un composant spécialement créé pour ce genre de capteurs. Il comporte un émetteur et un récepteur infrarouge **Fig.-III.4.2.c**. L'émetteur et le récepteur sont accordés sur la même longueur d'onde. L'émetteur est diode infrarouge centrée sur la longueur d'onde 950 nm. Le récepteur est un phototransistor.

Chapitre III Etude et Réalisation du système

Voici l'implantation d'un CNY70 [13].

L'émetteur et le récepteur doivent être face à la surface à analyser. Si la surface est claire, les infrarouges seront réfléchis. Si la surface est mat ou foncée, les infrarouges seront absorbés. Dans le premier cas le Récepteur détectera la présence d'infrarouge, et dans le second, le récepteur sera bloqué.

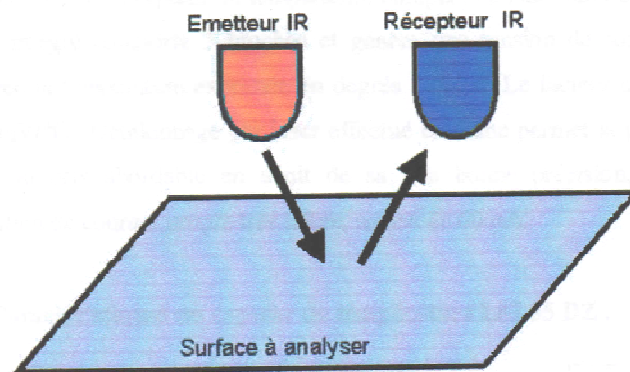


Fig.III.4.2.c Fonctionnement du capteur CNY70

c)- circuit d'acquisition : pour le circuit imprimé [voir ANNEXE]

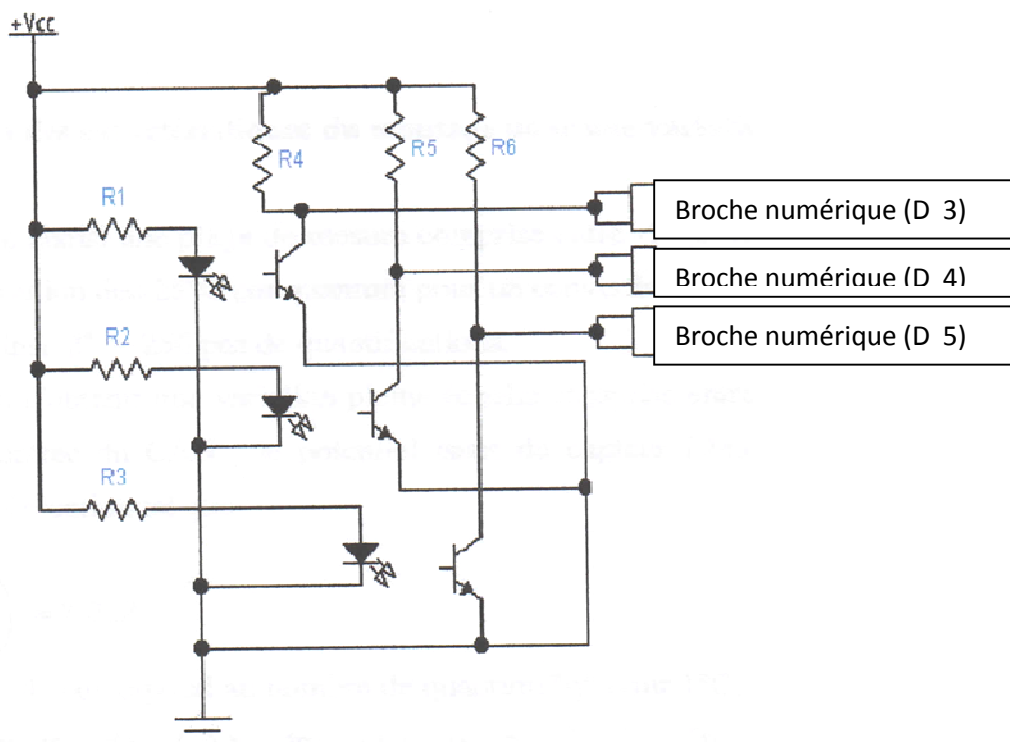


Fig.III.4.2.c Circuit de conditionnement de la girouette.

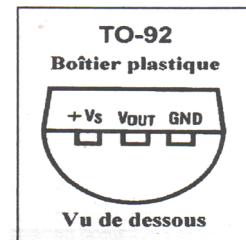
Chapitre III Etude et Réalisation du système

III.4.3 Capteur de température LM35DZ :

Le LM35 DZ est un capteur de température compact de National Semi-conducteur. Ce circuit intégré comporte 3 broches et génère une tension de sortie en relation linéaire avec la température exprimée en degrés Celsius. Le facteur d'échelle adopté est de 10 mV/°C. L'étalonnage Par laser effectué en usine permet la production d'un capteur à un prix abordable en dépit de sa très Bonne précision, ainsi que sa consommation de courant propre très faible, moins de 60 µA.

III.4.3.1 caractéristiques du capteur de température LM 35 DZ :

- plage de la tension d'alimentation 0,2 volt à 35 volts
- sensibilité 10 mv/ °C
- précision +/-0,5°C (à25°C)



III.4.3.2 Calcul des caractéristiques du montage pour une variation pleine échelle du CAN :

- ❖ On a choisi une plage de mesure comprise entre 0°C et +100°C avec une Résolution de 1°C par quantum pour un convertisseur 10 bits.
- ❖ Ainsi $2^{10} = 1024$ pas de quantifications.
- ❖ Le LM35 n'aime pas du tout être monté à l'envers (il part en fumée). La patte +Vs est à connecter à la borne 5V de la carte (au niveau de POWER). La patte GND est à connecter à la borne Gnd de la carte (au niveau de POWER). La patte Vout est à connecter à la borne analogique A1 (au niveau de ANALOGIN fig.III.4.3.2).
- ❖ Le convertisseur analogique / numérique de 10 bits intégré à la carte Arduino numérise le signal analogique sur 1024 niveaux (de 0 à 1023), le 0 correspondant à 0 volt et le 1023 à 5 volts. Le pas entre deux niveaux successifs est donc de $5/1023 = 0,00489 \text{ V} = 4,89 \text{ mV}$. Puisque le capteur LM35DZ fonctionne de manière linéaire (10 mV par °C), le pas entre deux niveaux successifs représente $4,89/10 = 0,489 \text{ °C}$.

Chapitre III Etude et Réalisation du système

Le capteur LM35DZ dispose d'une caractéristique linéaire $V_{\text{capteur}} = f(T)$ suivante :

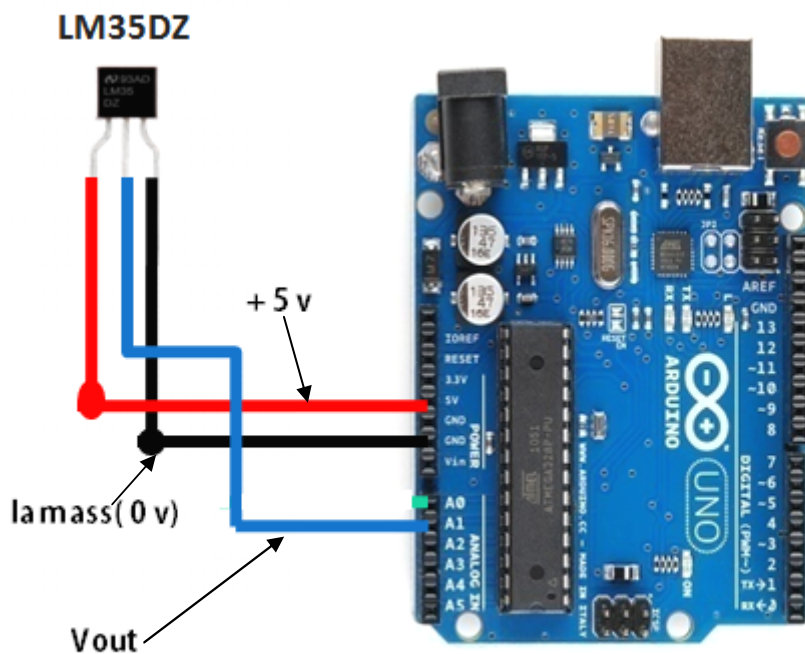
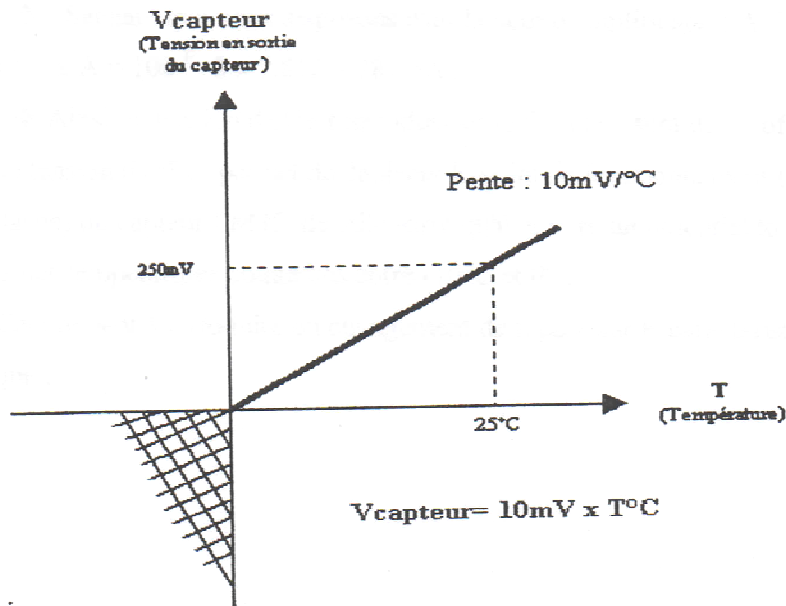


fig.III.4.3.2 Le montage du capteur de température LM35DZ avec l'arduino

Chapitre III Etude et Réalisation du système

III.4.3.3 Conversion la tension Vout du LM35DZ en température avec l'arduino uno[14] :

a) Lire la tension sur une broche analogique :

Un truc très simple avec Arduino, c'est que c'est facile à prendre en main. Et ça se voit une fois de plus avec l'utilisation des convertisseurs numérique –> analogique En effet, vous n'avez qu'une seule nouvelle fonction à retenir : `analogRead()` `analogRead(pin)`

Cette fonction va nous permettre de lire la valeur lue sur une entrée analogique de l'Arduino. Elle prend un argument et retourne la valeur lue :

L'argument est le numéro de l'entrée analogique à lire

La valeur retournée (un `int`) sera le résultat de la conversion analogique>numérique

Exemple `Vout = analogRead(A1)`

b) Conversion la valeur de CAN en tension :

Nous avons une valeur entre 0 et 1023. Cette valeur est l'image de la tension mesurée, elle même comprise entre 0V et +5V. Nous avons ensuite déterminé que le pas du convertisseur était de 4.88mV par unité. Donc, deux méthodes sont disponibles :

- Avec un simple produit en croix
- En utilisant le pas calculé plus tôt

Exemple : La mesure nous retourne une `valeur` de `Vout = 54`.

Avec un produit en croix on obtient : $54 \times 5 / 1024 = 0.264 \text{ V}$

En utilisant le pas calculé plus haut on obtient : $54 \times 4.88 = 0.264 \text{ V}$

c) Conversion la tension en température :

La conversion de la tension issus de LM35 en température T(°c) avec la méthodes suivi :

On a 10 mV par 1°C donc $0.264\text{v} = 26.4\text{mV} = 26.4 \text{ °C}$ l'équation est $T(\text{°c}) = (500 * Vout) / 1023$

Exemple : `Vout= 54` la température est $T(\text{°C}) = (500 * Vout) / 1023 = 26.4\text{mV} = 26.4 \text{ °C}$

Chapitre III Etude et Réalisation du système

d)- Résultat théoriques :

Température (°C)	Tension (V)	Valeur CAN
0	0	0
2	0,015	3
5	0,054	11
10	0,107	22
16	0,156	32
21	0,210	43
24	0,234	48
26	0.264	54
29	0,293	60
35	0,352	72
38	0,386	79
43	0,430	88
46	0,459	94
50	0,503	103

III.4.4 capteur de la lumière :

Pour la réalisation de ce capteur nous avons utilisé une LDR [6] : (Light Dependant Resistor), résistance variable avec la lumière (Fig. III.4.4)

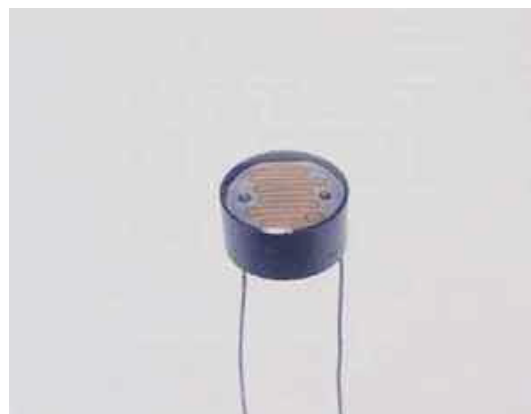


Fig.III.4.4 LDR

III.4.4.1 caractéristique [6] :

Il existe sur le marché quelques modèles de LDR dont la résistance à l'obscurité du modèle dit « standard » est supérieure à $10\text{M}\Omega$, tandis que des modèles un peu plus performants offrent une résistance à 10 lux de l'ordre de $20\text{k}\Omega$ à $100\text{k}\Omega$ et une résistance d'obscurité de $20\text{M}\Omega$

Le circuit de conditionnement est le suivant :

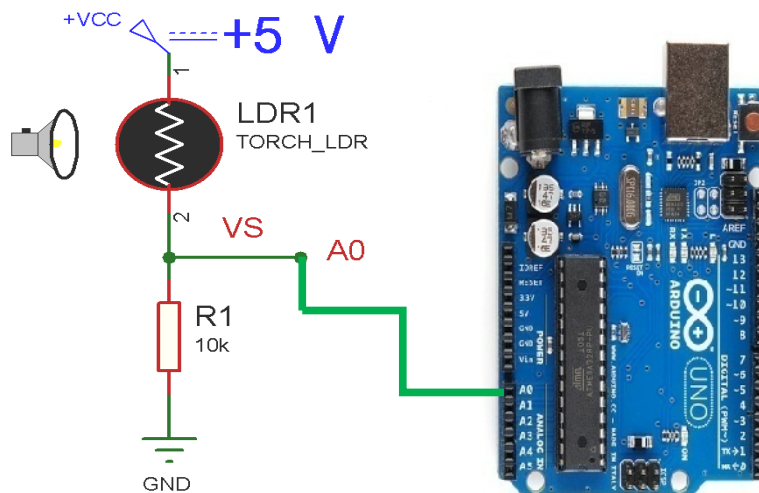


Fig.-III.4.4.1 Circuit de conditionnement du LDR.

La LDR fait varier le potentiel d'un diviseur de tension comme ci-dessus. La tension résultante est acheminée sur la première entrée analogique de la carte arduino-uno (la borne A0), afin d'être traitée par le programme.

III.4.4.2 Linéarisation :

Notre capteur n'est pas linéaire pour cela on va le linéariser avec la méthode software suivante (Fig.III.4.4.2) :

- chargement de la carte « arduino » avec les valeurs linéaires correspondantes.
- lecture des sorties de capteur (entre 0 et 5v).
- comparaison des données du capteur avec les valeurs correspondantes (dans la carte arduino).

Chapitre III Etude et Réalisation du système

L (lux)	0	10	20	25	30	35	40	45	50
Vs (capteur)	0	0.17	0.34	0.42	0.53	0.59	0.74	0.86	0.95
VL (linéaire)	0	0.2	0.41	0.52	0.62	0.72	0.83	0.93	1.04

55	60	65	70	75	80	85	90	95	100
0.98	1.16	1.25	1.33	1.42	1.51	1.58	1.68	1.81	1.95
1.14	1.25	1.35	1.45	1.56	1.66	1.77	1.87	1.97	2.08

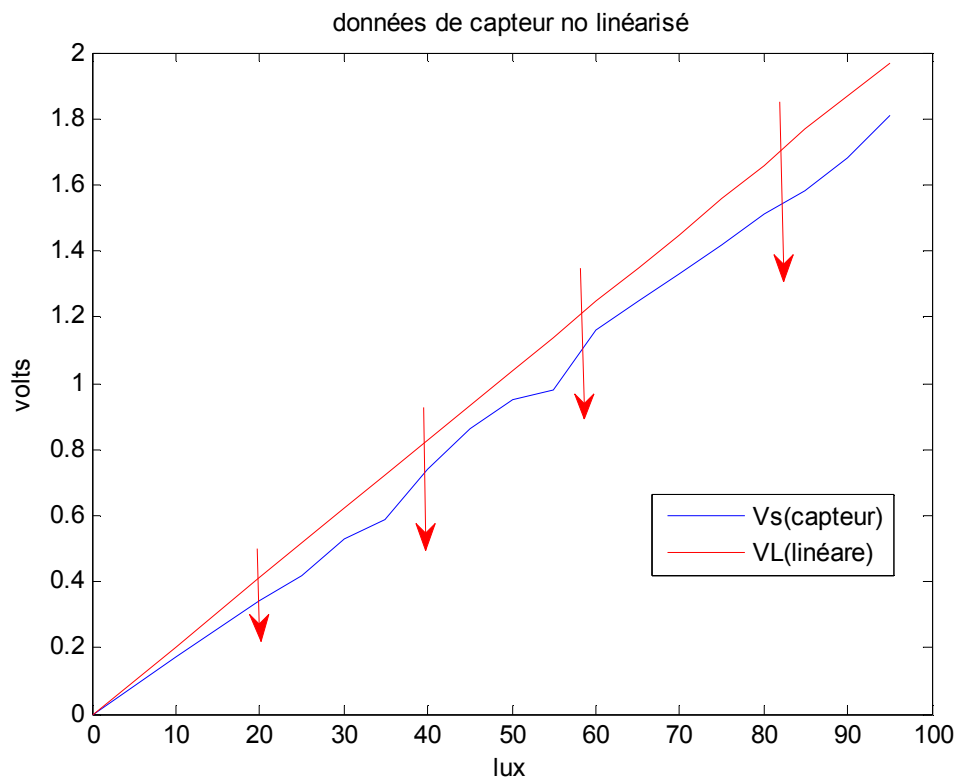


Fig.III.4.4.2 variation de la tension de sortie du capteur en fonction de luminosité.

