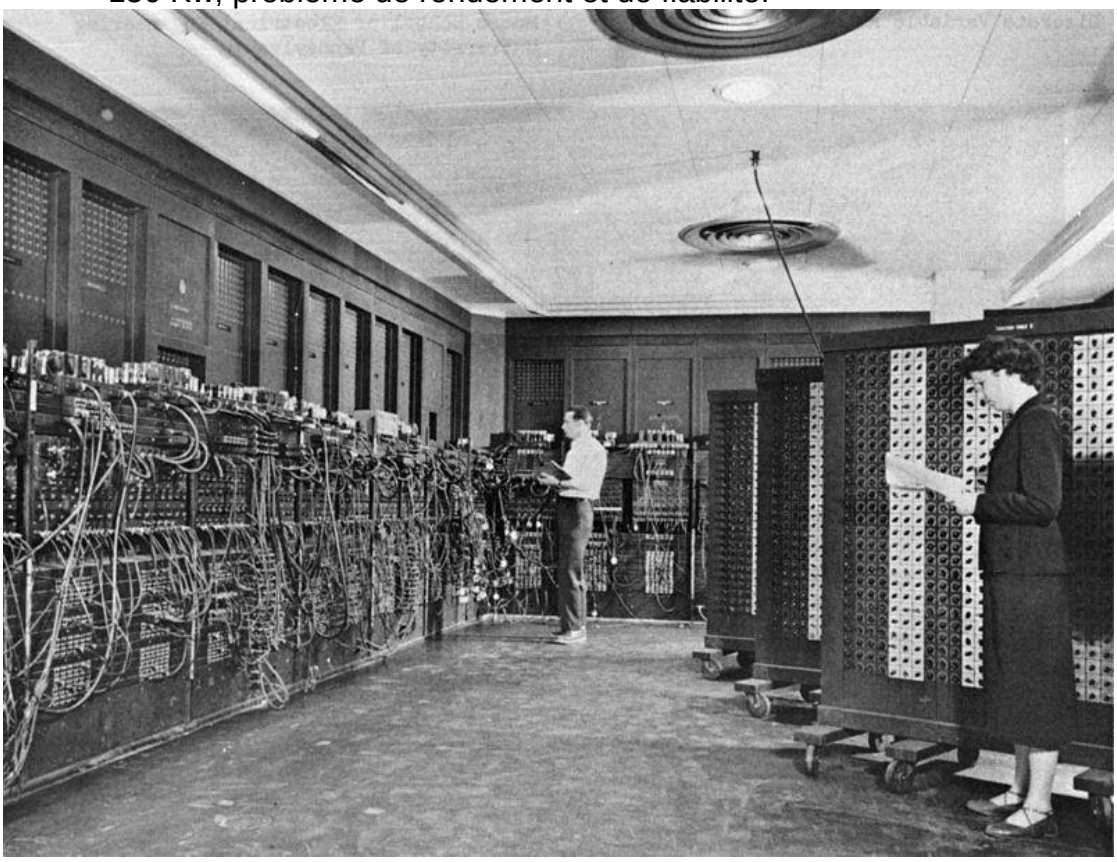
Présentation du matériel Arduino

1. Historique
2. Rappel sur le microprocesseur
3. Rappel sur le microcontrôleur
4. La carte Arduino Uno, la carte phare de Arduino
5. Les shields Arduino (carte d'extension)
6. Les autres cartes microcontrôleurs produites par Arduino
7. L'environnement de développement
8. Survol de quelques développements
9. Une réalisation effectuée par un non informaticien
10. Étude de cas, la BatteryBox
11. Où acheter le matériel ?
12. Arduino en résumé

1. Historique

- 1940 Premier ordinateur à relais mécaniques (Navy)

- 1946 Premier ordinateur à tubes à vide (1800, grande dissipation): 150 Kw, problème de rendement et de fiabilité.



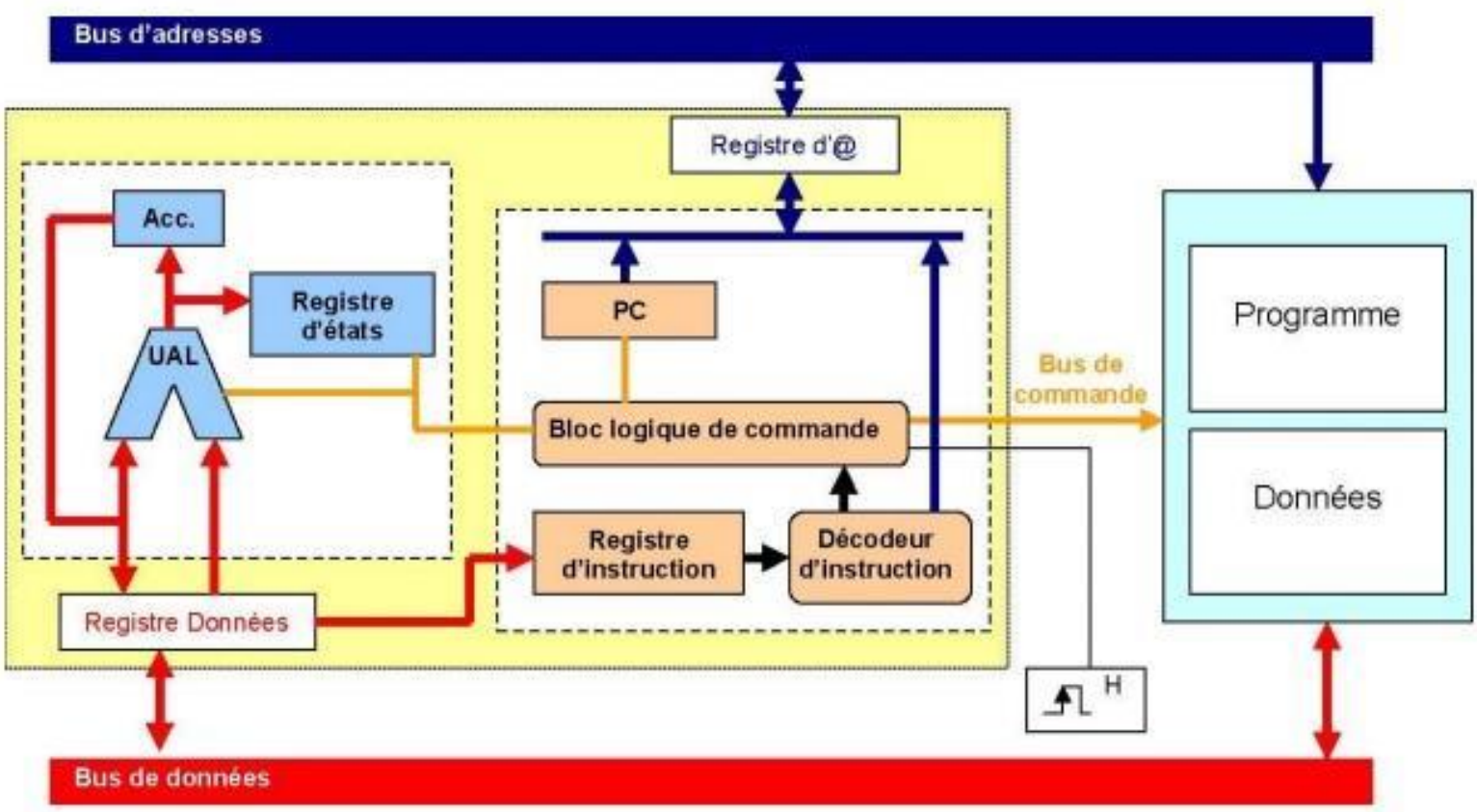
Une cause fréquente de panne était la combustion d'un insecte sur un tube chaud, provoquant un stress thermique local et la rupture de l'ampoule de verre. Le terme anglais désignant un insecte est *bug*. Ce terme, par extension, serait devenu synonyme de dysfonctionnement informatique

- 1948**:** découverte de l'effet transistor.
- 1958: 1er circuit intégré (5 transistors)
- 1975: microprocesseur 6800 Motorola

