

Télé-Information

Compteur

Electrique/Gaz/Eau

Dossier d'étude	
Auteur	Mami Wadi
Date de mise à jour	

Version				
N°	Date	Auteur	Nature de la modification	Statut
1.0.0	02/08/2012	Mami Wadi	Création du document	A valider

Blog <https://einsteinbohr.blogspot.com>

Github : <https://www.github.com/didipostman>

Email : wmami@steg.com.tn / wady.mami@gmail.com

Table des matières

1. Caractéristiques.....	3
1.1 Versions de référence.....	3
2. Introduction.....	4
3. Compteur Electrique à disque (http://community.cosm.com/node/65).....	5
4. Compteur Electrique (électronique).....	10
5. Arduino http://arduino.cc/fr/Main/DebuterIntroduction.....	19
5.1.1 Qu'est-ce qu'Arduino?.....	19
5.1.2 Pourquoi Arduino ?.....	19
6. Arduino GSM Shield	21
7. Conclusion.....	22

1. Caractéristiques

1.1 Versions de référence

Environnement	Description

2. Introduction

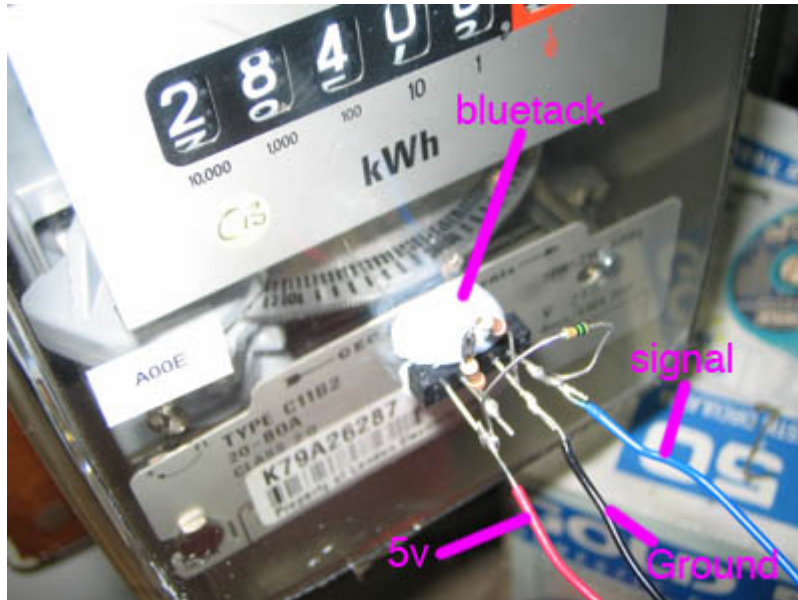
L'idée de collecter les informations à distance (Surtout la consommation en énergie) des compteurs électrique/Gaz/eau m'est venue à l'esprit depuis un bon moment depuis 2012. Cette idée a surgi de nouveau lors de la recherche d'un projet de fin d'études pour un stagiaire. Bref comment faire pour relever la consommation en électricité à distance sans avoir à se déplacer et sans avoir à changer les compteurs pour des compteurs dit intelligents. Et ce tout en se prenant en considération que la STEG dispose de deux types de compteurs (compteur à disque majoritaire et compteur électronique).

Sans pour autant se vanter d'avoir réinventé la roue la solution se trouvait bel et bien sur la toile (internet), il suffisait de googler un peu.

Dans ce qui suit je présenterai en bref la solution technique. Toutefois une étude de faisabilité et de coût s'impose.

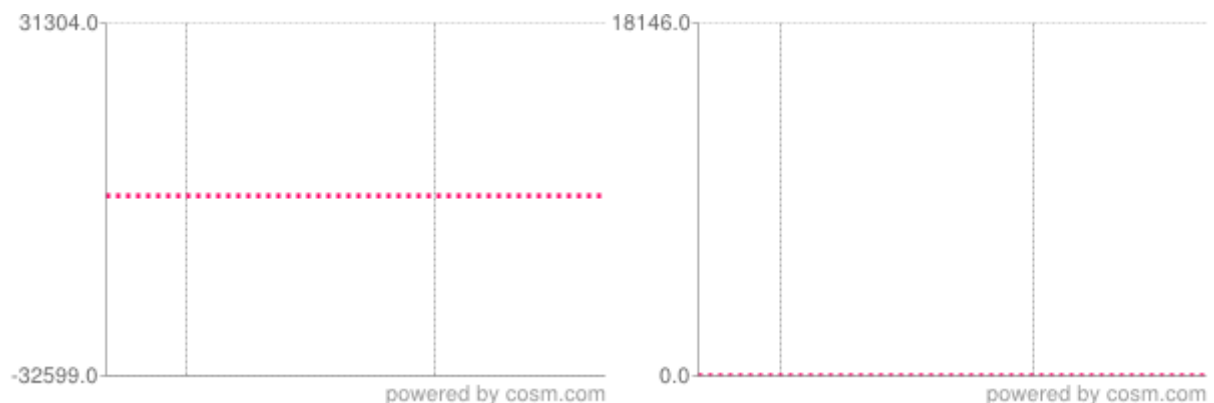
3. Compteur Electrique à disque (<http://community.cosm.com/node/65>)