Chapitre III Etude et Réalisation du système

III.4.5 Capteur de l'humidité :

le capteur d'humidité le plus le plus répandue dans les stations météo c'est le capteur qui port la référence RTC 6919001 appelé aussi capteur VALVO, se présente sous forme d'un boitier de matière plastique comportant de nombreuses perforations permettant à l'air ambiant de pénétrer à l'intérieur .

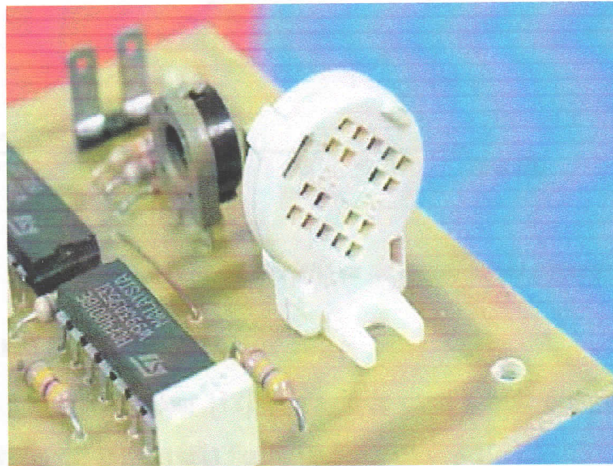


Fig. III.4.5. à le composant RTC 6919001

Ses caractéristiques métrologiques qui sont présenté par la suite ainsi que leur constitution consistions internes, qu'il s'agit d'un condensateur dont les armatures ne sans rien de moins que des feuilles d'or très fines, se qui rendre ce capteur plus puissant et plus cher.

III .4.5.1 Caractéristiques métrologique :

frequence de fonctionnement : **4KHZ à 1MHZ** ,

influence de temperature :**0.1 % K**

Capacité : **122pF +/- 15% (T=25 °c,Hrel=43%, F=100KHZ).**

Sevsibilité : **(0.4 +/- 0.05%)pF à Hrel=43%.**

Tention d'alimentation maximale : **+15 V**

La capacité de capteur dépend de l'humidité (humidité relative Hrel). La gamme de cette humidité est comprise entre 10% et %90.il suffit d'observer la courbe donné en

Chapitre III Etude et Réalisation du système

(Fig.-III.4.5.b) afin de voir la capacité leu en sortie du capteur en fonction de l'humidité relative (la précision nominale est de 5%).

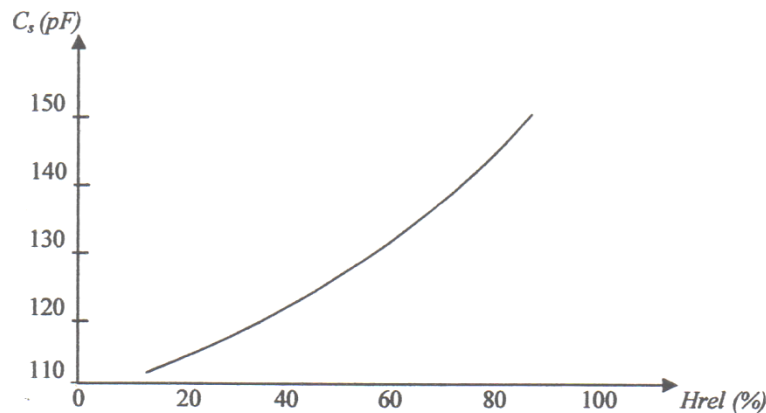


Fig.III.4.5.b : caractéristique de transfert du capteur d'humidité

III.5 La carte arduino-uno [12] :

Un module Arduino est généralement construit autour d'un microcontrôleur ATMEL AVR (Atmega328), et de composants complémentaires qui facilitent la programmation et l'interfaçage avec d'autres circuits. Chaque module possède au moins un régulateur linéaire 5V et un oscillateur à quartz 16 MHz (ou un résonateur céramique dans certains modèles). Le microcontrôleur est pré-programmé avec un boot loader de façon à ce qu'un programmeur dédié ne soit pas nécessaire.

Les modules sont programmés au travers une connexion série RS-232, mais les connexions permettant cette programmation diffèrent selon les modèles. Les premiers Arduino possédaient un port série, puis l'USB est apparu sur les modèles uno, tandis que certains modules destinés à une utilisation portable se sont affranchis de l'interface de programmation, relocalisée sur un module USB-série dédié (sous forme de carte ou de câble).

L'Arduino utilise la plupart des entrées/sorties du microcontrôleur pour l'interfaçage avec les autres circuits. Le modèle UNO, possède 14 entrées/sorties numériques (0 à 13), dont 6 peuvent produire des signaux PWM, et 6 entrées analogiques (A0 à A5). Les connexions sont établies au travers de connecteurs femelle HE14 situés sur le dessus de la carte,

Chapitre III Etude et Réalisation du système

les modules d'extension venant s'empiler sur l'Arduino. Plusieurs sortes d'extensions sont disponibles dans le commerce.

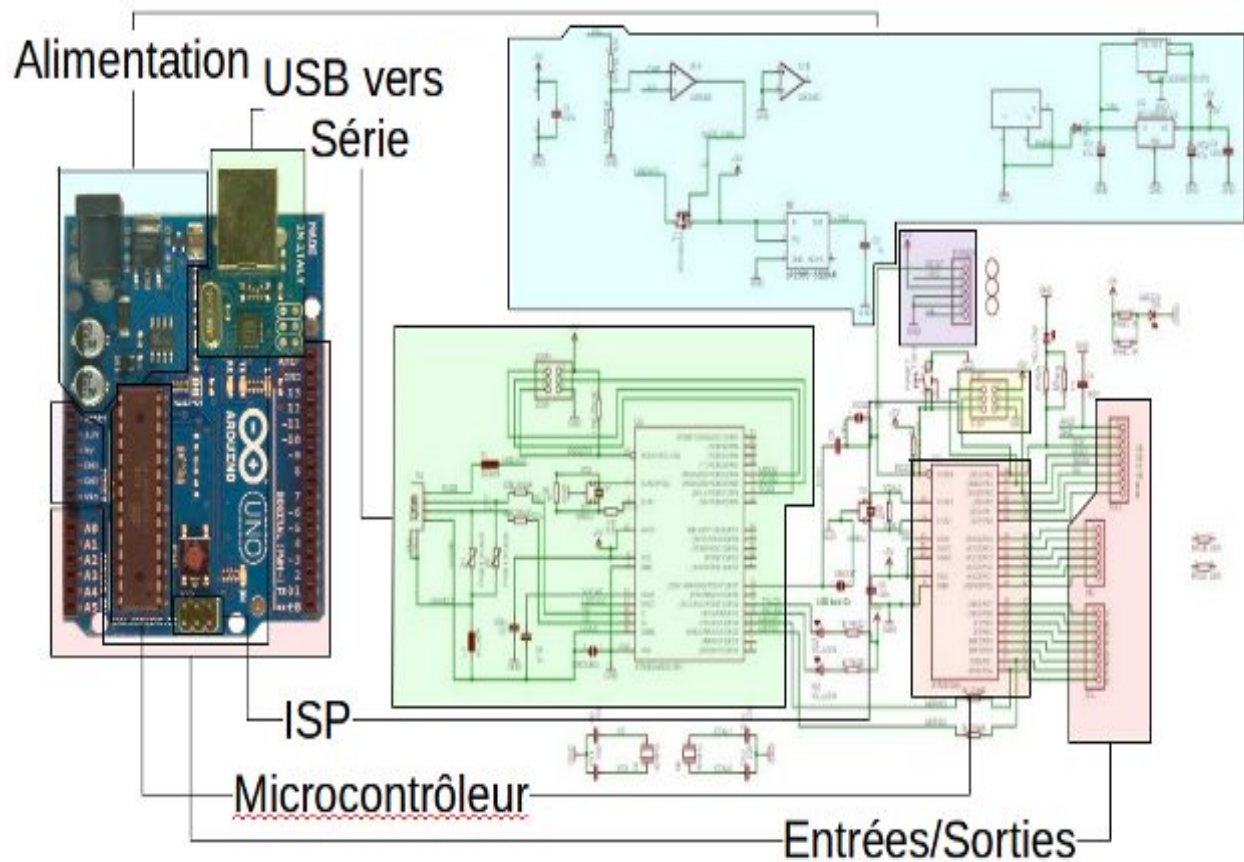


Fig.III.4 Schéma simplifié de la carte Arduino - UNO

Chapitre III Etude et Réalisation du système

III.6 Afficheur a cristaux liquides (LCD : Liquid Crystal Display)[9] :

Les afficheurs à cristaux liquides (Fig.III.6) sont des modules compacts

Et intelligents et nécessitent peu de composants externes pour un bon fonctionnement.

Un exceptionnel microprocesseur « pilote » de la famille C-MOS diminue

considérablement leur consommation (inférieur à 0.1 MW). Ils sont pratiquement les seuls à être utilisés sur les appareils à alimentation par piles.



Fig.III.6 vue de face d'un afficheur « LCD »

Plusieurs afficheurs sont disponibles sur le marché et ne diffèrent les uns des autres, non seulement par leurs dimensions, (de 1 à 2 lignes de 8 à 16 caractères ou de 1 à 4 lignes de 6 à 80 caractères), mais aussi par leurs caractéristiques techniques et leurs tensions de service. Certains sont dotés d'un rétro éclairage de l'affichage. Cette fonction fait appel à des LED montées derrière l'écran du module, cependant, cet éclairage est gourmand en intensité (250 mA max).

III.6.1 principe de fonctionnement :

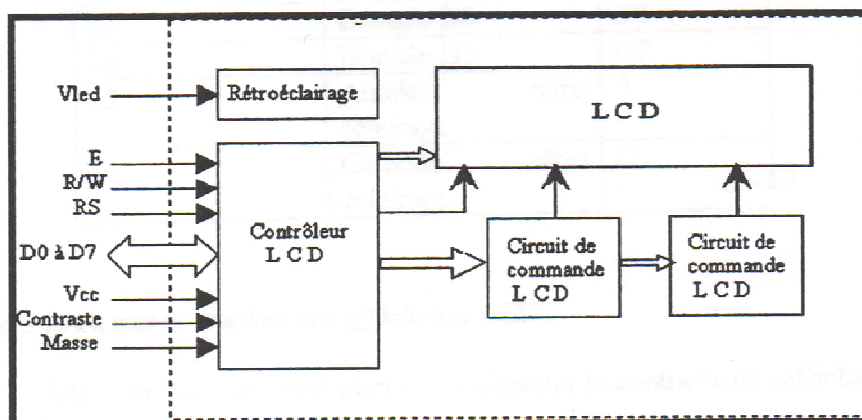


Fig.III.6.1 schéma fonctionnel d'un afficheur LCD

Chapitre III Etude et Réalisation du système

Comme montre le schéma fonctionnel, l'affichage comporte d'autres composants que l'afficheur à cristaux liquides (LCD) seul. Un circuit intégré de commande spécialisé, le **LCD-Controller**, est chargé de la gestion du module. Le « contrôleur » remplit une double fonction : d'une part il commande l'affichage et de l'autre se charge de la communication avec l'extérieur.

III.6.2 Brochage des afficheurs LCD :

Plusieurs modèles d'afficheurs de différentes marques sont compatibles avec cette interface. En effet, les afficheurs LCD disposent d'une interface parallèle normalisée répondant au tableau suivant :

Numéro de broche de l'afficheur LCD	Fonction	appellation
1	mass	VSS
2	+5 volts	VDD
3	contraste	V0
4	Commande/ Données	RS
5	Lecture / Ecriture	R/W
6	Validation LCD	E
7	Donnée 0	D0
8	Donnée 1	D1
9	Donnée 2	D2
10	Donnée 3	D3
11	Donnée 4	D4
12	Donnée 5	D5
13	Donnée 6	D6
14	Donnée 7	D7
15	Anode rétro éclairage	A
16	Cathode rétro éclairage	C

Chapitre III Etude et Réalisation du système

a. Rôle des principales broches des afficheurs LCD :

- **V0** : potentiel continue permettant d'ajuster le contraste de l'affichage.
- **RS** : l'état logique de cette ligne définit la nature des informations envoyées à l'afficheur. Si RS=1 on envoie une donnée. Si RS=0 on envoie une commande.
- **R/W** : permet de lire ou écrire au sein de la RAM de l'afficheur.
- **E** : validation de l'information transmise à l'afficheur.
- **D0 à D7** : 8 lignes de données bidirectionnelles (selon la position de R/W).
- **A** et **C** : ces deux broches supplémentaires sont disponibles uniquement à rétroéclairage de l'afficheur LCD.

Remarque : les afficheurs à logique intégrée peuvent être pilotés en mode 8 bits (un octet) ou en mode 4 bits (un quart) par transfert multiplexé sur les bits **D4 à D7** du LCD

b. Le jeu de commandes standard [10] :

Instructions	Code										Description	Durée
	RS	R/W	D7	D6	D5	D4	D3	D2	D1	D0		
Clear Display	0	0	0	0	0	0	0	0	0	1	Efface l'ensemble de la mémoire de données sans toucher au générateur de caractères. Ramène le curseur en position « home », à l'adresse 00.	1,64 ms
Return home	0	0	0	0	0	0	0	0	1	X	Ramène le curseur en position « home », à l'adresse 00. Si l'affichage était décalé, il est remis à sa position d'origine : l'adresse 00 se trouve à nouveau en haut à gauche.	1,64 ms
Entry mode set	0	0	0	0	0	0	0	1	I/D	S	Définit le sens de déplacement du curseur après l'apparition d'un caractère (vers la gauche si I/D=1, vers la droite si I/D=0) et si l'affichage accompagne le curseur dans son déplacement ou non (S).	40 µs

Chapitre III Etude et Réalisation du système

Display on/off control	0	0	0	0	0	0	1	D	C	B	Met l'affichage en ou hors fonction l'affichage (D), le curseur (C), le clignotement du curseur (B).	40 μ s
Cursor and display shift	0	0	0	0	0	1	S/C	R/L	X	X	Déplace le curseur (S/C=1) ou l'affichage (S/C=0) d'une position vers la gauche (R/L=1) ou la droite (R/L=0) sans changer la DD RAM.	40 μ s
Function set	0	0	0	0	1	DL	N	F	X	X	Définit la taille de l'interface (DL=0 pour mode 4 bits, DL=1 pour mode 8 bits), le nombre de lignes (NL=0 pour 1 ligne, N=1 pour 2 ou 4 lignes), et la taille des fontes (F=0 pour des caractères 5x7, F=1 pour des caractères 5x10).	40 μ s
Set CG RAM address	0	0	0	1	A5	A4	A3	A2	A1	A0	Définit l'adresse de la CG RAM. Les données de la CG RAM sont envoyées après cette commande.	40 μ s
Set DD RAM address	0	0	1	A6	A5	A4	A3	A2	A1	A0	Définit l'adresse de la DD RAM. Les données de la DD RAM sont envoyées après cette commande.	40 μ s
Read busy flag & address	0	1	BF	A6	A5	A4	A3	A2	A1	A0	Lit le flag busy (BF), et l'adresse de la position du curseur. BF vaut 0 si l'afficheur accepte une instruction, 1 s'il est occupé	1 μ s
Write data to CG or DD RAM	1	0	D7	D6	D5	D4	D3	D2	D1	D0	Ecrit des données dans la DD RAM ou la CG RAM.	40 μ s
Read data	1	1	D7	D6	D5	D4	D3	D2	D1	D0	Lit les données de la DD RAM ou de la CG RAM.	40 μ s

DD RAM : mémoire d'affichage, **CG RAM** : mémoire du générateur de caractères.

X : Indifférent.