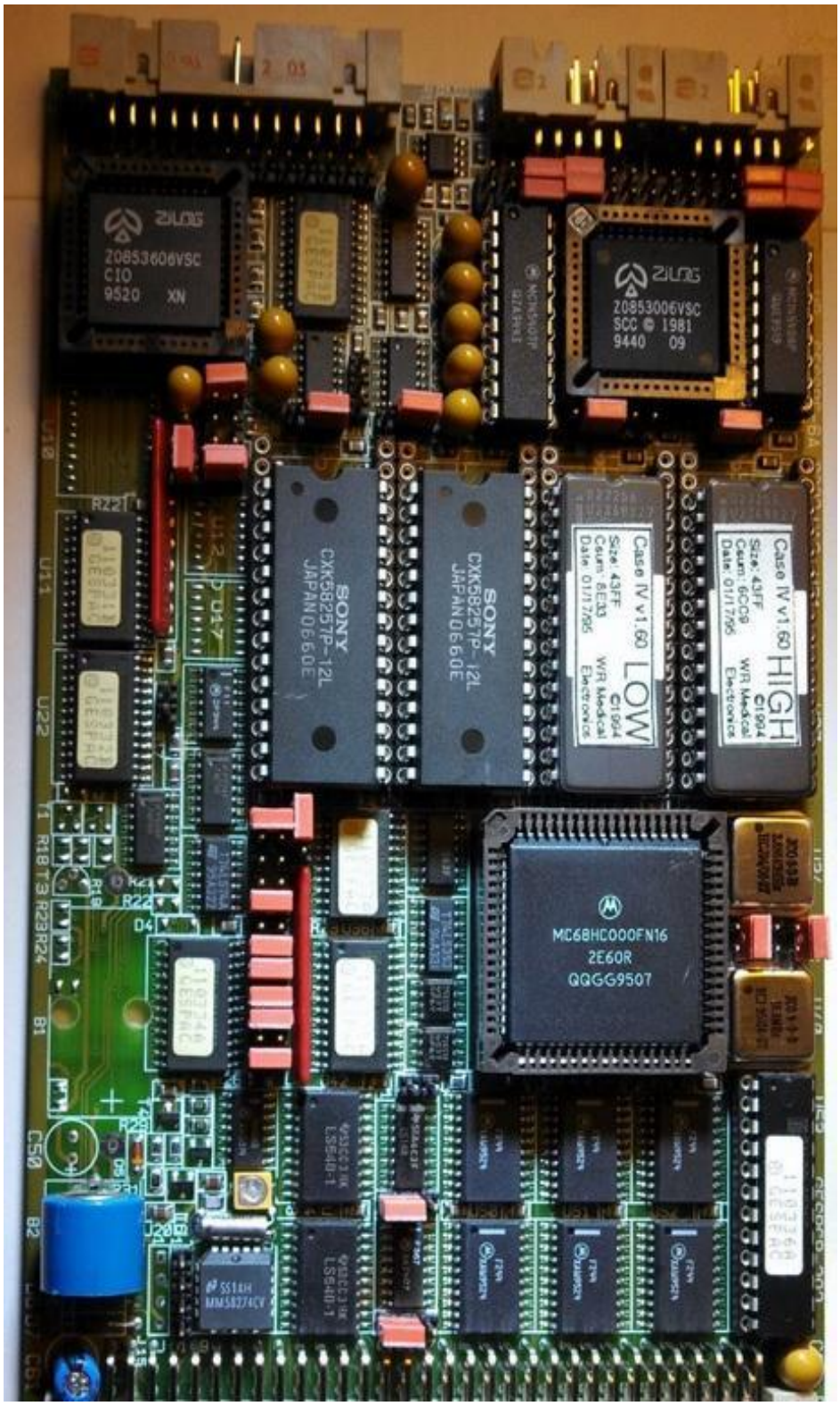
2. Rappel sur le microprocesseur

- Bus de d'adresse
- Bus de données
- Unité arithmétique
- Registre d'états
- Mémoire de programme
- Mémoire de données

Le logiciel est stocké en dehors du microprocesseur.
Il est réalisé par une chaine de développement : compilateur, éditeur de liens conversion de format et programmateur d'EPROM par exemple.



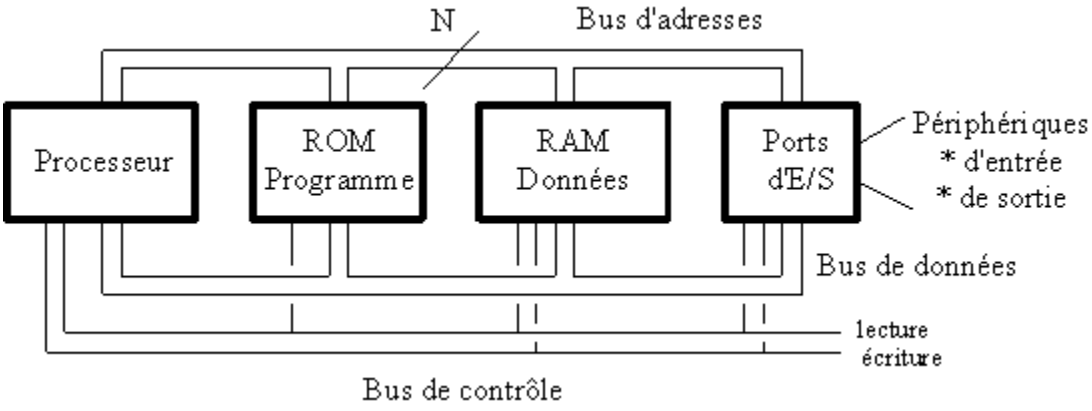
Carte Gespac 68000



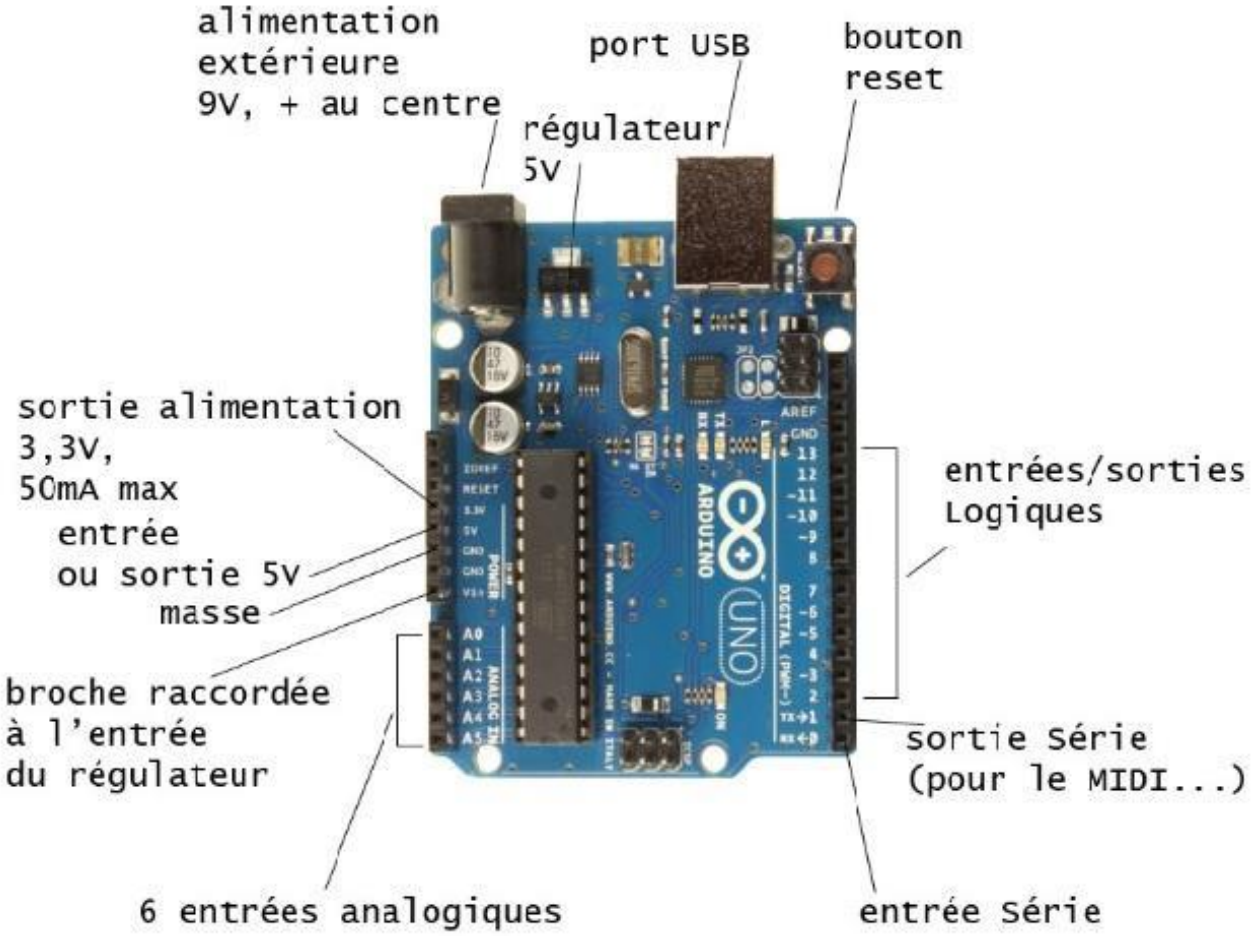
3. Rappel sur le microcontrôleur

- RAM / EEPROM:; intégré
- UART: intégré
- Entrées / Sorties logiques
- Boot loader permet de recevoir le programme depuis l'extérieur

Les microcontrôleurs d'aujourd'hui ont beaucoup de périphériques intégrés.
Ils en ont bien plus que le nombre de pins disponible sur le circuit.
Le logiciel permettra de choisir la configuration souhaitée.
Une partie des choix est fait dans le boot loader donc hors de notre portée !