- [Arduino](#) (voir 4)
- [electriciy mètres](#)
- [contrôle de l'énergie](#)
- [pachube](#)



Voici un moyen rapide et pas cher pour connecter votre compteur d'électricité à Pachube, afin de pouvoir incorporer des graphiques dans les pages Web, de le surveiller de loin, ou tout simplement pour montrer aux autres ce que votre consommation est. Il utilise un Arduino et un «capteur d'objet de réflexion». Cela fonctionne sur les compteurs au Royaume-Uni qui ont un disque visible de filature; vous remarquerez que le disque a une bande noire - notre capteur va mesurer la rapidité avec laquelle le disque est en rotation, qui est proportionnelle à la consommation d'énergie actuelle. Il peut travailler dans d'autres pays qui ont le même type de disque visible, et vous pourriez obtenir que cela fonctionne à l'aide d'un mètre avec un clignotement de la LED aussi. Bien sûr, avec un bouclier ethernet il est alors assez simple de connecter votre compteur d'électricité à Pachube.

Voir en action ici: <http://www.pachube.com/feeds/504>



(1) Puissance en Watts; (2) Puissance moyenne sur 15 minutes

Tout d'abord, vérifiez que votre appareil possède un disque en rotation et une bande noire. Il pourrait ressembler à quelque chose un peu comme ceci:



Ensuite, assurez-vous que vous pouvez savoir comment le disque est calibré. Sur le mien il a déclaré: «200 tours par kWh». Donc, je savais que si je pouvais mesurer le temps entre les bandes noires (les révolutions du disque par exemple), alors je pourrais savoir combien de kilowatts sont utilisés à un moment donné:



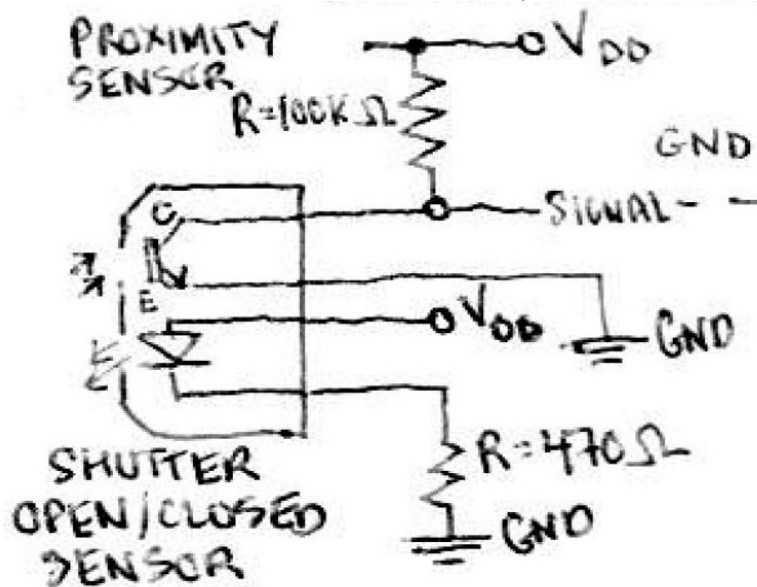
Pour détecter les rayures, j'ai utilisé un «capteur d'objet de réflexion» que vous pouvez trouver à RS ici: <http://uk.rs-online.com/web/search/searchBrowseAction.html?method=getPro...>