I/D=0 : déplacement vers la gauche, **I/D=1** : déplacement vers la droite.

I/D=0 : déplacement vers la gauche, **I/D=1** : déplacement vers la droite.

S=1 : afficheur accompagne le curseur dans son déplacement.

Chapitre III Etude et Réalisation du système

S/C = 0 : le curseur se déplace,

S/C = 1 : l'afficheur se déplace.

R/L = 0 : vers la gauche,

R/L = 1 : vers la droite.

DL = 0 : interface 4 bits,

DL = 1 : interface 8 bits.

N = 0 : 1 ligne,

N = 1 : 2(ou 4) Lignes.

F = 0 : caractères 5x7,

F = 1 : caractères 5x10.

BF = 0 : accepte une instruction, **BF = 1** : occupé.

III.7 Le branchement de l'écran LCD avec l'arduino [7] :

L'afficheur LCD utilise 6 à 10 broches de données (D4 à D7 Fig.III.7 + RS + E) et deux d'alimentations (+5V et masse). La plupart des écrans possèdent aussi une entrée analogique pour régler le contraste des caractères. Nous brancherons dessus un potentiomètre de 10 kOhms. Les 10 broches de données peuvent être placées sur n'importe quelles entrées/sorties numériques de l'Arduino. En effet, nous indiquerons ensuite à la **librairie LiquidCrystal** qui est branché où.

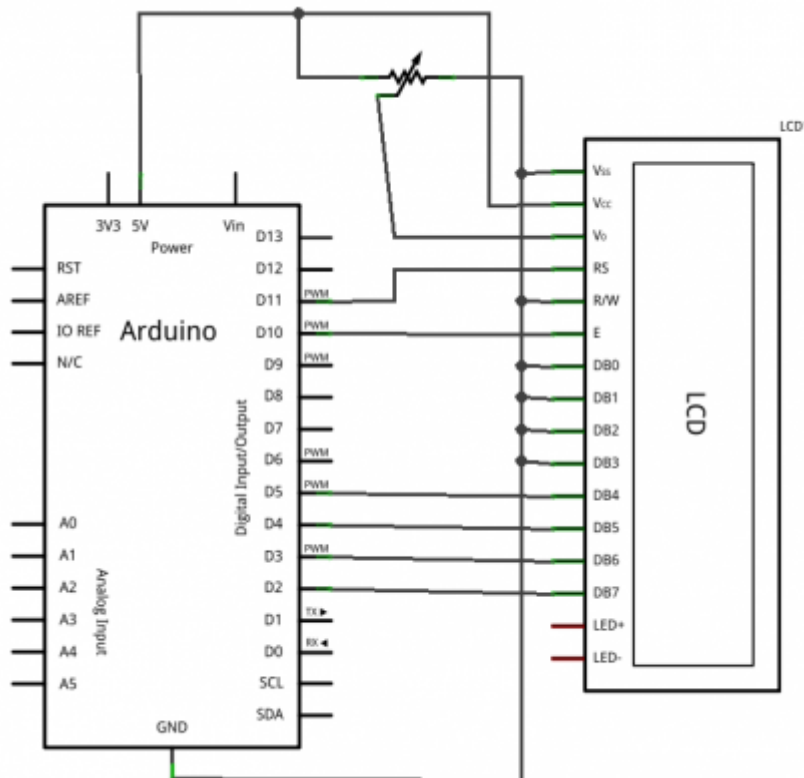


Fig.III.7 Le montage à 4 broches de données

III .7.1 Le démarrage de l'écran LCD avec Arduino :

Comme écrit plus tôt, nous allons utiliser la librairie "**LiquidCrystal**". Pour l'intégrer c'est très simple, il suffit de cliquer sur le menu "Import Library" et d'aller chercher la bonne. Une ligne **#include "LiquidCrystal.h"** doit apparaitre en haut de la page de code ,Ensuite, il ne nous reste plus qu'à dire à notre carte Arduino où est branché l'écran (sur quelles broches) et quelle est la taille de ce dernier (nombre de lignes et de colonnes). Nous allons donc commencer par déclarer un objet **lcd**, de type **LiquidCrystal** et qui sera global à notre projet. La déclaration de cette variable possède plusieurs formes :

- Liquid Crystal(rs, enable, d4, d5, d6, d7) où rs est le numéro de la broche où est branché "RS", "**enable**" est la broche "E" et ainsi de suite pour les données.

Ensuite, dans le **setup ()** il nous faut démarrer l'écran en spécifiant son nombre de **colonnes** puis de **lignes**. Cela se fait grâce à la fonction **begin (cols, rows)**.

Chapitre III Etude et Réalisation du système

III .7.2 Ecriture de texte et Gérer le curseur sur l'écran LCD avec l'Arduino :

Voici maintenant d'autres fonctions que vous attendez certainement, celles permettant de déplacer le curseur sur l'écran. En déplaçant le curseur, vous pourrez écrire à n'importe quel endroit sur l'écran par la fonction « **LCD.setCursor ()** » et la fonction **LCD.print ()** : pour écrire un texte ou caractères ou variable.

III .7.2.1 Organisé le repère de l'écran LCD avec l'arduino :

C'est assez simple, mais il faut être vigilant quand même. Tout d'abord, sachez que les coordonnées s'expriment de la manière suivante **(x, y)**. **x** représente les abscisses, donc les pixels horizontaux et **y** les ordonnées, les pixels verticaux. L'origine du repère sera logiquement le pixel le plus en haut à gauche (comme la lecture classique d'un livre, on commence en haut à gauche) et à pour coordonnées ... (0,0) Eh oui, on ne commence pas aux pixels (1,1) mais bien (0,0). Quand on y réfléchit, c'est assez logique. Les caractères sont rangés dans des chaînes de caractères, donc des tableaux, qui eux sont adressés à partir de la case 0. Il paraît donc au final logique que les développeurs aient gardé une cohérence entre les deux. Puisque nous commençons à 0, un écran de **16x2** caractères pourra donc avoir comme coordonnées de 0 à 15 pour **x** et 0 ou 1 pour **y**. Ceci étant dit, nous pouvons passer à la suite. La prochaine fonction que nous allons voir prend directement en compte ce que je viens de vous dire. Cette fonction nommée **setCursor ()** vous permet de positionner le curseur sur l'écran. On pourra donc faire **setCursor(0,0)** pour se placer en haut à gauche (équivalent à la fonction "**home ()**") et en faisant **setCursor(15,1)** on se placera tout en bas à droite (toujours pour un écran de 16x2 caractères).

Un exemple du programme arduino :

```
#include <LiquidCrystal.h>
// initialise l'écran avec les bonnes broches
LiquidCrystal lcd(8,9,4,5,6,7);
void setup()
{
  lcd.begin(16, 2);
  lcd.setCursor(2,1); //place le curseur aux coordonnées (2,1)
  lcd.print("Texte centré"); //texte centré sur la ligne 2
}
```

Chapitre III Etude et Réalisation du système

III.8 La communication entre le PC et la cartes Arduino [8] :

Le but de ce montage est de communiquer entre l'ordinateur et l'Arduino par la liaison série. Cette liaison série passe par le câble USB. Savoir utiliser cette connexion est une très bonne base à la réalisation d'autres connexions à l'aide de module qui utilise le même mode de fonctionnement comme le module Bluetooth par exemple.

Et le Composants nécessaires : **Un Arduino, Un câble USB.(fig.III.8)**

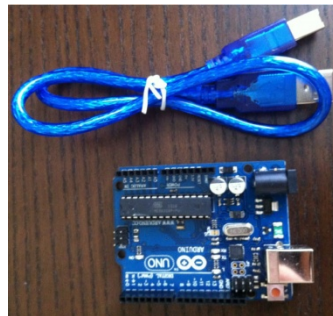


fig.III.8 arduino + Un câble USB

III.8.1 Principe du montage :

Ce montage ne comporte aucun composant externe, il suffit de câbler l'Arduino avec le PC via le câble USB. Comme la figure ci- dessous (Fig.III.8.1)

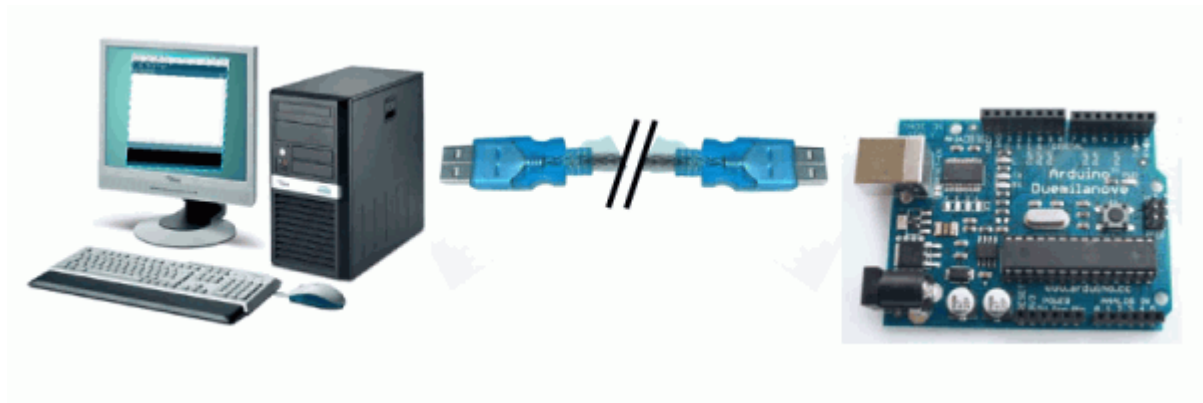


Fig.III.7.1 La communication entre le PC et la cartes Arduino

Chapitre III Etude et Réalisation du système

III.8.2 Programmation du montage :

La majeure partie de ce montage est la programmation. Pour établir une liaison série, on utilise la librairie Serial. Comme c'est une librairie, pour utiliser ces fonctions, il faudra écrire « **Serial.nom de la fonction** ». Elle contient les fonctions suivantes :

- **begin ()** : Fixe le débit de communication en bits par secondes (l'unité est le baud) pour la communication série. Pour communiquer avec l'ordinateur, utiliser l'un de ces débits : 300, 1200, 2400, 4800, 9600, 14400, 19200, 28800, 38400, 57600, or 115200. Dans notre exemple nous utiliserons 9600 bauds.
- **available()** : Donne le nombre d'octets (caractères) disponible pour lecture dans la file d'attente (buffer) du port série. Cette fonction est utile pour savoir si l'on a un message sur la liaison série.
- **read()** : Lit les données entrantes sur le port Série.
- **flush ()** : Vide le buffer de réception de données du port série. Par conséquent, tout appel de la fonction **Serial.read ()** ou **Serial.available ()** renverra seulement les données reçues après le plus récent appel de la fonction **Serial.flush ()**.
- **Print ()** : Cette fonction affiche le texte compris comme paramètre
- **Println ()** : Identique à print () mais sur une nouvelle ligne
- **Write ()** : Ecrit des données binaires sur le port série. Ces données sont envoyées comme une série d'octet; pour envoyer les caractères correspondants aux chiffres d'un nombre, utiliser plutôt la fonction **print ()**.

Exemple de programme :

Le programme doit établir une connexion avec le PC et dire « Avez-vous quelques choses à me dire ? » et lorsque le PC répond. L'Arduino répète ce que le PC a dit puis dit « OK ».

Le programme est :

Chapitre III Etude et Réalisation du système

```
const int debit = 9600; // On défini le débit dans une variable

byte texte; // Variable pour contenir le texte reçu

void setup () { // Fonction d'initialisation de la carte
Serial.begin(debit); // On initialise la liaison série
Serial.println ("Arduino : Avez-vous quelques choses a me dire ?"); // On demande si
l'utilisateur à une question
}

void loop() // Fonction principale, elle se répète (s'exécute) à l'infini
{
while (Serial.available()) // On attend des messages sur la liaison série
{
while (Serial.available()) // On attend des messages sur la liaison série
{
texte = Serial.read(); // Stockage des messages dans texte
Serial.write(texte); // Réécrire le message
delay(10);
}
Serial.println("\nArduino : OK"); // Nouveau message pour valider la réception
Serial.println("Arduino : Avez-vous quelques choses a me dire ?"); // On demande si
l'utilisateur à une question
}
}
```

Il reste à brancher l'Arduino pour compiler le programme et le téléverser.

Une fois terminé cela nous donne comme la figure ci- dessous (**Fig.III.8.2**):



Fig.III.7.2 Physiquement

Pour aller plus loin : On peut faire des tests avec les fonctions **print**, **println** au lieu de la fonction **write** pour voir les messages en binaire

Chapitre III Etude et Réalisation du système

III.9 Carte de puissance :

Après avoir fait l'acquisition et le traitement de nos données à savoir celle de la température, l'humidité, la luminosité la vitesse et la direction de vent, on établira alors la commande nécessaire pour contrôler notre serre, en activant les actionneurs. Résultant de notre traitement cela se fera à travers une carte de puissance (**fig.III.9**).

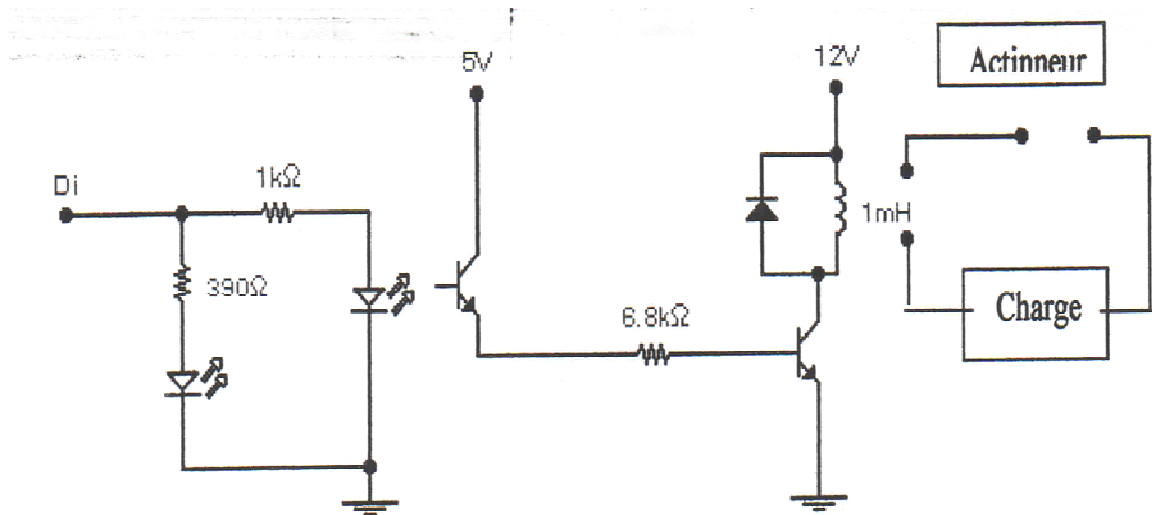


fig.III.9 la chaine de commande

La carte de puissance reçoit des commandes sous forme de tension TTL elle sera converti en tension adéquat. Elle est faite à base opto-coupleurs permettant d'isoler« galvaniquement » la carte de traitement et de commande (**arduino-uno**). Les relais sont utiliser pour assurer une isolation galvanique et une transmission de la puissance nécessaire à la charge, cette dernière représente n'importe quel actionneur. La présence d'un « 1 » logique au niveau du bit Di entraine conduction du transistor d'où l'activation des actionneurs, la désactivation de ces dernières se fait par le blocage du transistor à la suite de la présence d'un « 0 » logique au niveau de Di. Une diode de roue-libre permet d'inhiber les effets de self provoquée lors de l'ouverture du relais.

Chapitre III Etude et Réalisation du système

Conclusion :

Dans ce chapitre on vient de voir la représentation en détail de notre processus physique, l'étude du matériel électronique utilisé, les capteurs, la carte de commande et la carte de puissance. Nous verrons dans le chapitre qui suit le logiciel de gestion réalisé et les résultats obtenus.

IV.1 Introduction :

La partie soft de ce projet est la partie la plus importante du travail, puisque c'est elle qui gère les différentes fonctions à savoir :

- ❖ Le traitement des différents signaux issus des capteurs.
- ❖ Affichage sur l'afficheur LCD 2*16
- ❖ Commande des actionneurs.
- ❖ Communication entre la carte **arduino UNO** et le **PC**.

Et pour ce là et avant de commencer pour écrire le programme qui fait tout ça, on a partagé et organisé la tâche dans des organigrammes.