4. La carte Arduino UNO, la carte phare de Arduino

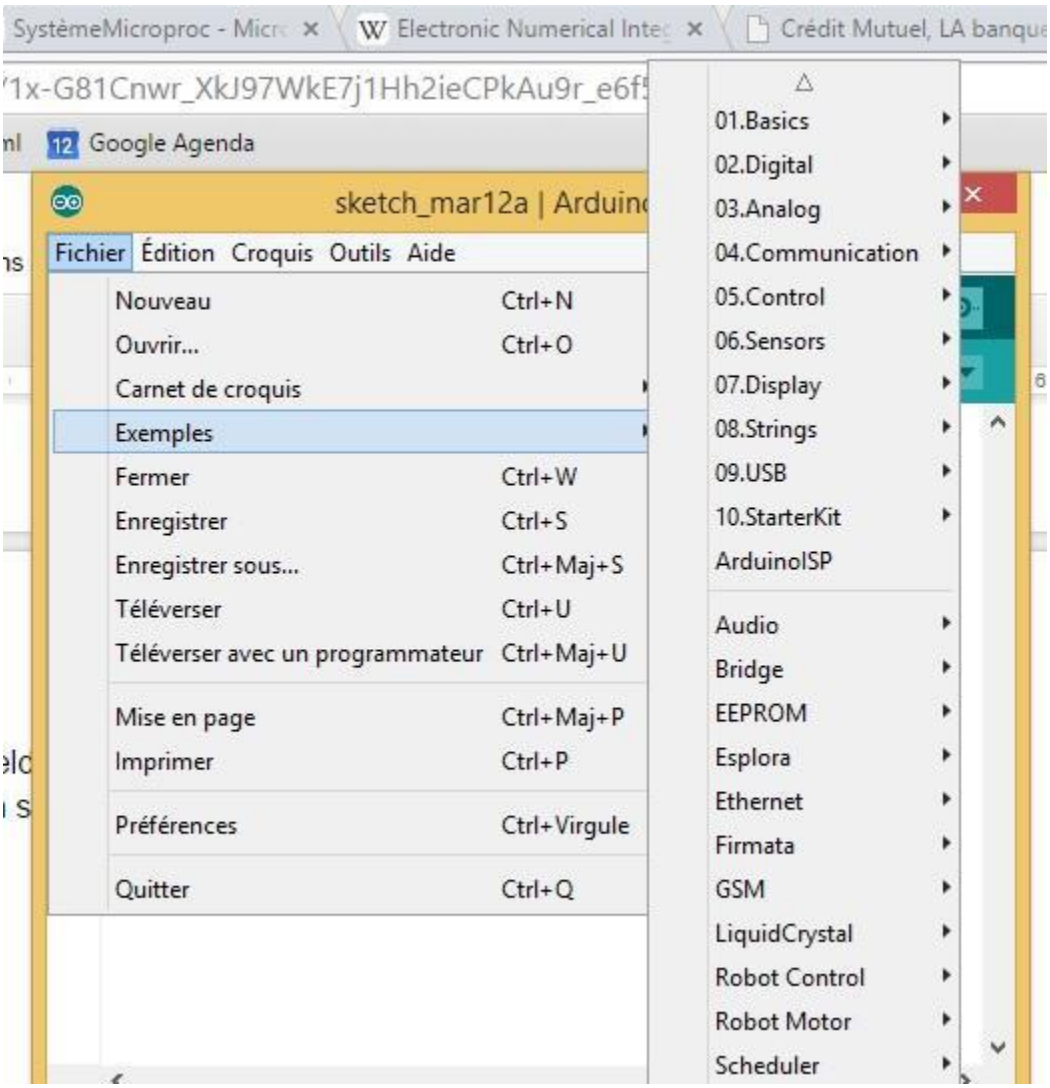


Tension d'entrée	6-20V
Entrées sorties digitale	14 (of which 6 provide PWM output)
Entrée analogique	6
Flash Memory Boot loader	32 KB (ATmega328)
RAM	2 KB (ATmega328)
EEPROM	1 KB (ATmega328)
Horloge	16 MHz
Longueur	68.6 mm
Largeur	53.4 mm

5. Les shields Arduino (carte d'extension)

Avec un shield, on reçoit:

- le hardware
- le schéma
- une librairie
- des exemples de programmes
- souvent, accès au forum du constructeur



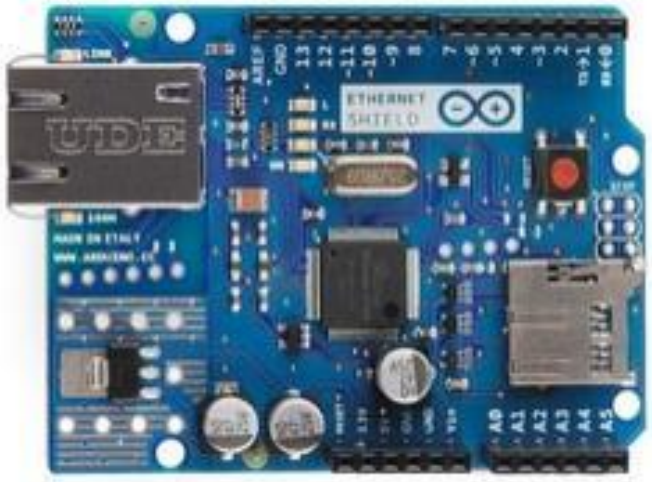
Différents shields (cartes d'extension enfichables sur un Arduino Uno)

- Carte à 4 relais

- Afficheur LCD, afficheur I2C avec boutons


```
class Adafruit_RGBLCDShield : public Print {
public:
  Adafruit_RGBLCDShield();

  void init(uint8_t fourbitmode, uint8_t rs, uint8_t rw, uint8_t enable,
          uint8_t d0, uint8_t d1, uint8_t d2, uint8_t d3,
          uint8_t d4, uint8_t d5, uint8_t d6, uint8_t d7);

  void begin(uint8_t cols, uint8_t rows, uint8_t charsize = LCD_5x8DOTS);

  void clear();
  void home();

  void noDisplay();
  void display();
  void noBlink();
  void blink();
  void noCursor();
  void cursor();
  void scrollDisplayLeft();
  void scrollDisplayRight();
  void leftToRight();
  void rightToLeft();
  void autoscroll();
  void noAutoscroll();

  void setBacklight(uint8_t status);

  void createChar(uint8_t, uint8_t[]);
  void setCursor(uint8_t, uint8_t);
#if ARDUINO >= 100
  virtual size_t write(uint8_t);
#else
  virtual void write(uint8_t);
#endif
  void command(uint8_t);
  uint8_t readButtons();
```

- Ethernet (avec carte SD)