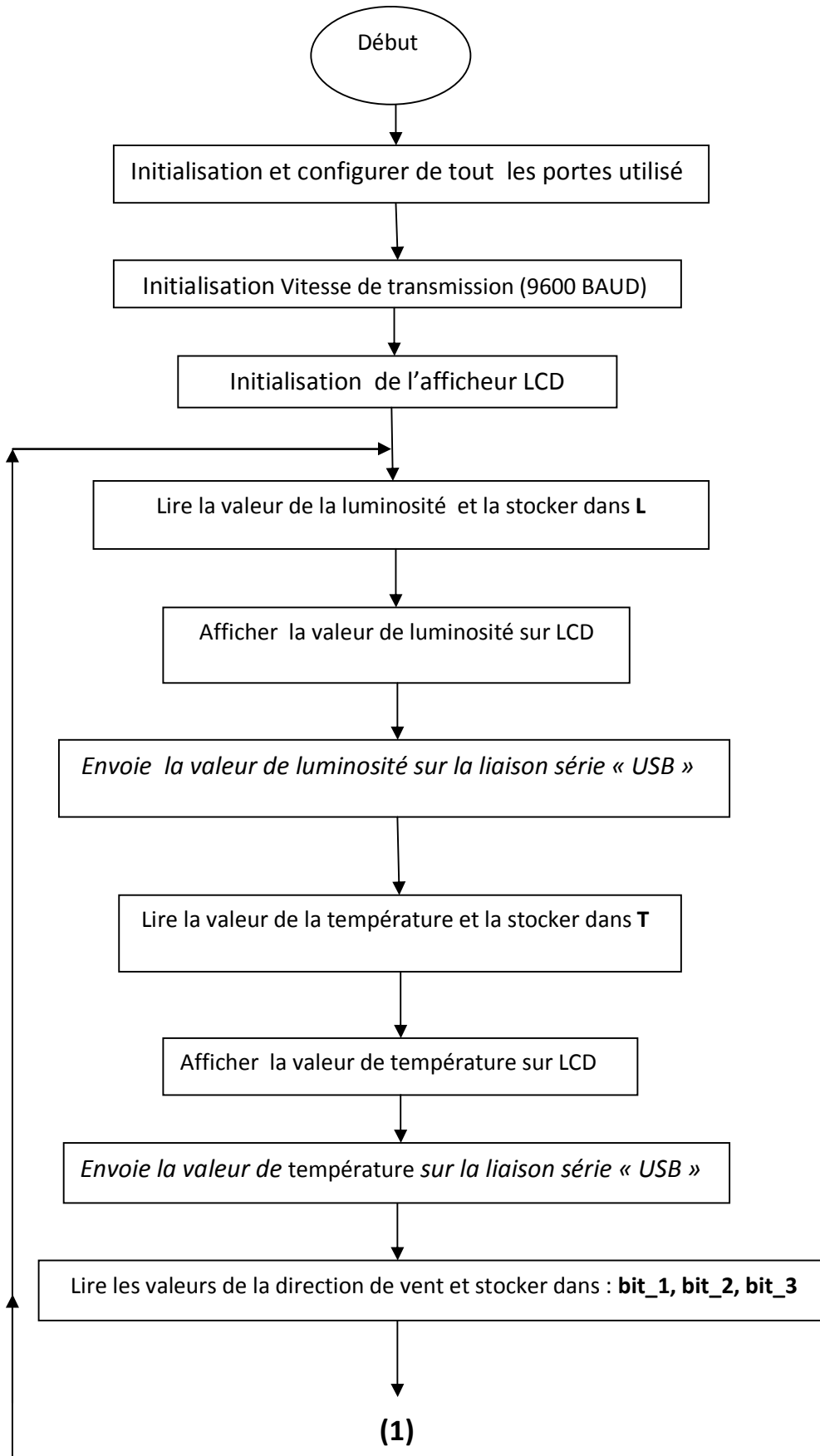
IV.2 organigramme général du programme principal :

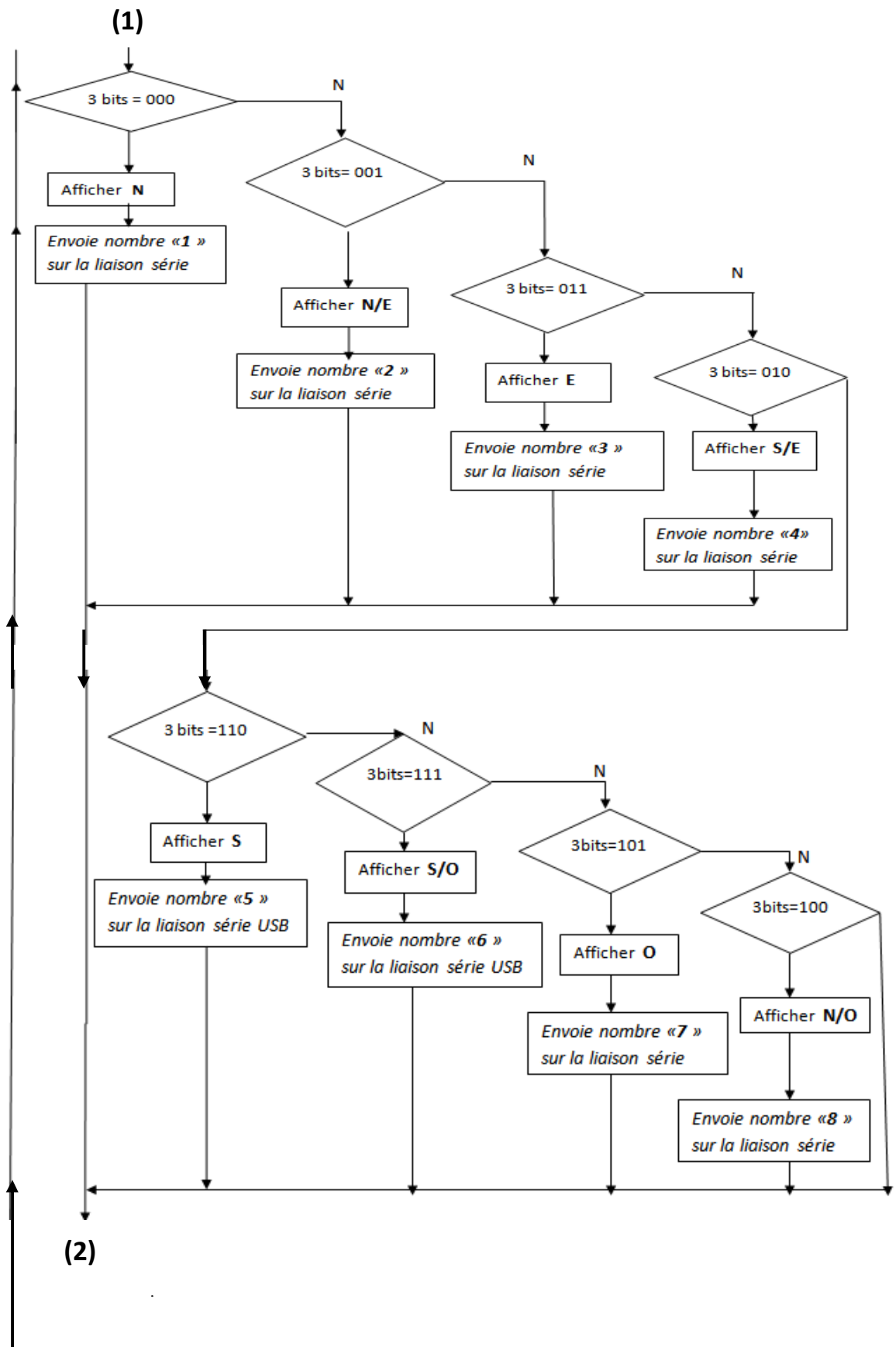
L'organigramme général du programme principal, est constitué d'une partie d'initialisation de la carte arduino dont on initialise tous les broches analogiques utilisera, en configurant toutes les ports en entrée ou en sortie, la deuxième c'est la partie de présentation de l'interface de l'afficheur, qui nous permettra de fixer les lettres et les caractères inchangeable, et la troisième partie c'est elle qui fait la liaison entre les différents sous programmes tel que :

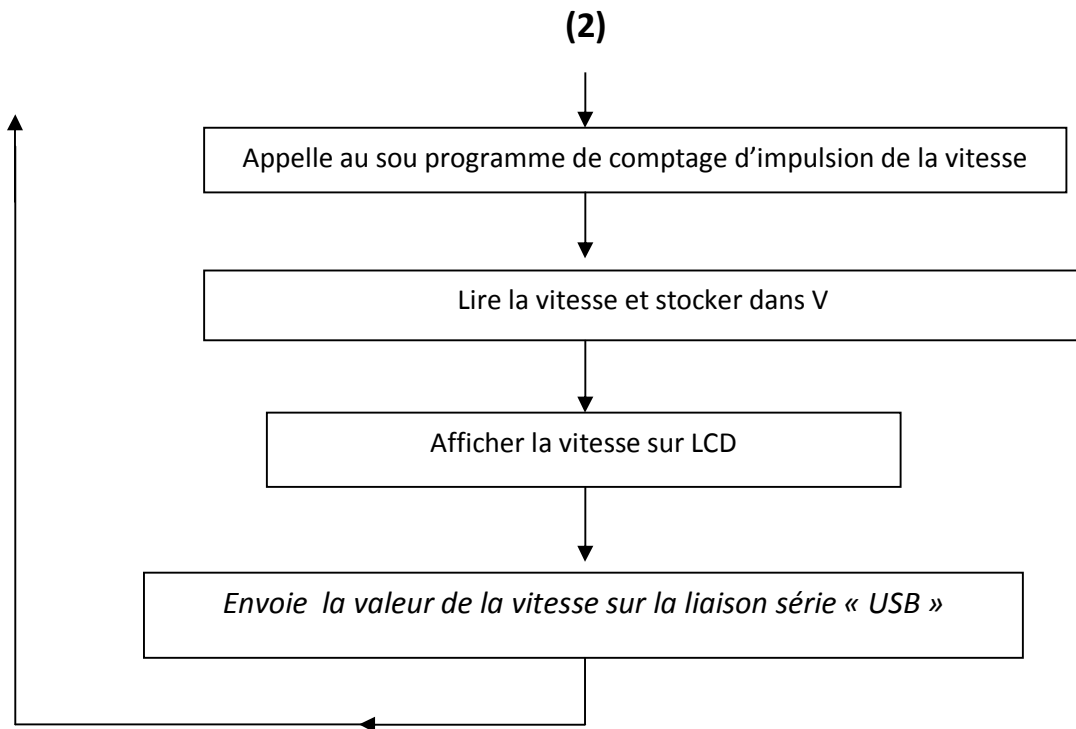
Le convertisseur **A/N** de l'**arduino** il lit la tension de capture de température **LM35** dans la broche analogique **A1** et stocke dans **T** et lit la tension de luminosité de **LDR** dans la broche **A0** et convertit en **LUX** et stocke dans **L**

Et le sous programme de vitesse qui compte le nombre d'impulsion que génère à l'interruption externe **n°0** (broche numérique **2**) pendant 1s, divise ce nombre par 10, puis il charge le résultat dans la variable **comptageImpuls** qui mémorise la valeur d'incrémenter le comptage d'impulsion puis convertit en vitesse.

Le programme de direction qu'il affiche directement la direction

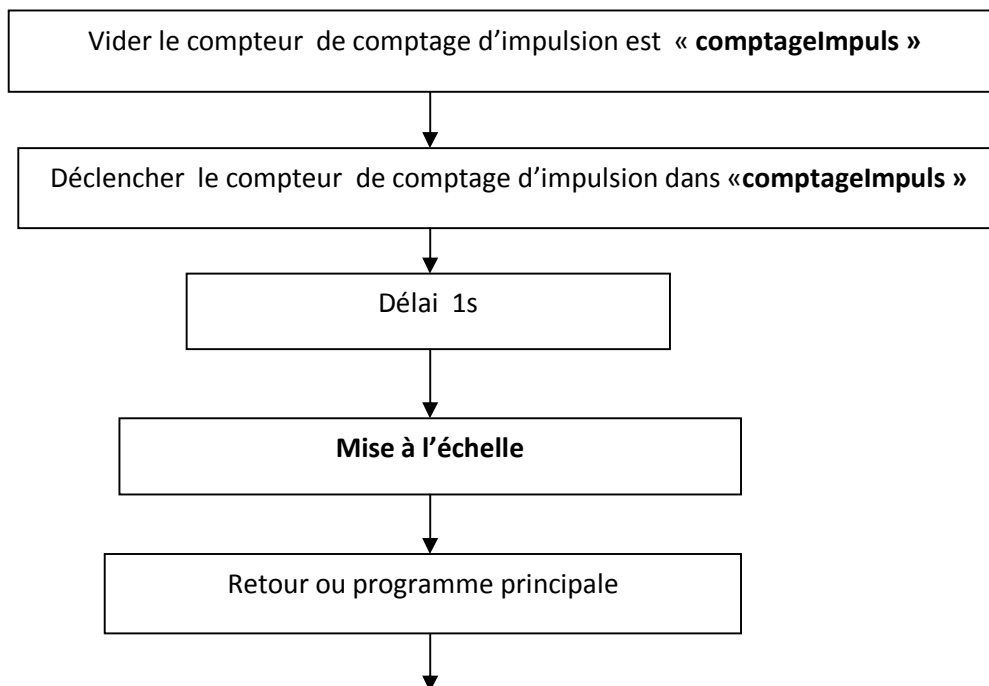






Org.IV.2 organigramme général du programme principal

IV.2.2 organigramme de sous programme de la vitesse :



IV.3 le langage de programmation utilisée :

Le développement de notre application est basé sur deux langages, le premier en logiciel Arduino qui est basé sur le langage C pour la programmation de la carte Arduino-uno d'acquisition et de commande et le deuxième en langage évolué « **matlab GUID** » pour la conception du logiciel de gestion.

IV.3.1 Présentation de l'Espace de développement Intégré (EDI) Arduino :

IV.3.1.1 Description de l'interface :

Le logiciel **Arduino** a pour fonctions principales :

- de pouvoir écrire et compiler des programmes pour la carte **Arduino**
- de se connecter avec la carte **Arduino** pour y transférer les programmes
- de communiquer avec la carte **Arduino**

Cet espace de développement intégré (EDI **fig. IV.3.1.1.a**) dédié au langage **Arduino** et à la programmation des cartes **Arduino** comporte :

- une **BARRE DE MENUS** comme pour tout logiciel une interface graphique (GUI),
- une **BARRE DE BOUTONS** qui donne un accès direct aux fonctions essentielles du logiciel et fait toute sa simplicité d'utilisation,
- un **EDITEUR** (à coloration syntaxique) pour écrire le code de vos programmes, avec onglets de navigation,
- une **ZONE DE MESSAGES** qui affiche l'état des actions en cours,
- une **CONSOLE TEXTE** qui affiche les messages concernant le résultat de la compilation du programme

- Un **TERMINAL SERIE** (fenêtre séparée **fig. IV.3.1.1.b**) qui permet d'afficher des messages textes reçus de la carte Arduino et d'envoyer des caractères vers la carte Arduino. Cette fonctionnalité permet une mise au point facilitée des programmes, permettant d'afficher sur l'ordinateur l'état de variables, de résultats de calculs ou de conversions analogique-numérique : un élément essentiel pour améliorer, tester et corriger ses programmes.

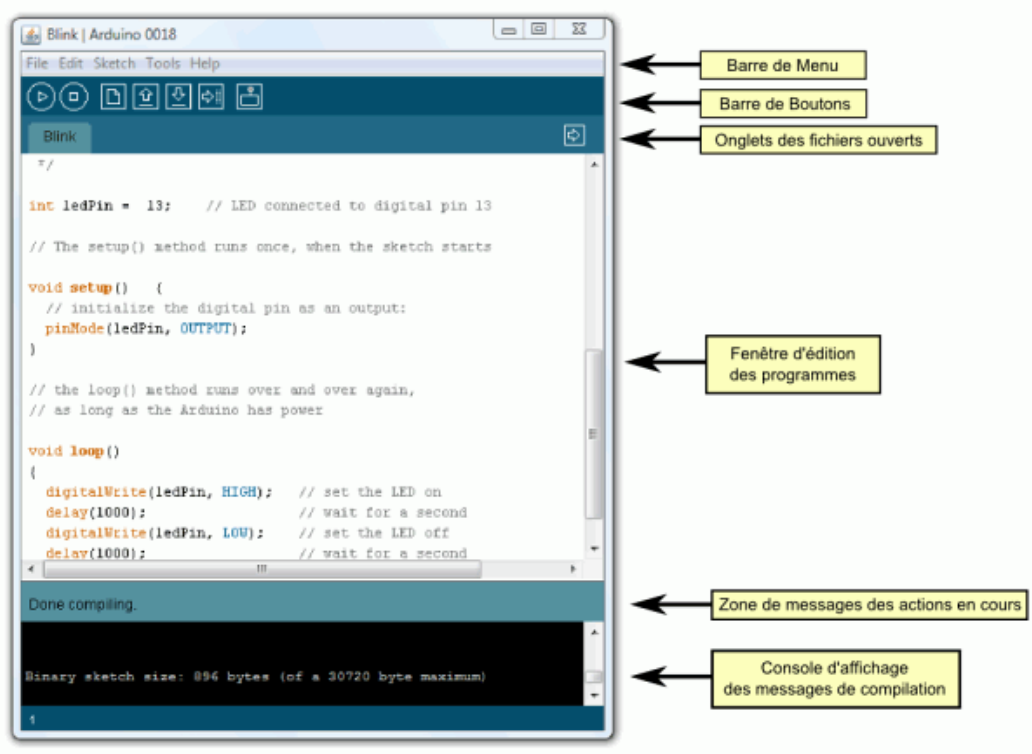


Fig. IV.3.1.1.a: présentation des éléments de l'ARDUINO software

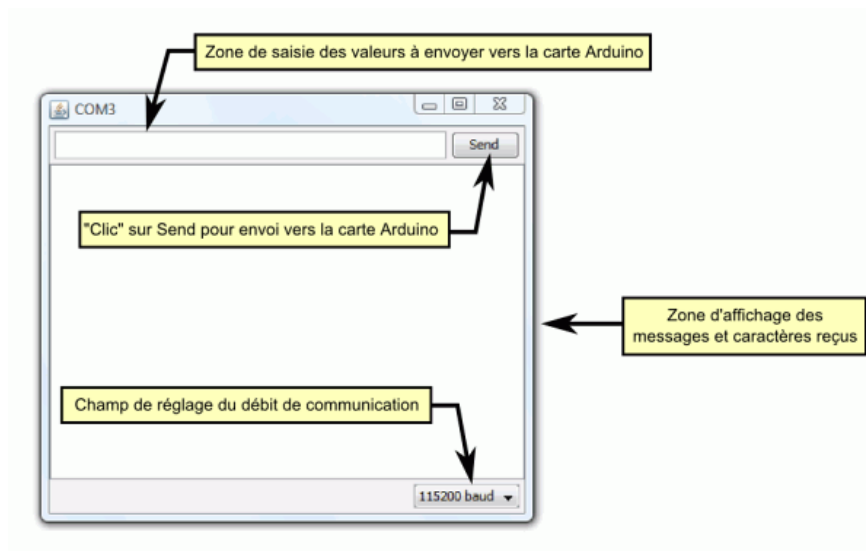



Fig. IV.3.1.1.b: module TERMINAL SERIE

IV.3.1.2 Description de la barre des boutons (fig. IV.3.1.2)

	Vérifier/compiler : Vérifie le code à la recherche d'erreur.
	Stop : Stoppe le moniteur série ou les autres boutons activés.
	Nouveau : Crée un nouveau code (ouvre une fenêtre d'édition vide)
	Ouvrir : Ouvre la liste de tous les programmes dans votre "livre de programmes". Cliquer sur l'un des programmes l'ouvre dans la fenêtre courante.
	Note: en raison d'un bug dans Java, ce menu ne défile pas. Si vous avez besoin d'ouvrir un programme loin dans la List, utiliser plutôt le menu File > Sketchbook .
	Sauver : Enregistre votre programme.
	Transférer vers la carte : Compile votre code et le transfère vers la carte Arduino. Voir ci-dessous "Transférer les programmes" pour les détails.
	Moniteur Série : Ouvre la fenêtre du moniteur (ou terminal) série.

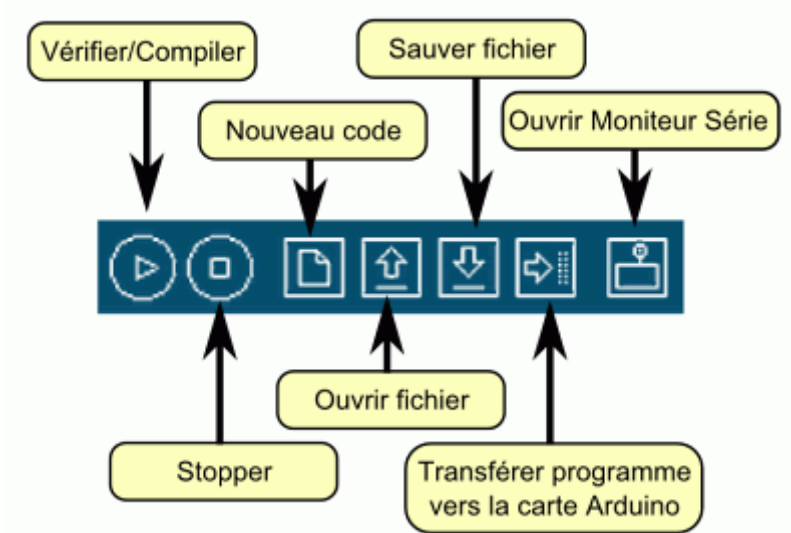


Fig. IV.3.1.2 Description de la barre des boutons

IV.3.1.3 Description de la structure d'un programme :

Un programme utilisateur Arduino est une suite d'instructions élémentaires sous forme textuelle, ligne par ligne. La carte lit puis effectue les instructions les unes après les autres, dans l'ordre défini par les lignes de code, comme mors d'une programmation classique.

Cette structure se décompose en trois parties (fig. IV.3.1.3.1) :

- Description des constantes et variables du programme
- Fonction principale : configuration des entrées/sorties et éléments à configurer (cette partie ne sera exécutée qu'une seule fois) dans la partie **VOID SETUP ()**
- Fonction boucle : description du fonctionnement général du programme (gestion des interactions entre les entrées/sorties) dans la partie **VOID LOOP ()**

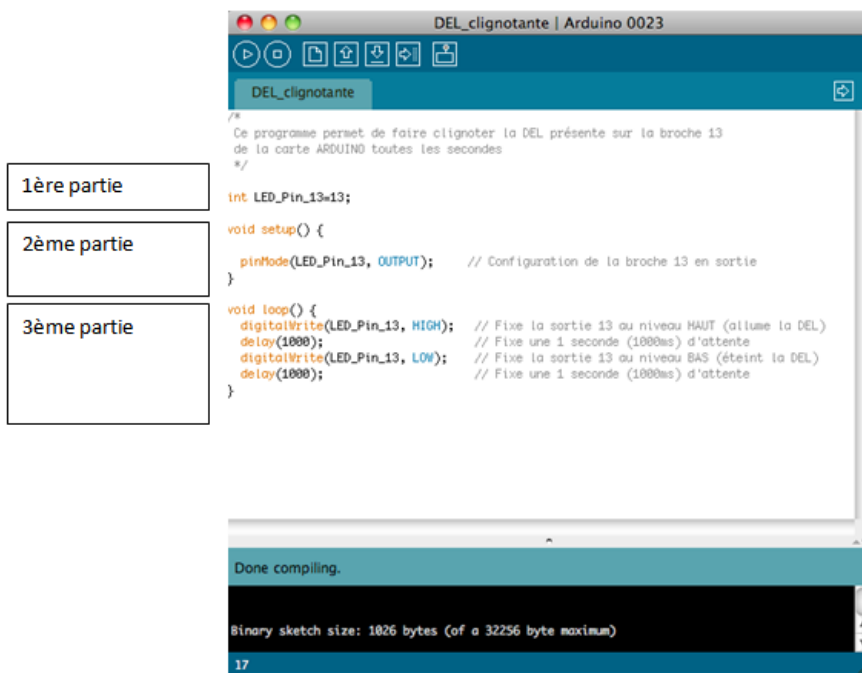


Fig. IV.3.1.3.1: fenêtre graphique de l'EDI

Remarque :

Il est possible d'ajouter des commentaires au programme. Pour cela on peut procéder de deux manière :

- à la fin de la ligne en ajoutant « // »
- en encadrant les commentaires entre « /* » et « */ »

IV.3.1.3.2. Description détaillée des parties :

a). Définition des variables et constantes : (1ère partie)

Dans cette partie, on déclare des éléments utilisés tout au long du programme : les constantes (statiques) et les variables (dynamiques). Ce sont des emplacements mémoire utilisés pour stocker des données (des valeurs) utilisables dans la suite du programme.

Variable : Une variable peut aussi bien représenter des données lues ou envoyées sur un des ports analogiques ou numériques une étape de calcul pour associer ou traiter des données, que le numéro « physique » de ces entrées ou sorties. Une “variable” n’est donc pas exclusivement un paramètre variant dans le programme. On la déclare de la façon suivante :

```
TYPE_DE_LA_DONNEE          NOM_DE_LA_DONNEE
```

Exemple : `int led`

Constante : Une constante est une variable dont la valeur est inchangeable lors de l’exécution d’un programme. On la déclare de la façon suivante :

```
CONST TYPE_DE_LA_DONNEE          NOM_DE_LA_DONNEE
```

Exemple : `const int led`

b). Fonction principale : `void setup ()` (2ème partie)

Cette fonction n’est exécutée qu’une seule fois au démarrage du programme. Elle permet la configuration des entrées et sorties de la carte. Les broches numériques de l’Arduino peuvent aussi bien être configurées en entrées qu’en sorties. Ici on a configuré `LED_Pin_13` en sortie.

`pinMode (nom, état)` est une des quatre fonctions relatives aux entrées et sorties numériques que nous verrons plus bas.

```
void setup ()
{
ici se trouve la configuration des entrées et des sorties
}
```


c). Fonction boucle : void loop() (3 ème partie)

Cette fonction loop() (boucle en anglais) fait exactement ce que son nom suggère et s'exécute en boucle sans fin, permettant à votre programme de s'exécuter. Dans cette boucle, on définit les opérations.

La fonction loop() est obligatoire, même vide, dans tout programme.

```
void loop()
{
  ici se trouve la description générale du programme en boucle
}
```

IV.3.1.3.3 Compilation et programmation de l'ARDUINO :

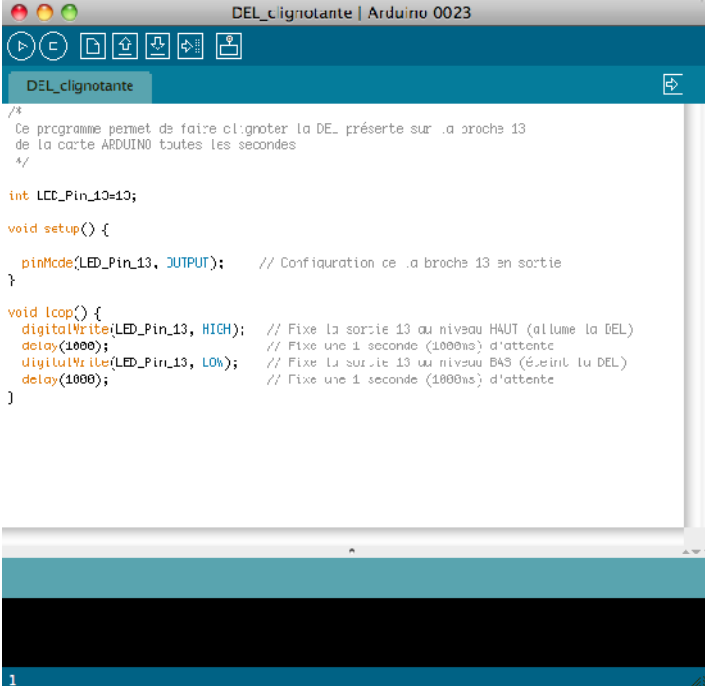
L'écriture d'un programme se déroule en plusieurs étapes.

1. Ecriture de l'algorithme : L'algorithme est une méthode pour résoudre un problème. L'algorithme est un moyen pour le programmeur de présenter son approche du problème à d'autres personnes.

2. Ecriture du programme :

La rédaction du programme se fait bien sur directement en rapport avec l'algorithme ci-dessus. Il faut absolument penser à mettre des commentaires compréhensifs par le non programmeur. Détaillé le programme et le partitionner en bloc logiques.

La rédaction du programme se fait dans la partie rayée ci- dessous (fig.2) :

The image shows a screenshot of the Arduino IDE interface. The window title is "DEL_cignotante | Arduino 0023". The code editor contains the following C++ code:

```
/*
  Ce programme permet de faire clignoter la DEL présente sur la broche 13
  de la carte ARDUINO toutes les secondes
  */
int LED_Pin_13=13;


void setup() {
  pinMode(LED_Pin_13, OUTPUT); // Configuration de la broche 13 en sortie
}

void loop() {
  digitalWrite(LED_Pin_13, HIGH); // Fixe la sortie 13 au niveau HAUT (allume la DEL)
  delay(1000); // Fixe une 1 seconde (1000ms) d'attente
  digitalWrite(LED_Pin_13, LOW); // Fixe la sortie 13 au niveau BAS (éteint la DEL)
  delay(1000); // Fixe une 1 seconde (1000ms) d'attente
}
```

The IDE interface includes a toolbar with icons for running, stopping, and saving, and a status bar at the bottom showing the number "1".

fig.2 La rédaction du programme

3. Compilation du programme : ans cette partie, on vérifie si le code contient des erreurs de syntaxes. En cas d'anomalie de compilation, le compilateur renseigne sur le type d'erreur et la ligne où elle se trouve.

Pour lancer la compilation, il faut appuyer sur . A ce moment-là, le bouton devient jaune et la zone de message affiche « Compiling » indiquant que la compilation est en cours.

Si la compilation se déroule sans erreur, le message « Done compiling » apparaît, suivi de la taille du programme

Un compilateur est un programme informatique qui traduit un langage (appelé le langage source) en un autre (le langage cible), généralement dans le but de créer un fichier exécutable.

Un compilateur sert le plus souvent à traduire un code source écrit dans un langage de programmation en un autre langage, habituellement un langage d'assemblage ou un langage machine.

Le programme en langage machine produit par un compilateur est appelé code objet.

4. Sélection de la cible et du port série : Avant de transférer le programme vers la carte Arduino, il faut, si ce n'est déjà fait, sélectionner la bonne carte Arduino (la bonne cible) depuis le menu **Tools>Board** (Outils>Carte (fig.4)).

La carte doit évidemment être connectée à l'ordinateur via un câble USB.

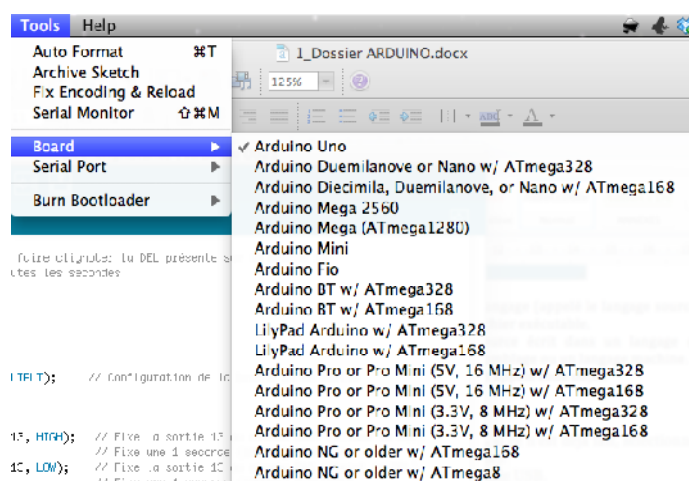



fig.4 Sélection de la cible et du port série

Vous devez également sélectionner le bon port série depuis le menu **Tools > Serial Port** (Outils > Port Série).

5. Transfert du programme vers la carte ARDUINO


Une fois que vous avez sélectionné le bon port série et la bonne carte Arduino, cliquez

sur le bouton **UPLOAD**  (Transfert vers la carte) dans la barre d'outils, ou bien sélectionner le menu **File>Upload to I/O board** (Fichier > Transférer vers la carte).

Sur la plupart des cartes, vous devez voir les LEDs des lignes RX et TX clignoter rapidement, témoignant que le programme est bien transféré. Durant le transfert, le bouton devient jaune et le logiciel **Arduino** affiche un message indiquant que le transfert est en cours.

IV.3.2 Guide matlab :

Pour avoir une interface conviviale pour l'utilisateur de notre produit. L'utilisateur du logiciel (**METEOROLOGICIEL**) associe à la carte d'acquisition et de commande pour visualiser toutes les données météorologiques sur le PC (graphes et tableaux), sauvegarder la base des données et gère la commande de l'application.

Le **GUIDE** est un constructeur d'interface graphique qui regroupe tous les outils (FIG. IV.3.2.a) dont le programmeur a besoin pour créer une interface graphique de façon intuitive. Il s'ouvre, soit en cliquant sur l'icône , soit en tapant **guide** dans le **Command Window de MATLAB**.

Le placement des objets est réalisé par sélection dans une boîte à outils. Leur mise en place et leur dimensionnement se font à l'aide de la souris.

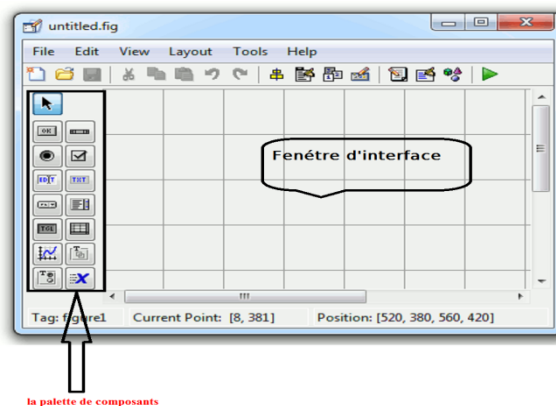


FIG. IV.3.2.a Fenêtre principale du GUIDE

Un double-clic sur un objet permet de faire apparaître le « **Property Inspector**» où les propriétés des objets (FIG. IV.3.2.b) sont facilement éditables. Leurs modifications et la visualisation de ces modifications sont immédiates.