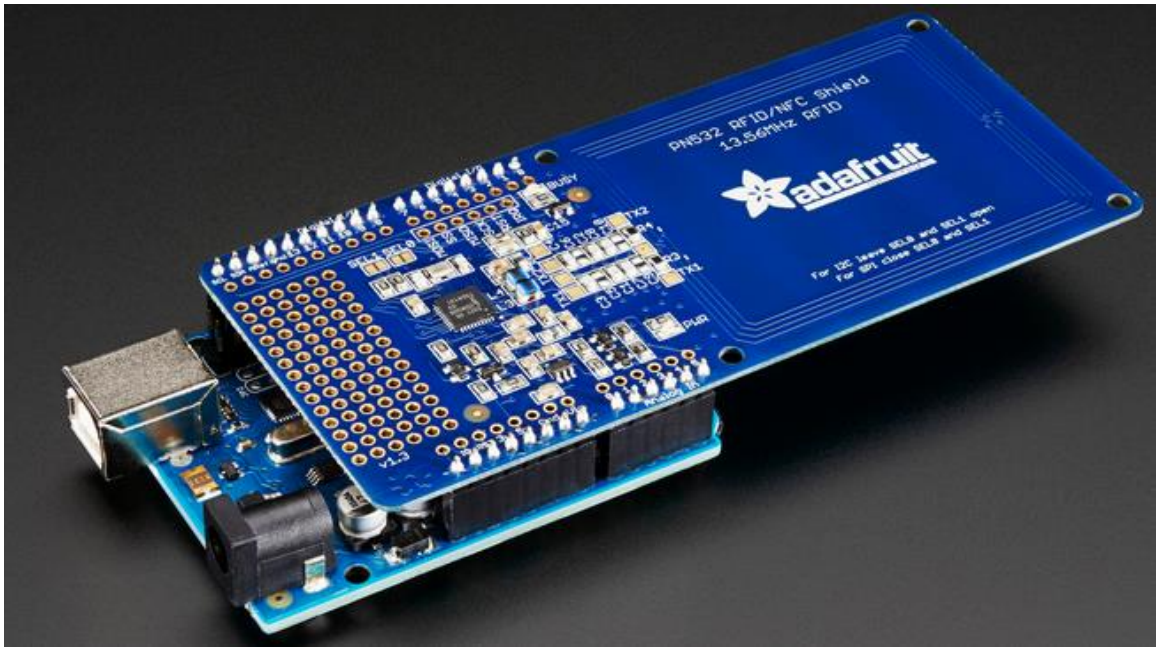


- GSM

- GPS

- Bluetooth
- Carte SD
- Moteurs, courant continu et pas à pas

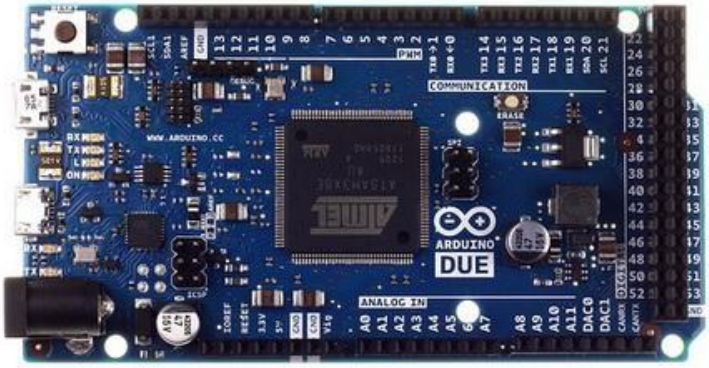
NFC (badge que l'on passe vers le lecteur, carte de crédit…)



6. Les autres cartes microcontrôleurs produites par Arduino

- La carte Arduino micro (même microcontrôleur)

- la carte Due (microcontrôleur beaucoup plus puissant)



Il existe une multitude d'autres carte utilisable avec l'environnement de développement Arduino

7. L'environnement de développement

- IDE Arduino

Le langage C

Pas tout à fait standard…..

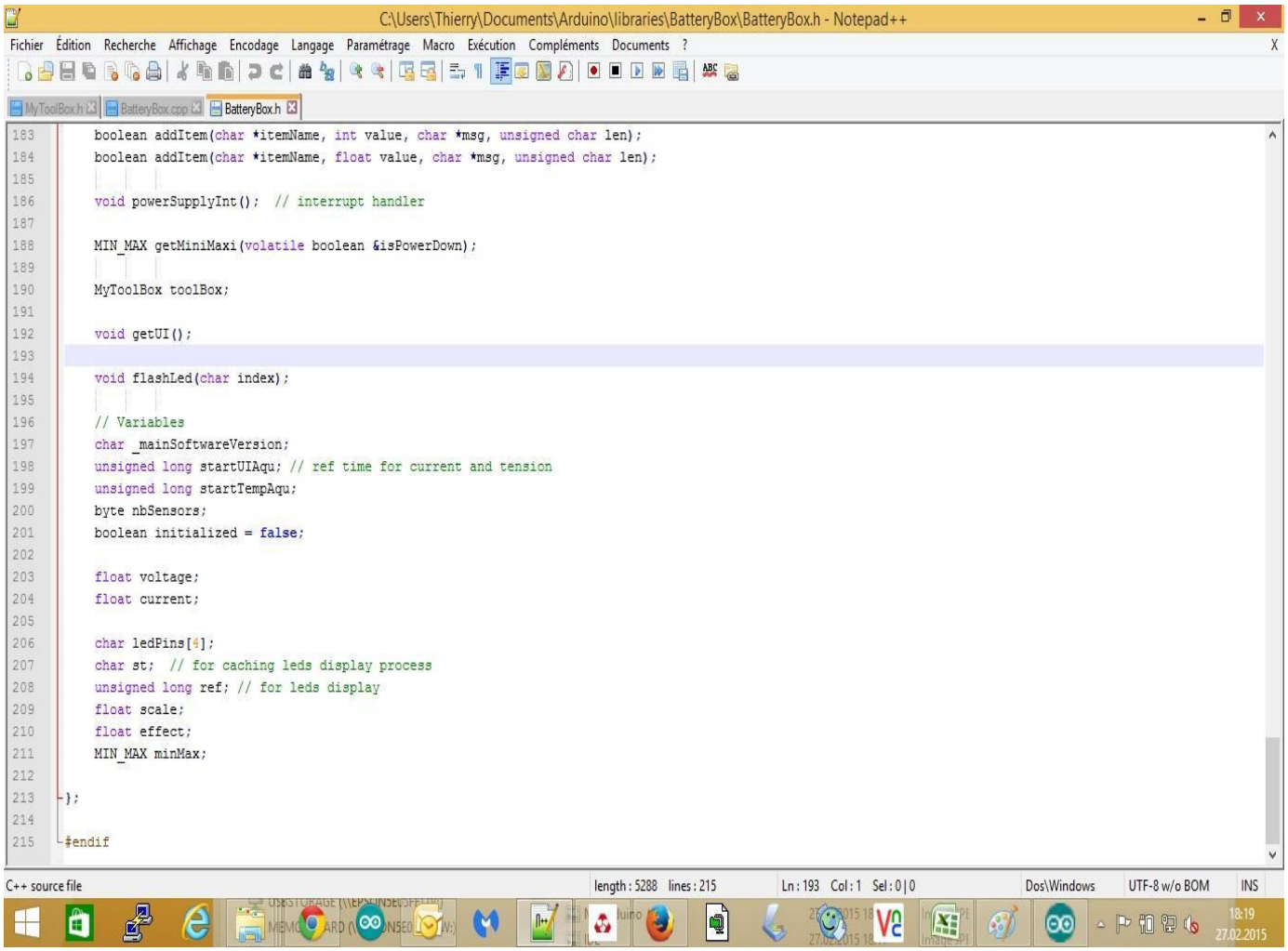
A noter qu'il existe d'autres environnements de développement:

Codeblocks: http://www.arduinodev.com/codeblocks/

ArduBloc: http://www.semageek.com/arduino-presentation-et-traduction-en-francais-de-ardublock/

- Les librairies:

On utilise un éditeur du genre notepad++
On les écrit en C++



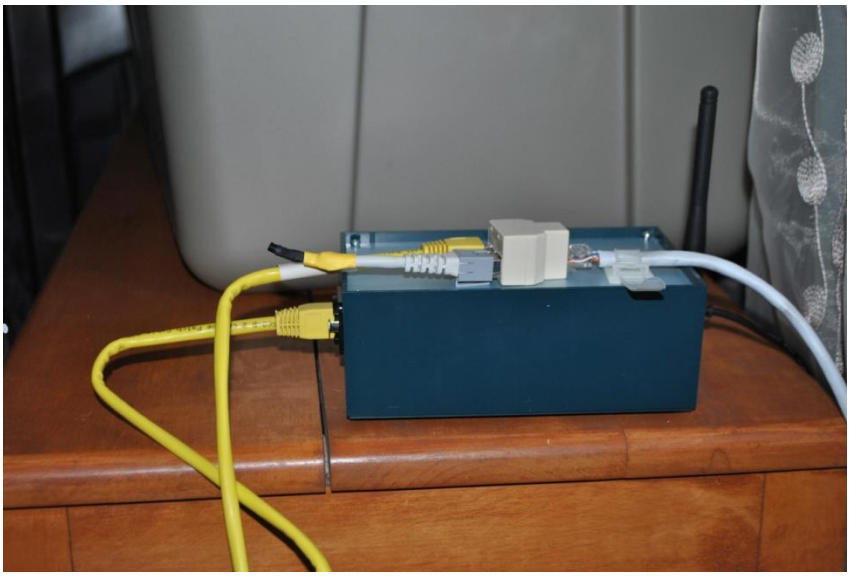
8. Survol de quelques développements:

- Importance de la documentation

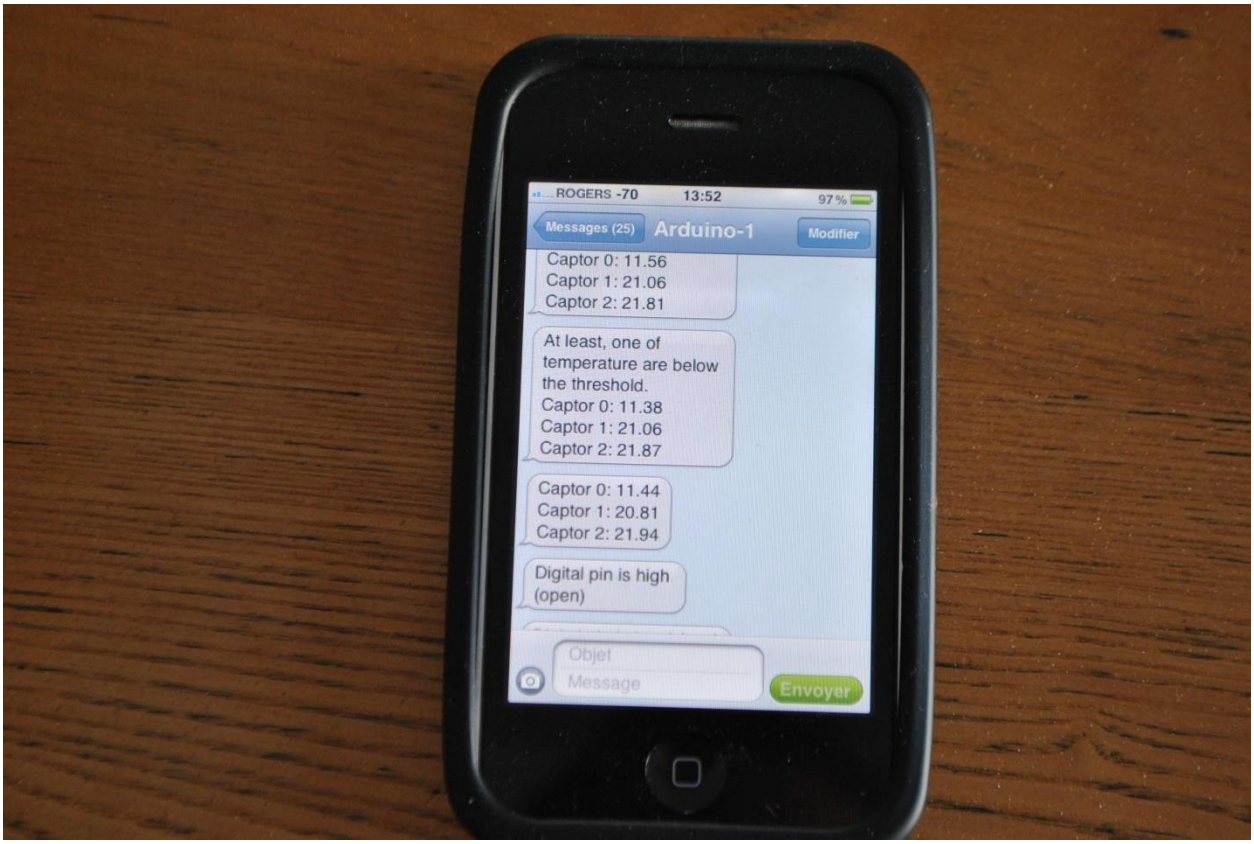
Mon premier développement Arduino: **une alarme de température à Québec, équipée d'une trappe à souris électronique**…

Il s'agit d'envoyer un SMS sur mon téléphone si:
- Une des températures passe en-dessous d'un seuil fixé
- Une souris se fait prendre par la trappe…

Il y a un petit interpréteur de commande qui permet de:
- Modifier le seuil pour l'envoi d'un SMS
- Retourner la liste des températures
- Donner les commandes disponibles



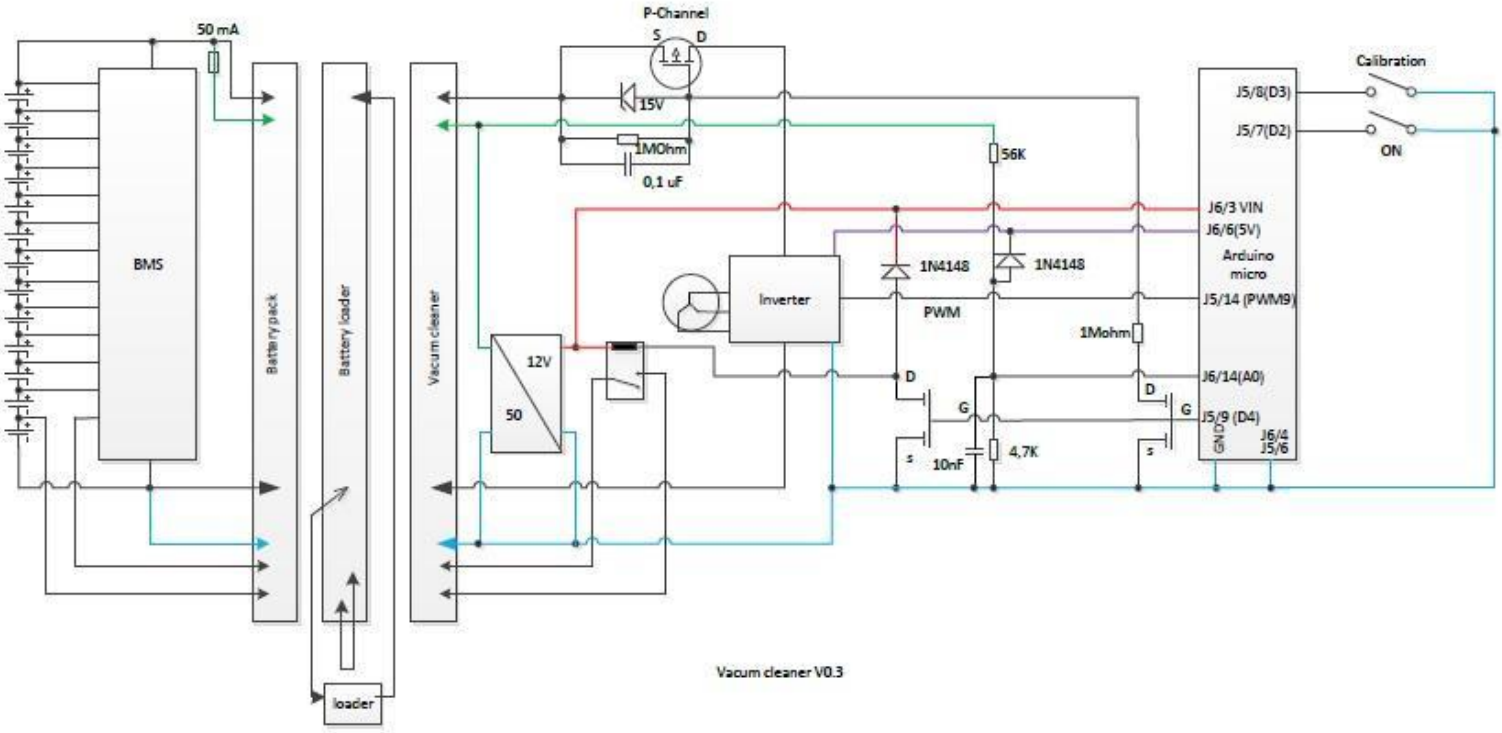

25M


ROGERS -70
13:52
97%
Messages (25)
Arduino-1
Modifier
Captor 0: 11.56
Captor 1: 21.06
Captor 2: 21.81
At least, one of temperature are below the threshold.
Captor 0: 11.38
Captor 1: 21.06
Captor 2: 21.87
Captor 0: 11.44
Captor 1: 20.81
Captor 2: 21.94
Digital pin is high (open)
Objet
Message
Envoyer

L’aspirateur autonome

C’est un aspirateur que l’on porte sur le dos.
Il est alimenté par le 220V, il fonctionne avec un moteur à collecteur classique.
On a enlevé le moteur et on l’a remplacé par un moteur brushless de modèle réduit.

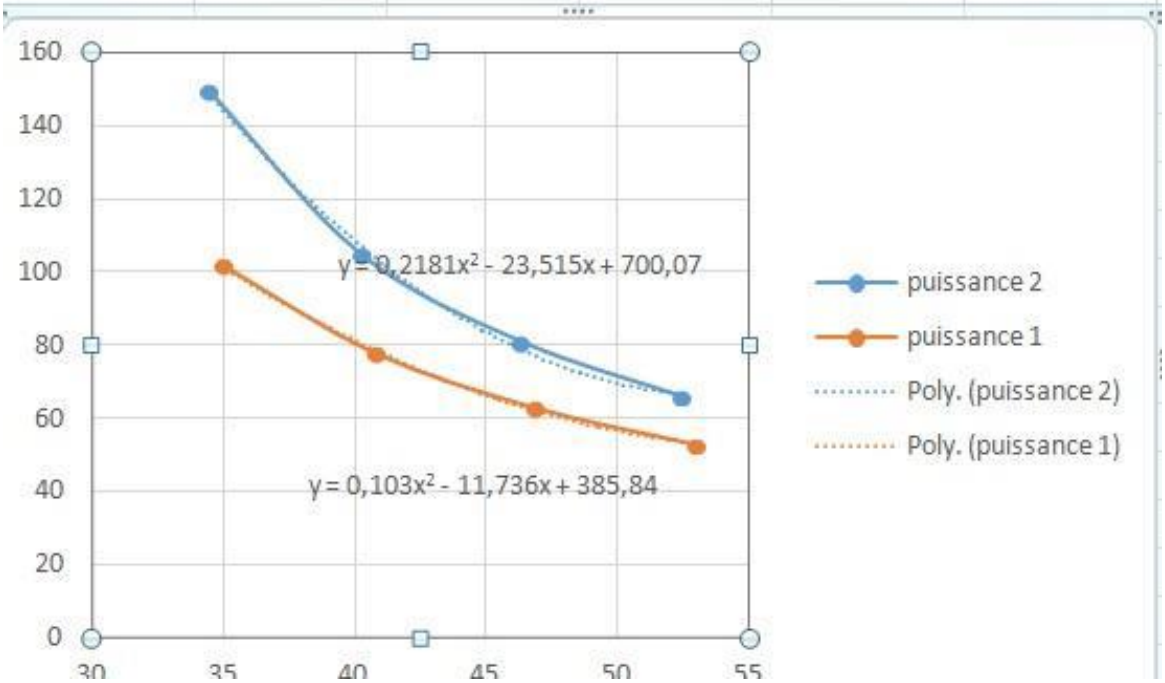
Schéma:



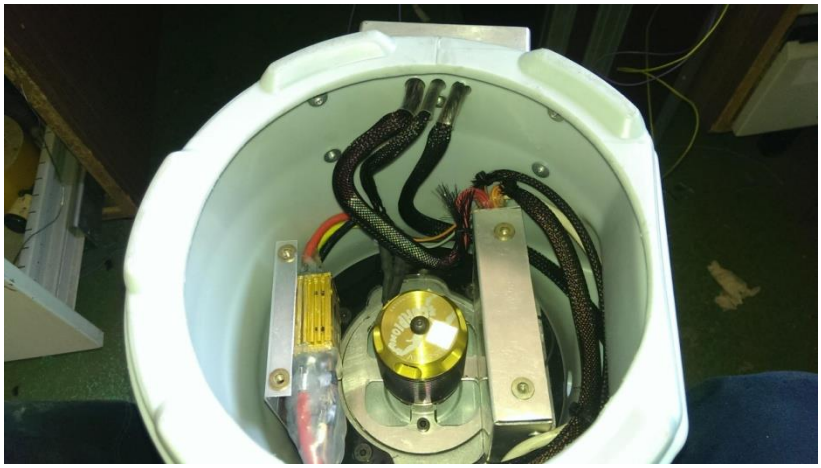
L’arduino:

- Génère les signaux PWM pour le servomoteur
- Commande par gâchette (2 vitesses)
- Asservi la vitesse de rotation en fonction de la tension batterie

puissance 2				
angle	tension	courant	vitesse	puissance
66	52,4	17,9	15770	937,96
81	46,3	19,2	15870	888,96
105	40,2	21	15750	844,2
150	34,4	23,7	15850	815,28




LAMB ELECTRIC
116111-00
Assembled in Mexico
220 Volts 50/60 Hz



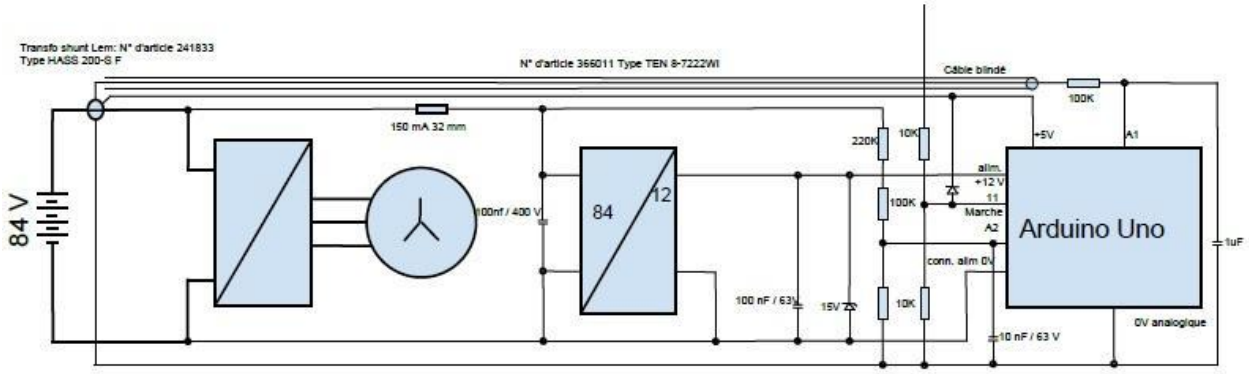
GPS tracker

Il s'agit de mémoriser sur une carte SD les déplacements d'un véhicule afin de pouvoir identifier les paramètres de consommation d'énergie par rapport à:

- L'importance du véhicule
- L'importance du conducteur.

Le développement fait l'acquisition de quelques signaux du véhicule:

- courant: mesure par transfo shunt jusqu'à 600 A
- tension: diviseur de tension
- contact
- position GPS, altitude



GPS tracker pour Opal version 1.1 24 juillet 2014 Thierry Vorms

1 acquisition (paramétrable) toutes les 2 secondes.

Un affichage sur 2 lignes permet de visualiser les principaux paramètres et d'étalonner l'équipement (offset du transfo shunt, diviseur de tension).

L'Arduino mémorise sur une carte SD les différentes valeurs.
La carte SD est relue sur un PC et les points sont importés dans un fichier Excel.
Des formules de trigonométrie permettent d'estimer les distances et de les déniveler entre 2 points.
Il est également possible d'importer les points dans Google Maps pour permettre d'effectuer un tracé des distances parcourues.

La documentation fournie est très complète. Elle motive les choix des technologies, explique la manière de calculer les différentes étapes, ainsi que la calibration.

9. Une réalisation effectuée par un non informaticien

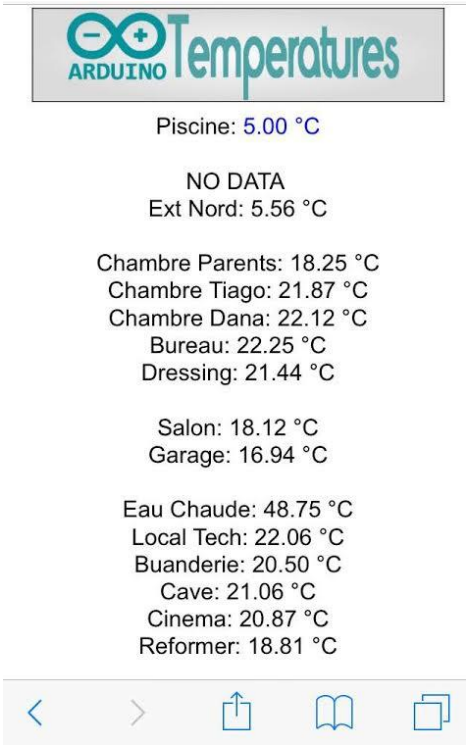
Pour montrer qu'il est possible d'effectuer des développements avec du matériel style Arduino sans être un expert, j'aurais voulu vous présenter une personne qui a mené à bien une telle réalisation.
Certes, cette personne s'est fait aider, mais elle est arrivée au bout et c'est cela qui compte.

Voici donc de quoi il s'agit:

Mesure de températures sur une dizaine de points dans une maison d'habitation.
Affichage sur une page WEB.
En fonction du seuil prédéfini pour chaque point, le fond de l'affichage change de couleur ce qui permet d'attirer l'attention en cas de problèmes.

La mesure de températures s'effectue avec des sondes DSB1820 dont la précision est de 12 bits. Ces sondes communiquent avec l'Arduino en utilisant le protocole OneWire.

Le protocole OneWire est disponible en téléchargement, de même qu'une librairie pour les sondes. En installant l'IDE pour l'Arduino, il y aura déjà des librairies pour les sockets et même un exemple de serveur WEB.