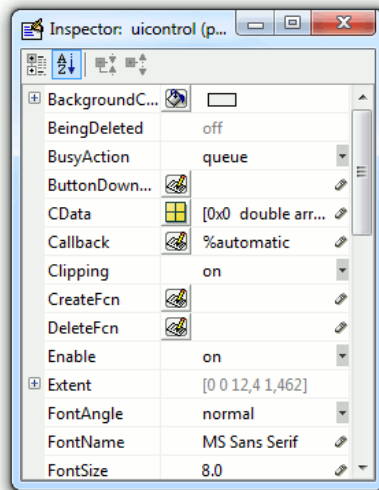


FIG. IV.3.2.b les propriétés des objets

Le GUIDE possède également des outils pour gérer l'alignement des objets et pour créer des barres d'outils ou des menus.

Une fois l'interface graphique terminée, son enregistrement donne deux fichiers portant le même nom mais dont les deux extensions sont .fig et .m.

Le fichier .fig contient la définition des objets graphiques (positions et propriétés). Ce fichier peut être ouvert ultérieurement avec le GUIDE pour modifier les objets graphiques.

Le fichier .m contient les lignes de code qui assurent le fonctionnement de l'interface graphique (actions des objets). Ce fichier peut être édité dans le MATLAB Editor pour y ajouter des actions à la main. C'est ce fichier qui doit être lancé pour utiliser l'interface graphique.

IV.3.2.1 Ajout de composants à la zone Mise en page:

Vous pouvez placer un composant dans la zone de mise en page dans une de ces façons:

- Faites glisser le composant de la palette de composants dans le domaine de la mise en page et laissez tomber.
- Sélectionnez le composant dans la palette de composants. Le curseur se transforme en une traverser.

- Placez le curseur dans la zone de mise en page où vous souhaitez placer l'angle supérieur gauche du composant à être et cliquez sur OK.
- Placez le curseur dans la zone de mise en page où vous souhaitez placer l'angle supérieur gauche de la composante à être, définissez la taille de la commande en cliquant sur et faisant glisser le curseur sur le coin inférieur gauche avant de relâcher la souris bouton.

IV.3.3.1 le programme de l'affichage :

Ce programme, en réalité, ne comprend pas que les instructions d'affichage, mais il contient aussi les instructions de la lecture.

La réception des données du port série peut être programmée par la fonction que nous trouver dans logicielle MATLAB est :

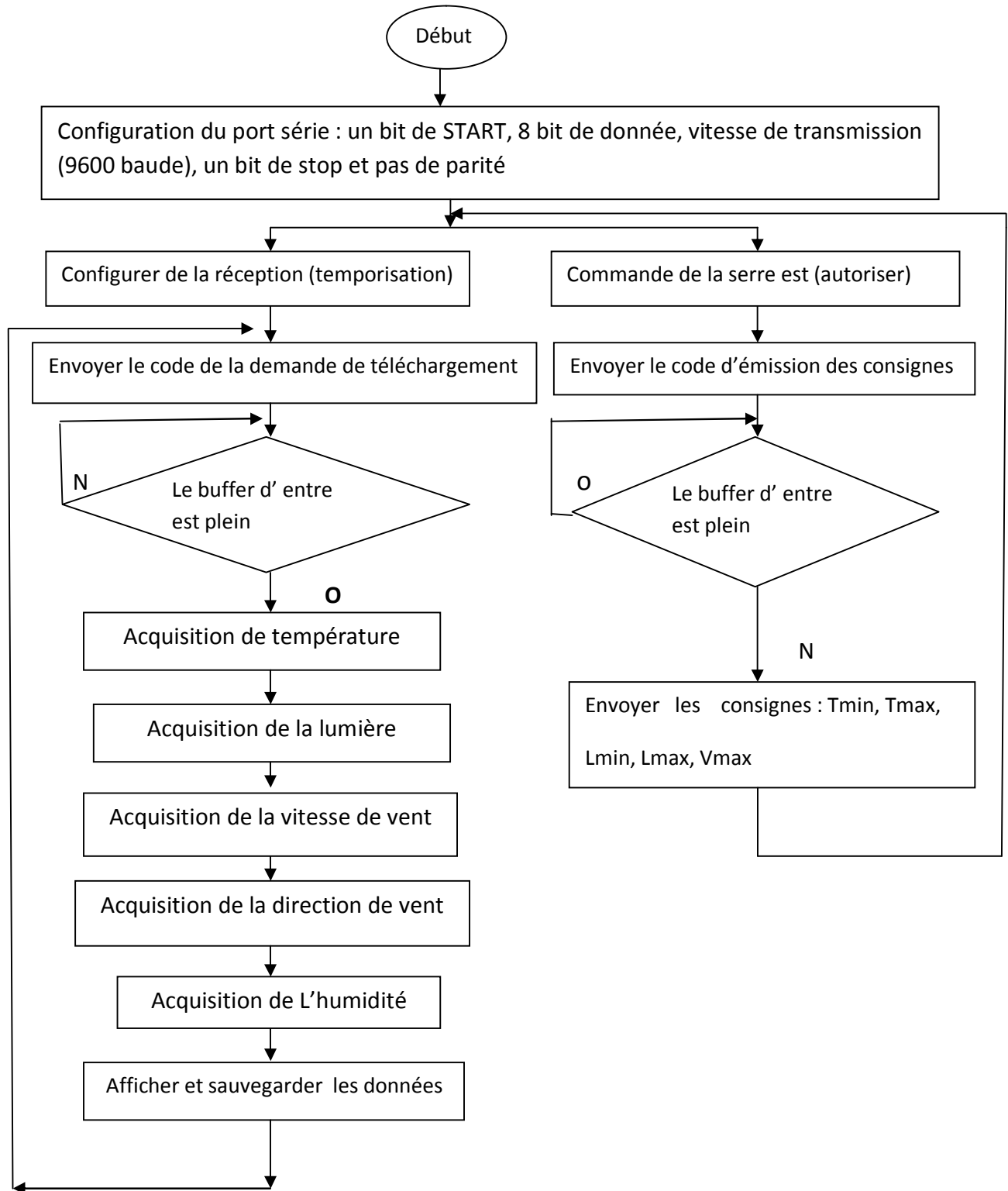
```
S = Serial ('COMX','BaudRate',9600);
```

Où **X** doit être remplacée par le numéro du port COM, cette fonction il est utilisé pour la communication série avec les appareils externes par RS232 ou USB. Cette fonction elle est plusieurs propriétés pour détailler les caractéristiques de la transmission et du port série .en plus il offre plusieurs possibilités pour lire et écrire dans les deux types de la transmission sérié (synchrone et Asynchrone).

Le port COM doit être ouvert par la fonction « **fopen(s)** » pour la lecture ou pour l'écriture, et a la fin doit fermer par la fonction « **fclose(s)** ».

Notres logiciel générale est surtout base sur les éléments que l'on vient de voir ci-dessus.

IV.3.3.2 organigramme général de gestion (logiciel) :



Org.IV.3.3.2 organigramme général de la gestion de logiciel

IV.3.3.3 Environnement de logiciel :

Notre logiciel, qui port le nom **METEOLOGICIEL**, a pour but de faciliter la tâche au technicien de gestion, par l'affichage des donnée météorologiques et de les sauvegarder dans une base ayant des capacités de sauvegarder très importante ainsi de contrôler la commande d'une serre agricole.

IV.3.3.4 Présentation du logiciel :

Nous allons voir maintenant les principaux éléments de notre interface logiciel.

A) Page d'accueil : c'est la page du logiciel, elle permet d'accéder à la configuration de la Réception de base de données ou la commande de la serre.

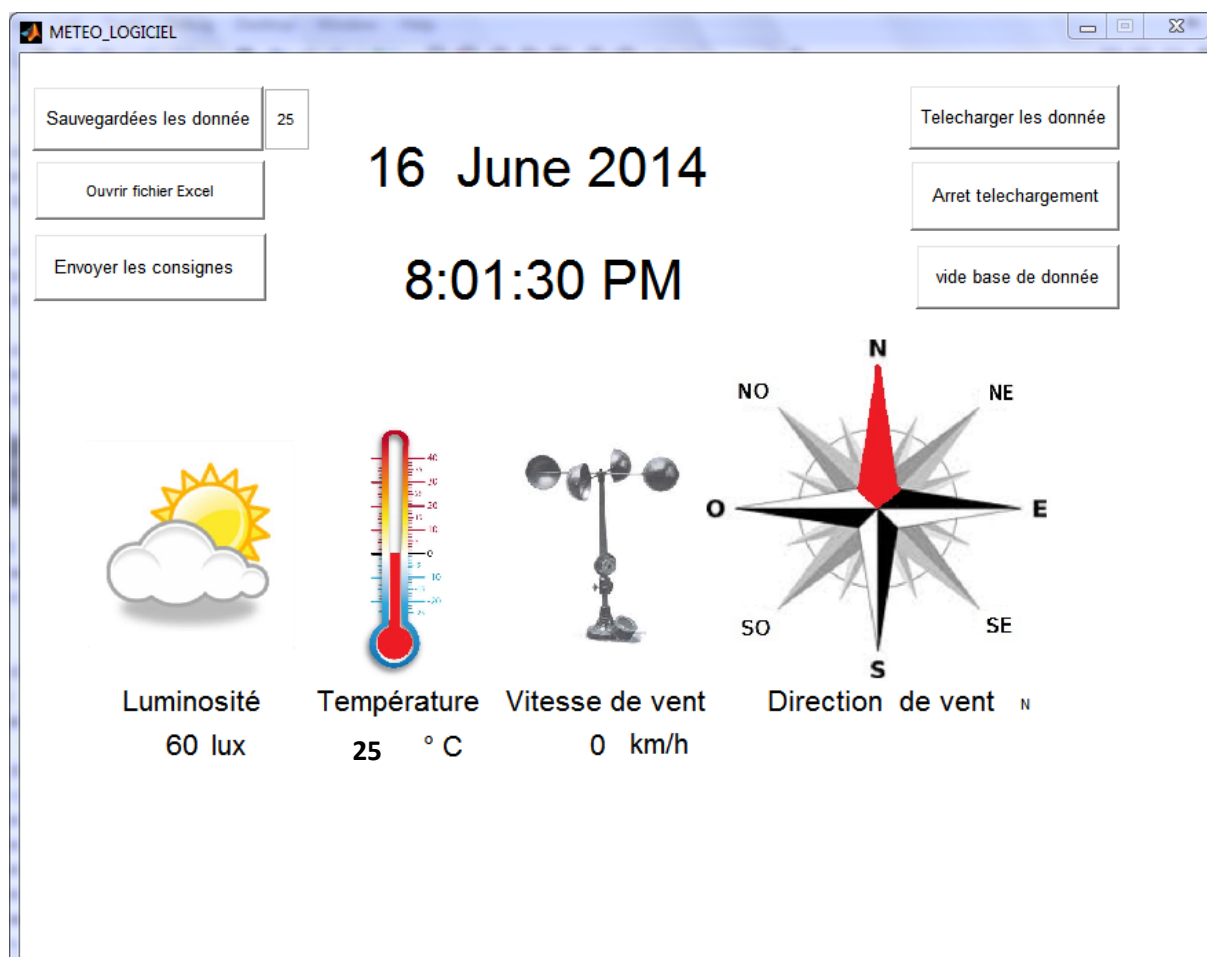


Fig. IV.3.3.4.A interface principale

B) le champ de travail et les commandes :

Dans le menu de l'interface principal, nous trouverons différentes commandes qui seront mis la disposition de l'utilisateur.

L'interface principal contient 5 **Bouton-poussoir** voir (fig. IV.3.3.4.B) en définir chaque Bouton :

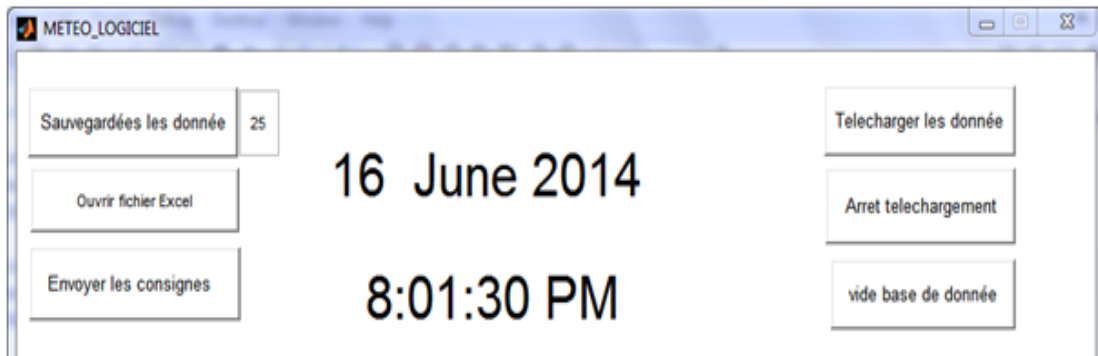
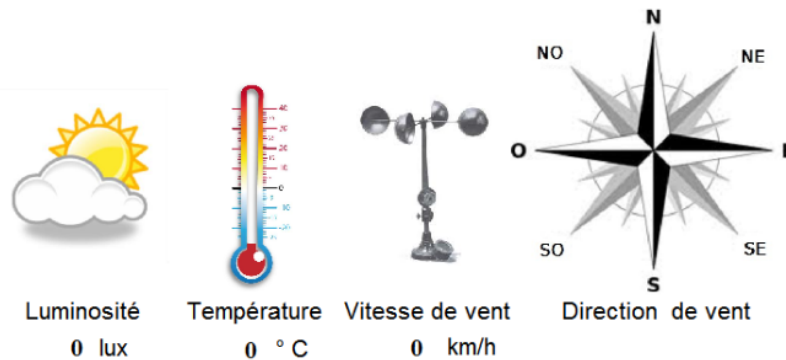


Fig. IV.3.3.4.B Les Boutons

- **Bouton télécharger les donnée** : lorsque on cliquez sur cet Bouton le programme de logiciel MATLAB ouvrir le port série USB et lire l'Acquisition de donnée (température, la lumière, la vitesse de vent, la direction de vent) et Afficher sur l'interface et sauvegarder les donnée dans un fichier EXCEL dans tableau avec date et heure comme la figure ci- dessous :

	A	B	C	D	E	F	G	H
1	date	heure	luminosité	temperatur	vitesse vent	direction de vent		
2	31 May 2014	4:43:50 PM	20	20	4	N		
3	31 May 2014	4:44:11 PM	80	19	8	N		
4	31 May 2014	4:44:21 PM	80	18	6	S / E		
5	31 May 2014	4:44:46 PM	40	26	4	E		
6	31 May 2014	4:45:07 PM	20	30	2	O		
7	31 May 2014	4:45:38 PM	60	22	6	N / O		
8	31 May 2014	4:46:31 PM	60	15	6	S / E		
9	31 May 2014	4:46:40 PM	80	17	8	S / E		
10	31 May 2014	4:48:14 PM	40	25	4	N / O		
11	31 May 2014	4:50:17 PM	40	24	4	S		
12	31 May 2014	4:50:29 PM	40	23	4	S		
13	31 May 2014	4:51:12 PM	20	26	2	O		
14	31 May 2014	4:51:18 PM	80	26	8	N / E		
15	31 May 2014	4:51:52 PM	80	26	8	E		
16	31 May 2014	4:51:57 PM	80	26	8	E		
17	31 May 2014	4:40:04 PM	20	26	2	N		
18								
19								
20								

- **Bouton Arrêt téléchargement** : lorsque on clique sur cet Bouton le programme de logiciel **MATLAB** fermer le port série USB et arrê l'Acquisition de donnée.
- **Bouton vider base de données** : lorsque on clique sur cet Bouton le programme de logiciel **MATLAB** il est effacer les donnée sur l'interface et mise à l'état initial comme la figure ci- dessous :



- **Bouton ouvrir fichier Excel** : lorsque on clique sur ce Bouton le programme de logiciel **MATLAB** ouvrir le fichier Excel **qui** sauvegarder les donnée.
- **Bouton envoyer les consignes** : lorsque on clique sur cet Bouton le programme de logiciel **MATLAB** envoyer les consignes (Tmin, Tmax, Lmin,Lmax,Vmax) vers la serre.

Conclusion :

Dans ce chapitre on vient de voir les principaux outils de programmation utilise pour notre application a savoir la programmation de logiciel **arduino** pour le chargement de la carte arduino d'acquisition et de commande ainsi que la programmation en langage évolue pour rendre l'utilisation de notre application très conviviale.

Bibliographie

- | Auteur | Titre |
|--|--|
| [1] Georges Asch et collorateurs
Edition DUNOD, PARIS 1999 | « les capteurs en instrumentations industrielles »
BIB. ELEC. Blida |
| [2] LURENT URBAN tom 1
Edition TEC.DOC, PARIS 1997 | « introduction a la production sous serre »
BIB.AGRO. Blida |
| [3] GUY ISABEL
Edition DUNOD, PARIS 1998 | « construire ses capteurs météo »
C.C.E Alger |
| [4] www.mon-club-elec.fr/pmwiki_reference_arduino/pmwiki.php?n=Main.Mate | |

Page last modified on May 11, 2012, at 09:37 PM

Enlighten theme originally by [styleshout](#) , adapted by [David Gilbert](#), powered by [PmWiki](#)

- [5] <http://f-leb.developpez.com>
- [6] <http://www.LDR-Datasheet.pdf>
- [7] www.siteduzero.com tres bon tutoriel d'arduino et TP.pdf.com
- [8] <http://arduino.cc/fr/Main/Serial>
- [9] [http://www.Les-afficheurs à cristaux liquides .htm](http://www.Les-afficheurs-a-cristaux-liquides.htm)
- [10] [http://www.AFFICHEURS A CRISTAUX LIQUIDES.htm](http://www.AFFICHEURS_A_CRISTAUX_LIQUIDES.htm)
- [12] [www.Arduino — Wikipédia.htm](http://www.Arduino-Wikipedia.htm)
- [13] [http:// :www.ancr.org/datasheet/opto/cny70.pdf](http://www.ancr.org/datasheet/opto/cny70.pdf)
- [14] [http://:www.frederic.ducroc.free.fr](http://www.frederic.ducroc.free.fr)
- [15] [http://:www.Météorologie — Wikipédia.com](http://www.Meteorologie-Wikipedia.com)

[http// :fr.wikipedia.org/wiki/Meteorologie](http://fr.wikipedia.org/wiki/Meteorologie)

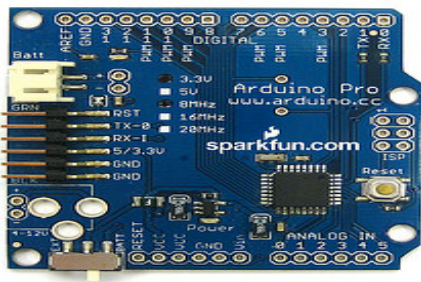
Les cartes Arduino



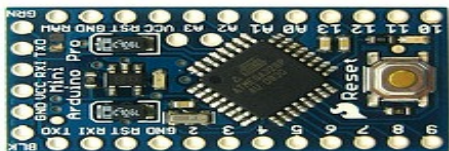
- La carte **Arduino Mega2560** : La version de la **Mega** sortie en même temps que la carte **UNO**. Cette version est basée sur l'**ATmega2560**, qui dispose de deux fois plus de mémoire, et utilise également l'**ATMega 8U2** pour la communication **USB-vers-série**.



- La carte **Arduino Fio**



- La carte **Arduino Pro**



- La carte **Arduino ProMini**



- La carte **Arduino Bluetooth**



- La carte **ArduinoMini**
- La **Mini USB Adapter**



- La carte **Arduino Nano**

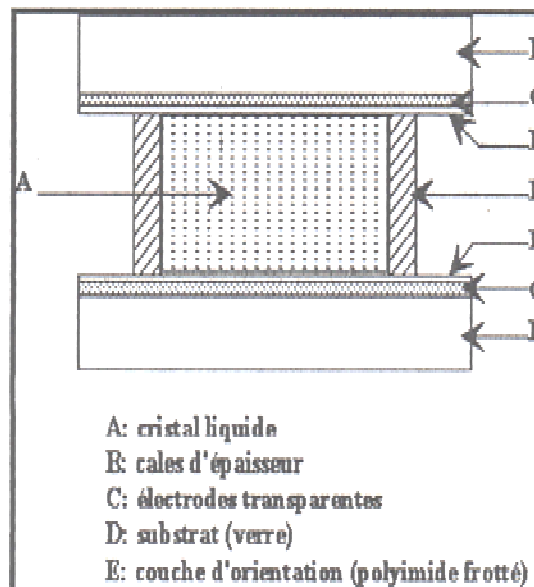
Afficheur LCD :

Principe des cristaux liquides

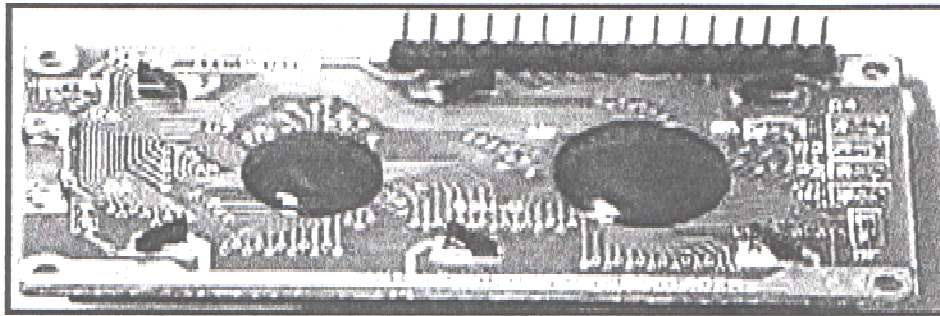
L'afficheur est constitué de deux lames de verre, distantes de 20 μm environ, sur lesquelles sont dessinées les mantisses formant les caractères. L'espace entre elles est rempli de cristal liquide normalement réfléchissant (pour les modèles réflexifs). L'application entre les deux faces d'une tension alternative basse fréquence de quelques volts (3 à 5 V) le rend absorbant. Les caractères apparaissent sombres sur fond clair. Les modèles transmissifs fonctionnent différemment: normalement opaque au repos, le cristal liquide devient transparent lorsqu'il est excité; pour rendre un tel afficheur lisible, il est nécessaire de l'éclairer par l'arrière.

a. Constitution d'une cellule à cristal liquide :

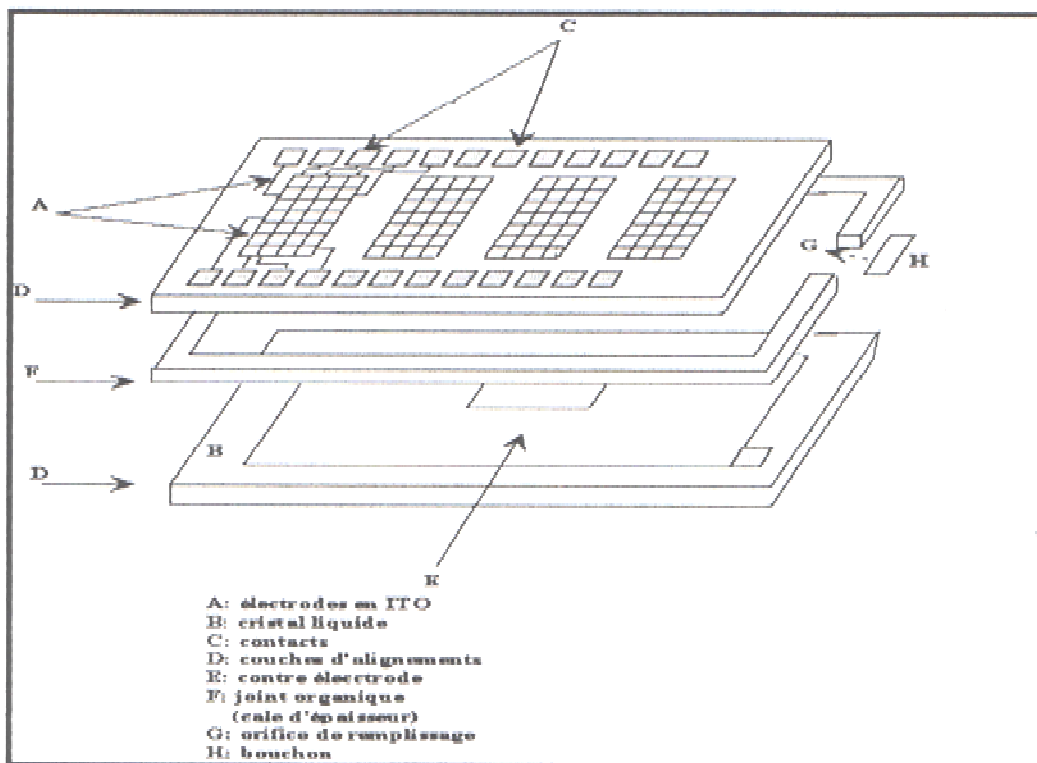
La figure suivante, donne les différentes couches d'une cellule cristal liquide.



c. l'afficheur LCD HYPER 1602B :



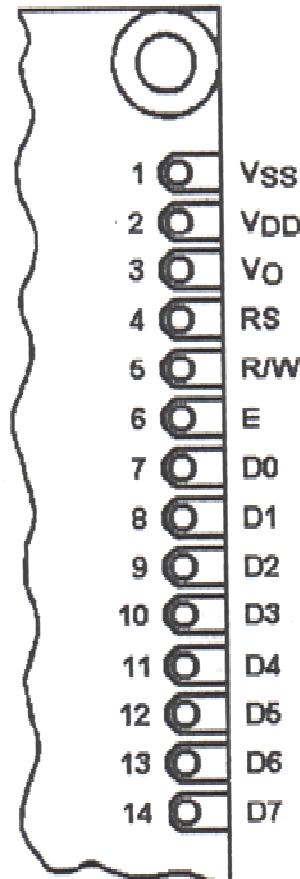
b. Constitution d'un afficheur à cristaux liquides



Brochage des afficheurs LCD :

Plusieurs modèles d'afficheurs de différentes marques sont compatibles avec cette interface. En effet, les afficheurs LCD disposent d'une interface parallèle normalisée répondant au tableau suivant :

	Nom	Niveau	Fonction
1	V _{ss}	-	Masses
2	V _{dd}	-	Alimentation positive +5V
3	V _o	0-5V	Cette tension permet, en la faisant varier entre 0 et +5V, le réglage du contraste de l'afficheur.
4	RS	TTL	Selection du registre (Register Select) Grâce à cette broche, l'afficheur est capable de faire la différence entre une commande et une donnée. Un niveau bas indique une commande et un niveau haut indique une donnée.
5	R/W	TTL	Lecture ou écriture (Read/Write) L : Écriture H : Lecture
6	E	TTL	Entrée de validation (Enable) active sur front descendant. Le niveau haut doit être maintenue pendant au moins 450 ns à l'état haut
7	D0	TTL	Bus de données bidirectionnel 3 états (haute impédance lorsque E=0)
8	D1	TTL	
9	D2	TTL	
10	D3	TTL	
11	D4	TTL	
12	D5	TTL	
13	D6	TTL	
14	D7	TTL	
15	A	-	Anode rétroéclairage (+5V)
16	K	-	Cathode rétroéclairage (masse)



Les broches 15 et 16 ne sont présentes que sur les afficheurs LCD avec rétroéclairage.

Les connexions à réaliser sont simples puisque l'afficheur LCD dispose de peu de broches. Il faut évidemment, l'alimenter, le connecter à un bus de donnée (4 ou 8 bits), et connecter les broches E, R/W et RS.

a. Rôle des principales broches des afficheurs LCD :

- V_O : potentiel continue permettant d'ajuster le contraste de l'affichage.
 - RS : l'état logique de cette ligne définit la nature des informations envoyées à l'afficheur. Si RS=1 on envoie une donnée. Si RS=0 on envoie une commande.
 - R/W : permet de lire ou écrire au sein de la RAM de l'afficheur.
 - E : validation des informations transmises à l'afficheur.
 - D0 à D7 : 8 lignes de données bidirectionnelles (selon la position de R/W).
 - A et C : Ces deux broches supplémentaires sont disponibles uniquement sur les afficheurs rétro éclairés.
- Pour de plus amples explications, voir un plus bas dans cette page.

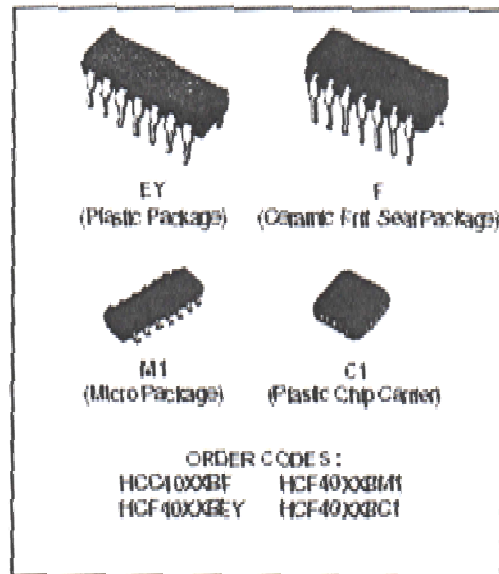


HCC4011B/12B/23B
HCF4011B/12B/23B

NAND GATES

QUAD 2 INPUT HCC/HCF 4011B
DUAL 4 INPUT HCC/HCF 4012B
TRIPLE 3 INPUT HCC/HCF 4023B

- PROPAGATION DELAY TIME = 60ns (typ.) AT $C_L = 50\text{pF}$, $V_{DD} = 10\text{V}$
- BUFFERED INPUTS AND OUTPUTS
- QUIESCENT CURRENT SPECIFIED TO 20V FOR HCC DEVICE
- INPUT CURRENT OF 100nA AT 18V AND 25°C FOR HCC DEVICE
- 100% TESTED FOR QUIESCENT CURRENT
- 5V, 10V AND 15V PARAMETRIC RATINGS
- MEETS ALL REQUIREMENTS OF JEDEC TENTATIVE STANDARD N° 13A, "STANDARD SPECIFICATIONS FOR DESCRIPTION OF 'B' SERIES CMOS DEVICES"

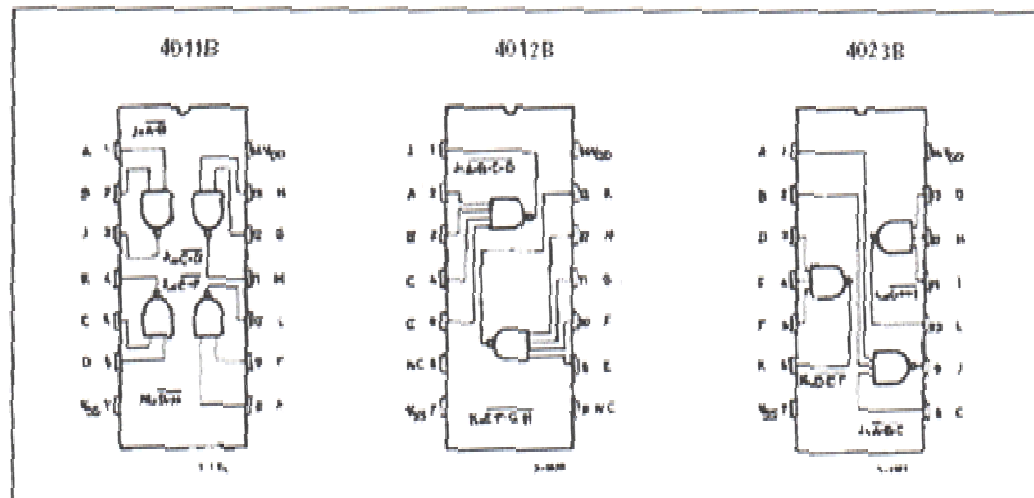


DESCRIPTION

The HCC4011B, HCC4012B and HCC4023B (extended temperature range) and HCF4011B, HCF4012B and HCF4023B (intermediate temperature range) are monolithic, integrated circuit, available in 14-lead dual in-line plastic or ceramic package and plastic micropackage.

The HCC/HCF4011B, HCC/HCF4012B and HCC/HCF4023B NAND gates provide the system designer with direct implementation of the NAND function and supplement the existing family of COS/MOS gates. All inputs and outputs are buffered.

PIN CONNECTIONS



HCC/HFC4011B/12B/23B

ABSOLUTE MAXIMUM RATINGS

Symbol	Parameter	Value	Unit
V_{DD}^*	Supply Voltage : HCC Types HCF Types	- 0.5 to + 20 - 0.5 to + 18	V V
V_i	Input Voltage	- 0.5 to $V_{DD} + 0.5$	V
I_i	DC Input Current (any one input)	± 10	mA
P_{tot}	Total Power Dissipation (per package) Dissipation per Output Transistor for T_{op} = Full Package-temperature Range	200 100	mW mW
T_{op}	Operating Temperature : HCC Types HCF Types	- 65 to + 125 - 40 to + 85	°C °C
T_{stg}	Storage Temperature	- 65 to + 150	°C

Stresses above those listed under "Absolute Maximum Ratings" may cause permanent damage to the device. This is a stress rating only and functional operation of the device at these or any other conditions above those indicated in the operational sections of this specification is not implied. Exposure to absolute maximum rating conditions for extended periods may affect device reliability.

* All voltage values are referred to V_{SS} pin voltage.