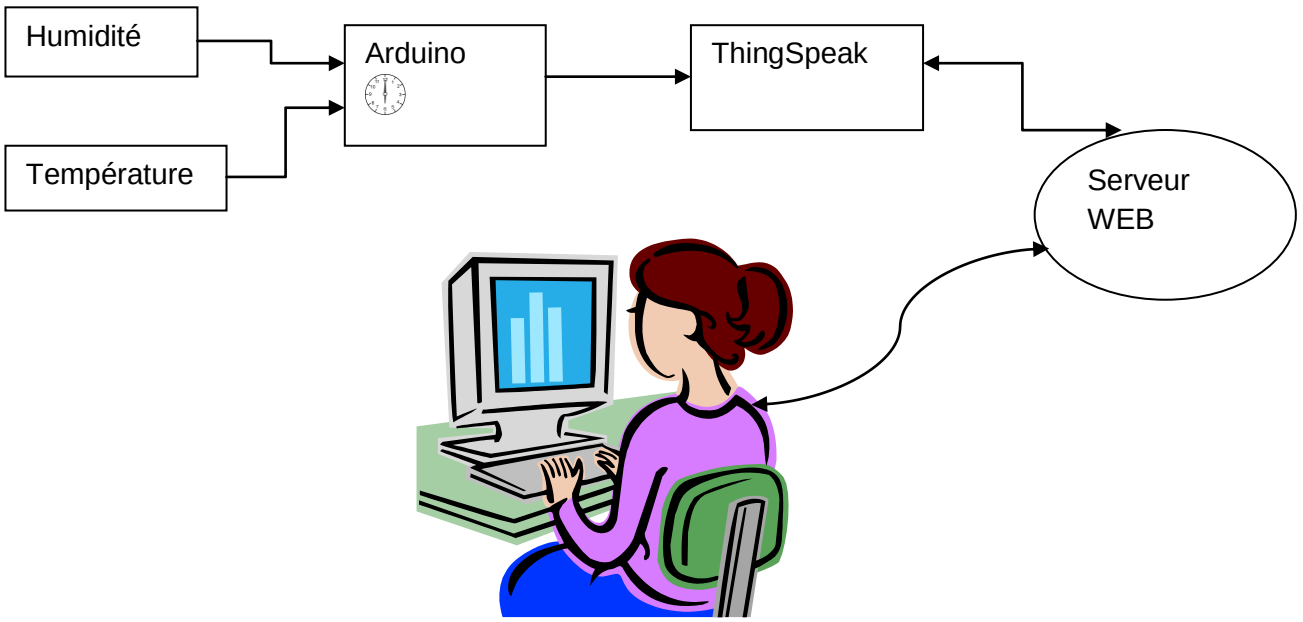
Mesure de température et humidité

Objectif:

Mesurer la température et l'humidité dans un studio.

Afficher les valeurs sous forme de courbes.

Avoir accès aux graphiques depuis partout sur la planète.



Contrôleur pour laveuse / sécheuse

Le but de ce boîtier est le contrôle de l'utilisation des machines.
Il a été décidé que la laveuse, le cas échéant la sécheuse, ne peut être utilisée que 2 fois par semaine pour un même client.

Le boîtier de commande comprend tout ce qu'il faut pour contrôler les machines.
Les prises volantes ont été supprimées pour éviter les fraudes….

L'utilisation des machines n'est autorisée qu'à des heures raisonnables (à partir de 8h. du matin jusqu'à 21h. le soir).

Le panneau frontal du boîtier contient 2 boutons et un espace pour passer le badge.

- Chaque badge doit être préalablement initialisé (avec un autre logiciel).
- Chaque badge a un identifiant unique.
- Il y a des badges qui ont le droit d'utiliser la sécheuse, d'autres non. Cela est écrit sur les badges.

Utilisation du boîtier

1. Appuyer sur le bouton *et le maintenir* pour sélectionner la machine (gauche: laveuse, droite sécheuse)
2. Passer le badge devant le lecteur et maintenir celui-ci quelques secondes, jusqu'à l'apparition d'un message bleu.

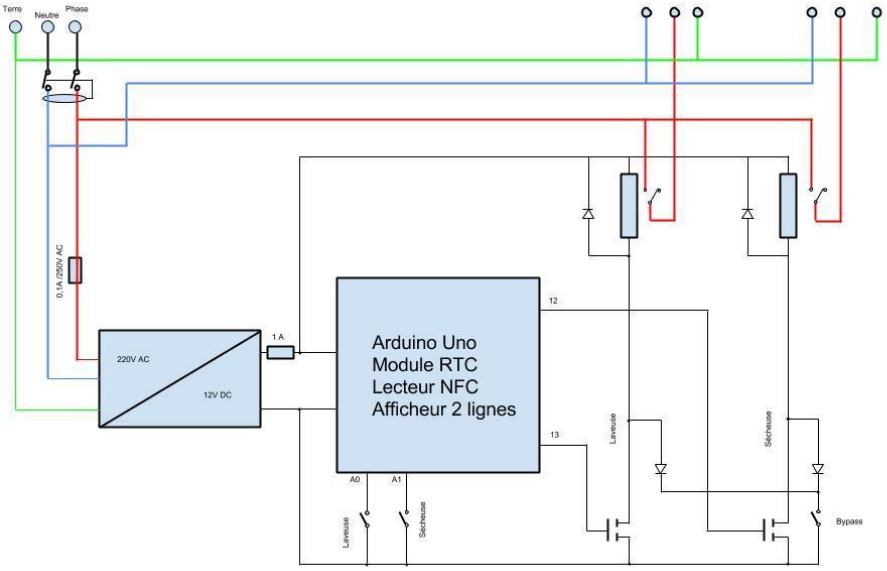
Si tout se passe bien..., il y a un affichage qui apparaît pendant quelques secondes:
Example: "Wash machine end" / "Drying cab. end" XX.XX

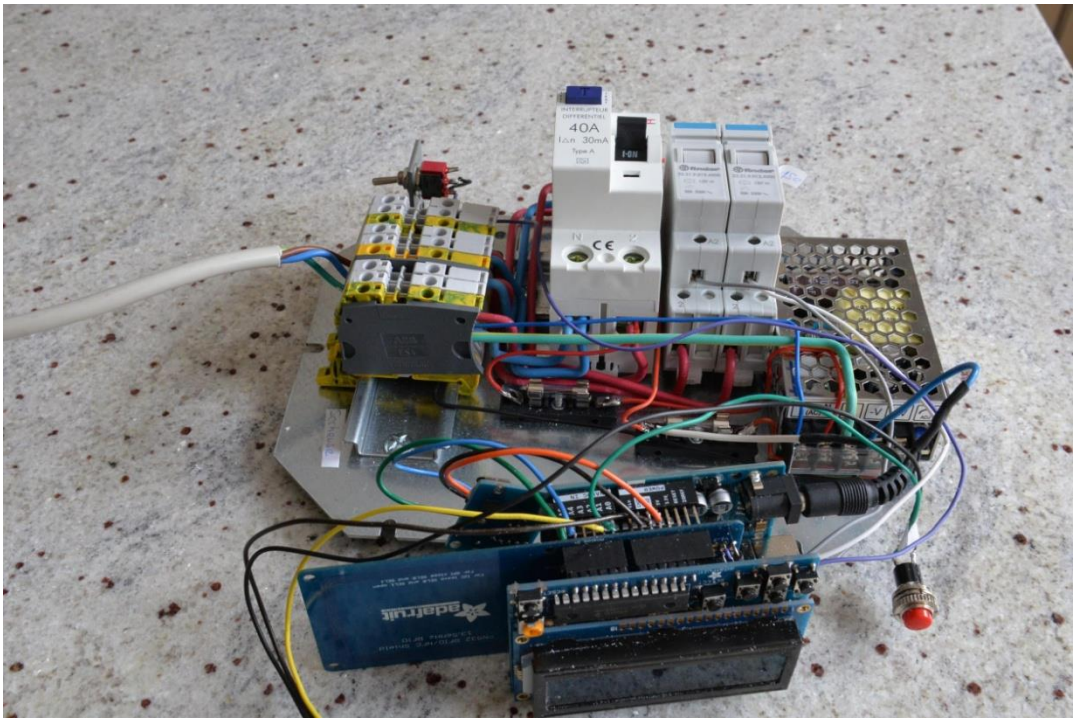
Dans le cas contraire, l'affichage passe au rouge et un message d'erreur est affiché.

2 autres logiciels ont été développés pour cette application:

- Préparation des badges (numéro unique, droit d'utilisation de la sécheuse, etc.)
- Gestion des badges (lecture des dernières utilisations, octroiement de droits supplémentaires, etc.

Schéma:


Terre
Neutre
Phase
0.1A /250V AC
220V AC
12V DC
1 A
Arduino Uno
Module RTC
Lecteur NFC
Afficheur 2 lignes
12
13
A0
A1
Laverie
Sécheuse
Bypass


40A


SIEMENS
B


Wash machine
Drying cabinet
2015/1/3
22:18 54

10.Etude de cas, la BatteryBox

Etude et support pour une boîte à batterie qui doit être capable de :

1) Le logiciel va compter les échanges d'énergie dans la batterie de manière à connaître en tout temps l'état de charge de la batterie. Cet état sera affiché sur un barre graphe constitué de 4 doubles leds. Le dernier groupe de leds pourra clignoter ce qui permet de disposer de 5 états distincts.

2) Détecter au moyen de contacts sur le connecteur de la boîte à batterie, le type de l'appareil connecté à celle-ci.
   À ce jour, les types suivants sont supportés :
   Chargeur, Vélo, Affichage LCD (I2C).
   Le logiciel doit être capable de détecter si le codage est incorrect.

3) Dans le but d'éviter de détruire les contacts du connecteur de la boîte à batterie lors de la connexion des différents appareils, des interrupteurs électroniques seront commandés avec un retard suffisant pour éviter tout arc électrique.

4) Pour diminuer les erreurs de mesures pour les faibles courants, en- dessous d'un seuil défini à l'avance, c'est le courant de repos de l'appareil connecté (mesuré à l'avance) qui sera utilisé.