RECOMMENDED OPERATING CONDITIONS

Symbol	Parameter	Value	Unit
V_{DD}	Supply Voltage : HCC Types HCF Types	3 to 18 3 to 15	V V
V_i	Input Voltage	0 to V_{DD}	V
T_{op}	Operating Temperature : HCC Types HCF Types	- 65 to + 125 - 40 to + 85	°C °C

Features

- High Performance, Low Power AVR[®] 8-Bit Microcontroller
- Advanced RISC Architecture
 - 131 Powerful Instructions – Most Single Clock Cycle Execution
 - 32 x 8 General Purpose Working Registers
 - Fully Static Operation
 - Up to 20 MIPS Throughput at 20 MHz
 - On-chip 2-cycle Multiplier
- High Endurance Non-volatile Memory Segments
 - 4/8/16/32K Bytes of In-System Self-Programmable Flash program memory (ATmega48PA/88PA/168PA/328P)
 - 256/512/512/1K Bytes EEPROM (ATmega48PA/88PA/168PA/328P)
 - 512/1K/1K/2K Bytes Internal SRAM (ATmega48PA/88PA/168PA/328P)
 - Write/Erase Cycles: 10,000 Flash/100,000 EEPROM
 - Data retention: 20 years at 85°C/100 years at 25°C⁽¹⁾
 - Optional Boot Code Section with Independent Lock Bits
 - In-System Programming by On-chip Boot Program
 - True Read-While-Write Operation
 - Programming Lock for Software Security
- Peripheral Features
 - Two 8-bit Timer/Counters with Separate Prescaler and Compare Mode
 - One 16-bit Timer/Counter with Separate Prescaler, Compare Mode, and Capture Mode
 - Real Time Counter with Separate Oscillator
 - Six PWM Channels
 - 8-channel 10-bit ADC in TQFP and QFN/MLF package
 - Temperature Measurement
 - 6-channel 10-bit ADC in PDIP Package
 - Temperature Measurement
 - Programmable Serial USART
 - Master/Slave SPI Serial Interface
 - Byte-oriented 2-wire Serial Interface (Philips I²C compatible)
 - Programmable Watchdog Timer with Separate On-chip Oscillator
 - On-chip Analog Comparator
 - Interrupt and Wake-up on Pin Change
- Special Microcontroller Features
 - Power-on Reset and Programmable Brown-out Detection
 - Internal Calibrated Oscillator
 - External and Internal Interrupt Sources
 - Six Sleep Modes: Idle, ADC Noise Reduction, Power-save, Power-down, Standby, and Extended Standby
- I/O and Packages
 - 23 Programmable I/O Lines
 - 28-pin PDIP, 32-lead TQFP, 28-pad QFN/MLF and 32-pad QFN/MLF
- Operating Voltage:
 - 1.8 - 5.5V for ATmega48PA/88PA/168PA/328P
- Temperature Range:
 - -40°C to 85°C
- Speed Grade:
 - 0 - 20 MHz @ 1.8 - 5.5V
- Low Power Consumption at 1 MHz, 1.8V, 25°C for ATmega48PA/88PA/168PA/328P:
 - Active Mode: 0.2 mA
 - Power-down Mode: 0.1 µA
 - Power-save Mode: 0.75 µA (Including 32 kHz RTC)



8-bit **AVR**[®]
Microcontroller
with 4/8/16/32K
Bytes In-System
Programmable
Flash

ATmega48PA
ATmega88PA
ATmega168PA
ATmega328P

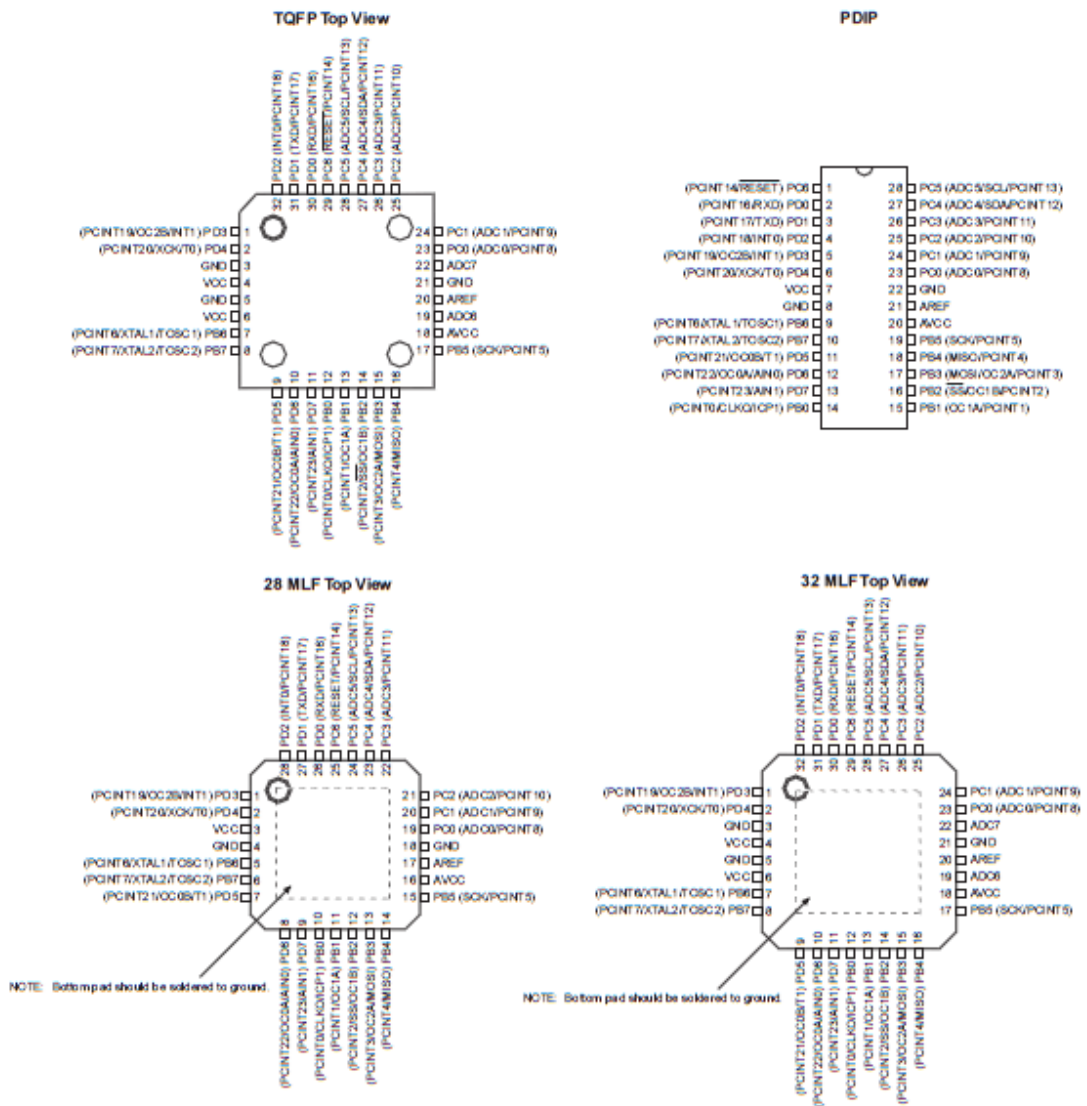
Rev. 8161D-AVR-1009



ATmega48PA/88PA/168PA/328P

1. Pin Configurations

Figure 1-1. Pinout ATmega48PA/88PA/168PA/328P



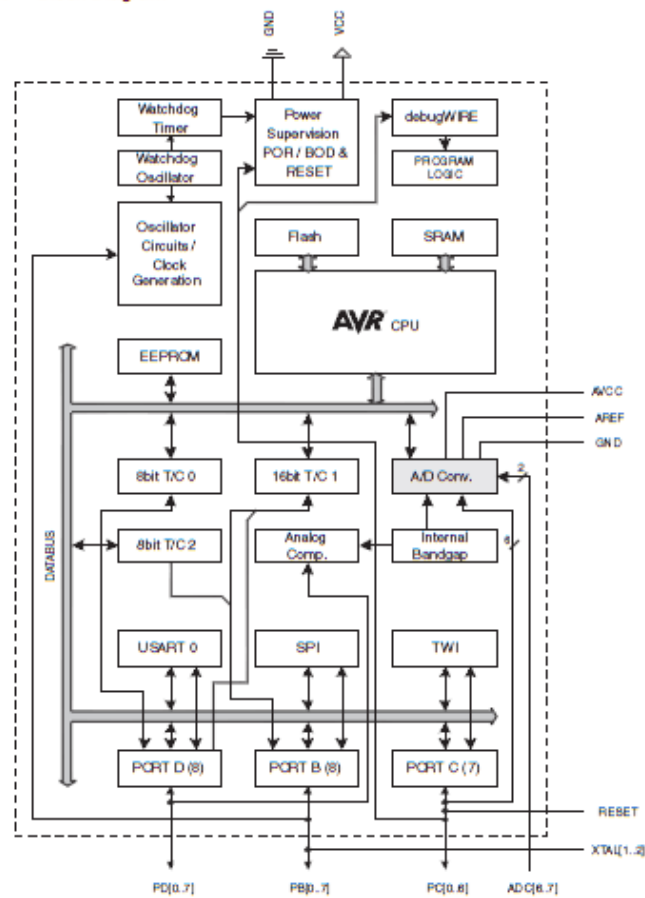
ATmega48PA/88PA/168PA/328P

2. Overview

The ATmega48PA/88PA/168PA/328P is a low-power CMOS 8-bit microcontroller based on the AVR enhanced RISC architecture. By executing powerful instructions in a single clock cycle, the ATmega48PA/88PA/168PA/328P achieves throughputs approaching 1 MIPS per MHz allowing the system designer to optimize power consumption versus processing speed.

2.1 Block Diagram

Figure 2-1. Block Diagram



The AVR core combines a rich instruction set with 32 general purpose working registers. All the 32 registers are directly connected to the Arithmetic Logic Unit (ALU), allowing two independent registers to be accessed in one single instruction executed in one clock cycle. The resulting

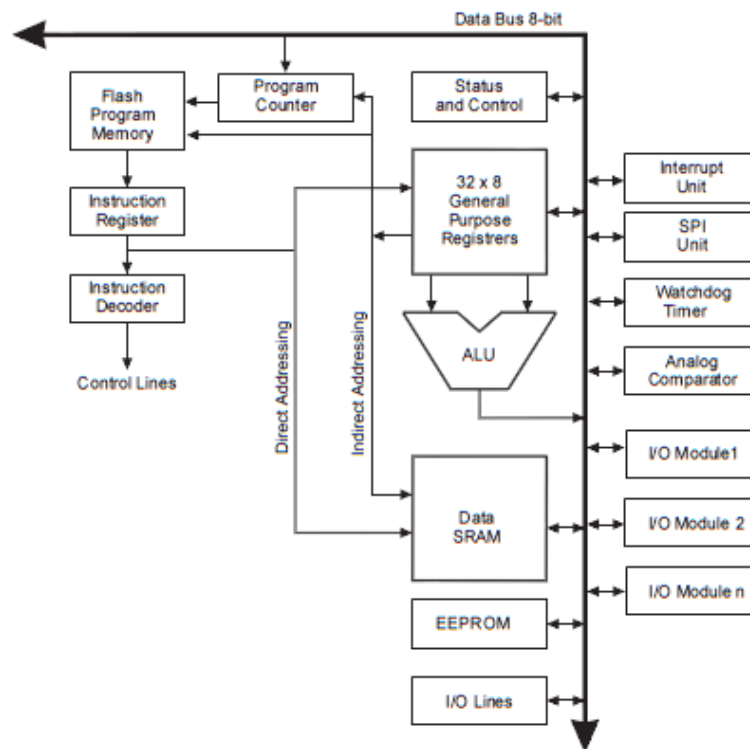
A Tmega48PA/88PA/168PA/328P

6. AVR CPU Core

6.1 Overview

This section discusses the AVR core architecture in general. The main function of the CPU core is to ensure correct program execution. The CPU must therefore be able to access memories, perform calculations, control peripherals, and handle interrupts.

Figure 6-1. Block Diagram of the AVR Architecture

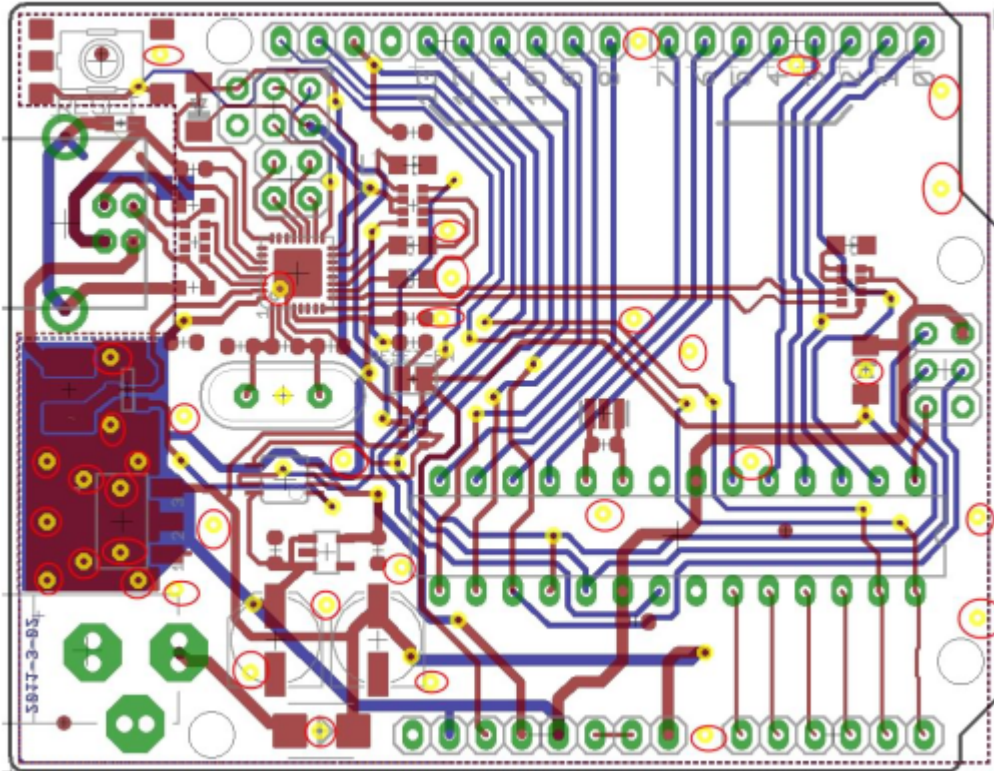


In order to maximize performance and parallelism, the AVR uses a Harvard architecture – with separate memories and buses for program and data. Instructions in the program memory are executed with a single level pipelining. While one instruction is being executed, the next instruction is pre-fetched from the program memory. This concept enables instructions to be executed in every clock cycle. The program memory is In-System Reprogrammable Flash memory.

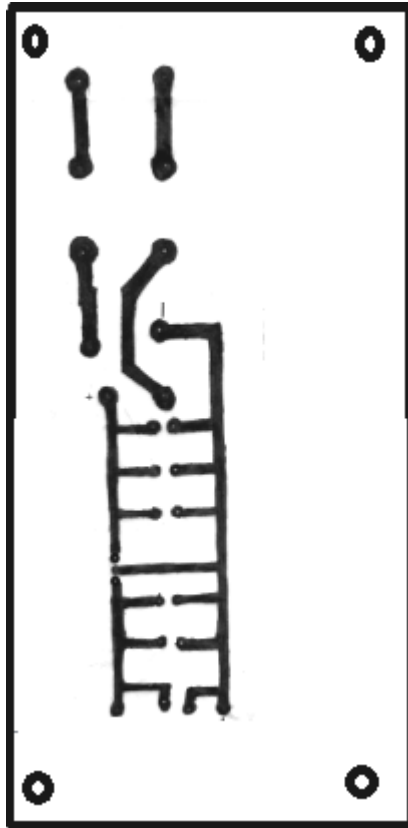
The fast-access Register File contains 32 x 8-bit general purpose working registers with a single clock cycle access time. This allows single-cycle Arithmetic Logic Unit (ALU) operation. In a typ-

Annexes

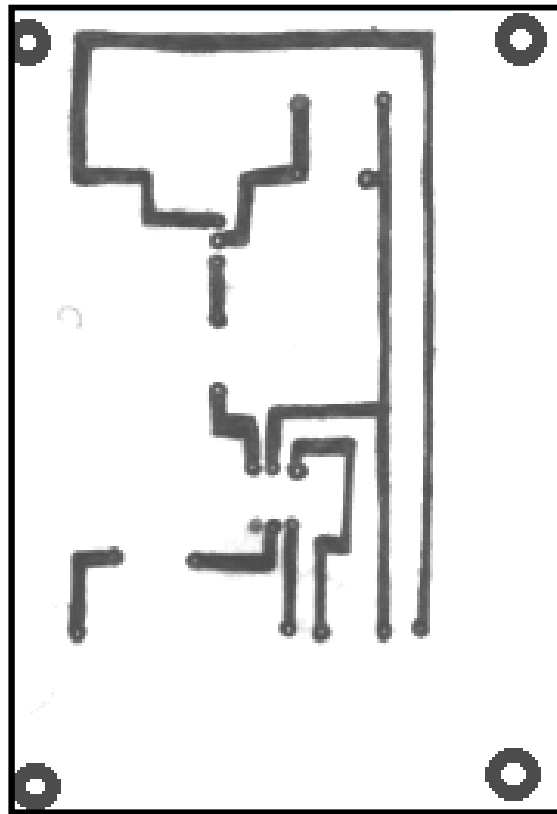
Circuit imprimé de la carte arduino uno



Annexes



Carte d'alimentation +12 v



cartes de puissance