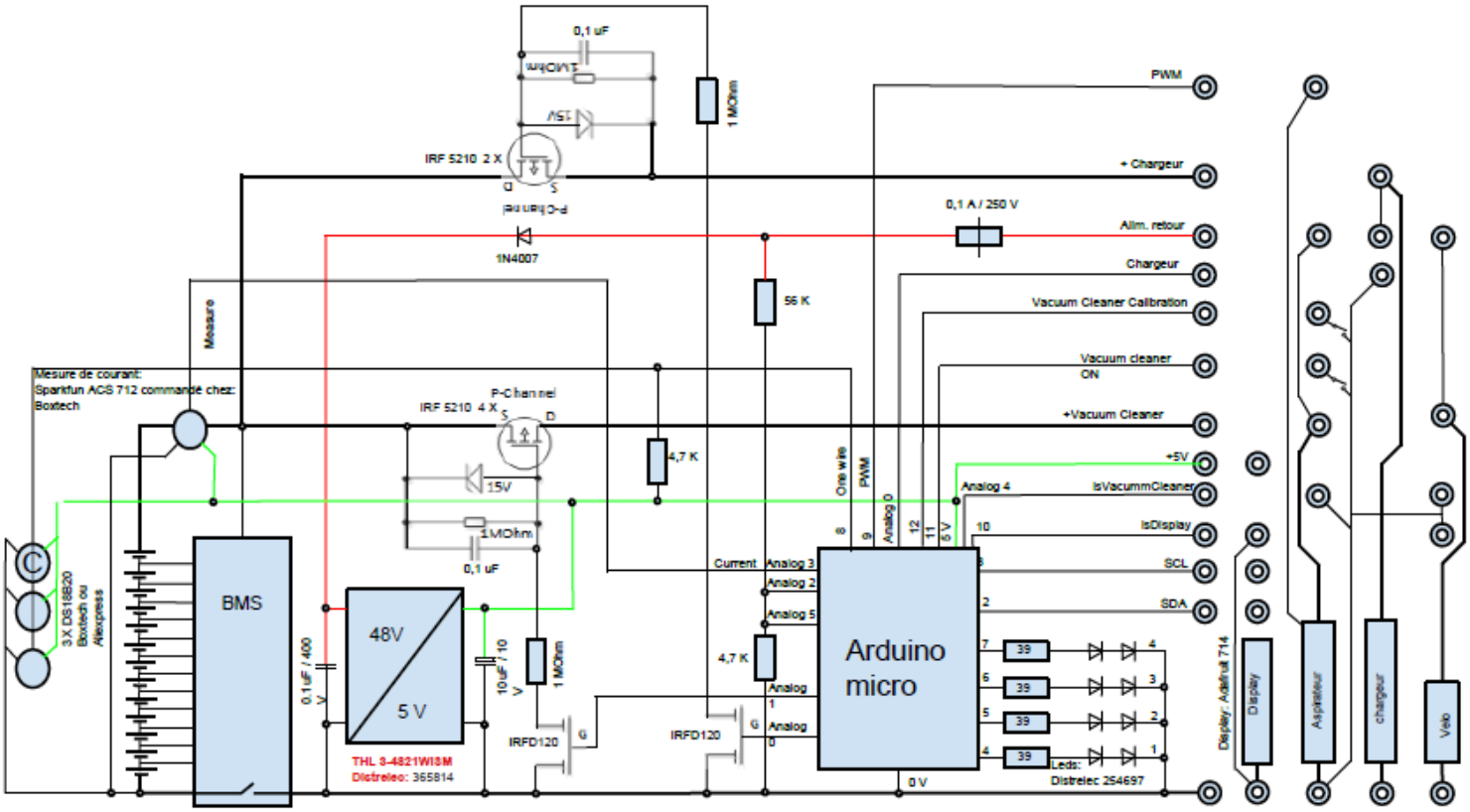
5) Le logiciel surveillera la température de la batterie en 3 points au moins et au besoin la connexion avec l'extérieur sera déclenchée temporairement.

6) Le logiciel effectuera des mesures de minimas et maximas de différentes variables telles que tension, courant et température. Il sera possible de lire certaines d'entre elles sur l'affichage LCD**.**

7) Les variables seront sauvegardées lors de la déconnection du périphérique de manière à assurer une consommation nulle au repos et de récupérer les dites valeurs à la mise sous tension.

8) Un contrôle de l'intégrité des valeurs sauvegardées sera effectué à chaque démarrage. Au besoin, les leds vont clignoter au démarrage, générant un code qui aidera à la recherche d'erreurs.

Schéma:



Partie logiciel:

Pas moins de 4 logiciels ont été développés pour cette réalisation

- **Temps réel:** c'est le logiciel qui tourne normalement sur la carte Arduino.
- **Tests:** c'est ce qui permet de tester les différentes fonctionnalités du module:
  - La reconnaissance du périphérique connecté ainsi que les 2 boutons poussoirs pour l'aspirateur.
  - Les interrupteurs de puissances.
  - L'affichage à leds de la capacité actuelle de la batterie.
  - La génération du signal PWM pour la commande du servomoteur de l'aspirateur.
  - Le générateur de code d'erreurs sur l'affichage à leds.

    - La mesure des températures.
    - L'affichage de tous les paramètres mémorisés dans l'Arduino.
    - Le reset des minimas et maximas.

- Préparation de l'EEPROM. Ce logiciel permet d'initialiser tous les paramètres afin de permettre d'épargner le temps de calibration.

- Logiciel de calibration

Le logiciel de calibration sert à saisir les valeurs de référence qui seront mémorisées dans la partie non volatile de l'Arduino.

Voici les différents paramètres qui sont ajustés avec ce logiciel:

1) Calibration de l'offset de la mesure de courant. Le capteur de courant fourni environ la moitié de la tension lorsque le courant mesuré est nul. Ainsi, on peut mesurer le courant dans les 2 sens. Il est nécessaire de calibrer cette valeur d'offset.
2) Calibration de l'échelle de mesures du capteur de courant. De part le principe de mesures du capteur, il est nécessaire de calibrer cette valeur.
3) Calibration de la mesure de tension. Afin d'annuler les erreurs causées par l'imprécision des résistances de mesures et de la tension de référence du convertisseur analogique / digital, il est nécessaire de calibrer cette valeur.
4) Calibration de la période d'échantillonnage. C'est le temps entre 2 mesures. Un temps trop court induira des imprécisions dans le calcul de la capacité de la batterie. On partira avec une valeur de 2 secondes.
5) Calibration de la capacité de la batterie. C'est la capacité en Ah de la batterie.
6) Seuil de mesure du courant. C'est le courant en-dessous duquel on utilisera la consommation à vide du périphérique en lieu et place de la mesure du courant par le capteur. C'est le faible courant consommé pour chaque périphérique.
7) Température maximum en charge et en décharge. Au-delà des seuils, la batterie est mise en protection.

8) Tension de floating. C'est le seuil de tension pour lequel on considère que la batterie est complètement chargée. Dans ce cas, on force l'état de charge à la capacité nominale de la batterie.

Ce logiciel permet également de:

1) Simuler un ampèremètre afin de contrôler les performances. (offset et échelle)
2) Simuler un voltmètre (échelle).
3) Imprimer sur la console toutes les valeurs stockées dans la partie non volatile de l'Arduino.
4) Tester l'affichage LCD avec ses boutons

Développement logiciel

Afin de ne pas avoir à répéter entre les 4 logiciels des parties communes, des librairies ont été développées:

- **BatteryBox:** Tout ce qui a attrait à la batterie, accès aux données, etc.
- **Toolbox:** Quelques méthodes pour afficher des nombres flottants, entrées de valeurs, etc.

Difficultés rencontrées:

Avec ce développement, on atteint les limites de mémoire programme.
Cela a été résolu de la manière suivante:

- Développement de 4 programmes distincts au lieu d'un seul.
- Utilisation des variables de compilations. Comme ces variables ne sont pas transmises entre la compilation du logiciel et des librairies, il est nécessaire de répéter la définition des switchs ce qui rend le développement moins aisé.

Evolution:
Communication avec un autre Arduino en utilisant le bus I2C pour pouvoir modifier les paramètres de la motorisation, calculer la vitesse, gérer les phares, etc.

11.Où acheter le matériel ?

- En Suisse: http://shop.boxtec.ch/index.php
  - Plusieurs fournisseurs de cartes microcontrôleurs
  - Capteurs

- En France: http://boutique.semageek.com/fr/
  - Composants électroniques
  - Cartes Arduino
  - Shields
  - Capteurs
  - Boîtiers

12. Arduino en résumé:

Les +

- C’est une plateforme simple, pas cher
- Le logiciel de développement est gratuit
- L'environnement de développement évolue
- La communauté est énorme.
- Beaucoup de cartes d’extension
- Beaucoup de librairies disponibles

Les -

- Puissance limitée de calcul.
- Mono thread.
- Arduino a fait des choix pour nous.
- Difficulté de gérer les modes sleep, wake.up interrupt.