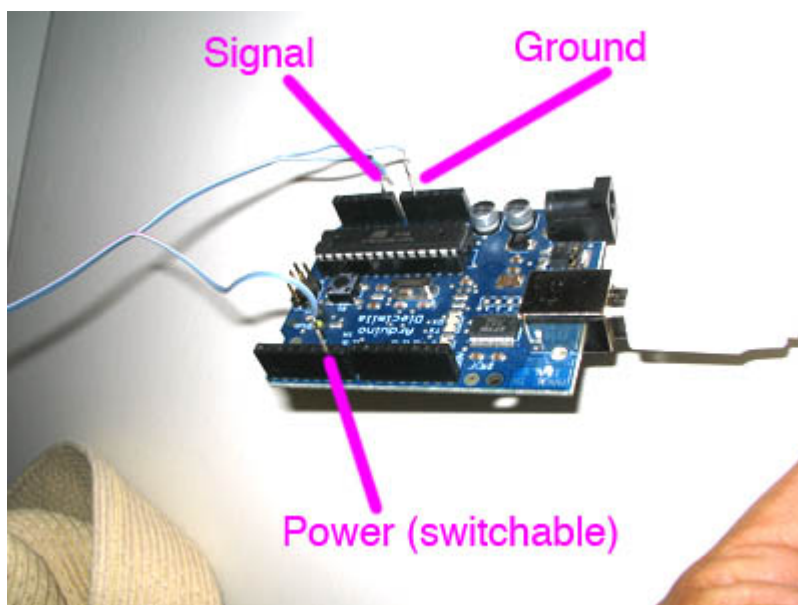
et ensuite collé sur l'utilisation de l'bluetack. C'est probablement la partie la plus difficile, parce que vous avez pour le faire pointer exactement dans la bonne direction afin qu'elle frappe le disque et le disque reflète vers le capteur - pas trop élevé, ni trop bas.



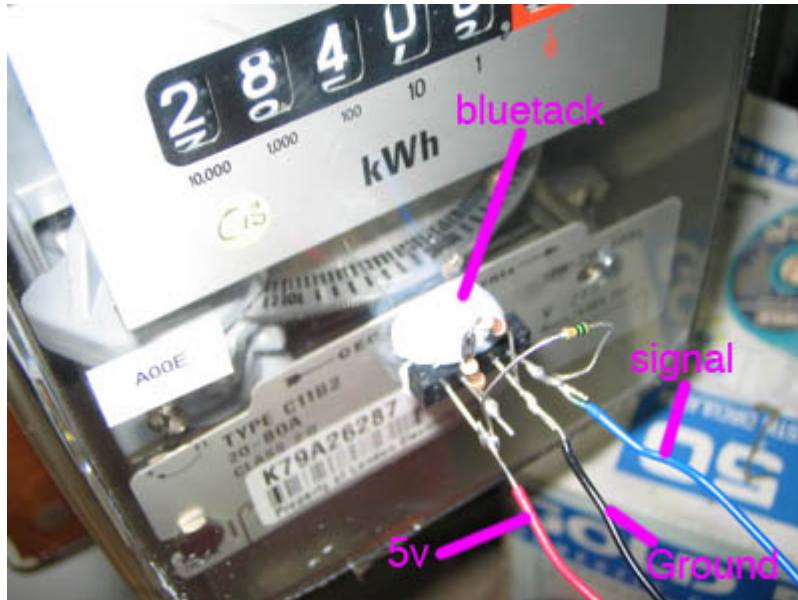
Cela a été câblé selon le schéma suivant (merci à Chris Leung pour les valeurs de résistance):



L'Arduino lit la valeur analogique du signal - vous pourriez modifier pendant des heures pour obtenir les résistances exactement droite de sorte que vous obtenez une grande définitive (lorsque la bande noire est présente) et un plus bas défini (lorsqu'il n'est pas présent), mais au lieu que je juste regardé les valeurs analogiques venant de la broche de signal et ensuite déterminé (en comparant les moyennes) quand il ya eu un changement soudain. J'ai aussi l'alimentation du capteur en utilisant une épingle pour que je puisse le mettre en marche et en arrêt, plutôt que de la mise sous tension directement à partir de l'alimentation 5V. Je l'ai fait parce que j'ai trouvé le niveau du signal errait un peu quand le côté transistor du capteur a été alimenté en permanence. Ainsi, au lieu, je le pouvoir sur le capteur d'environ 50 ms, puis vérifiez la valeur et de les traiter, puis éteignez pendant 50 ms.



Vous devez vous assurer que vous avez une nette différence dans les valeurs (les miennes étaient environ 300 en l'absence de bande était présent, et environ 400 quand il était présent et je regarde pour un changement en moyenne environ 1,1) et s'il n'y a pas, alors le capteur n'est pas correctement orienté.



Après avoir déterminé le nombre de secondes écoulées depuis la dernière rayure (intervalle), et en prenant notre valeur pour "tours par kWh" (qui était de 200 dans mon cas), nous pouvons travailler sur la consommation d'énergie actuelle à l'aide de ce qui suit:

```
currentPower = (3600 / interval) / revsPerkWh;
```

(Le 3600 est le nombre de secondes dans une heure).

Bien que, dans mon code que j'ai utilisé ce qui suit:

```
currentPower = (1000.0 * 3600000.0 / interval) / revsPerkWh;
```

où j'ai utilisé 3600000 que le nombre de millisecondes * Les * en une heure, et je multiplie par 1000 nouveau parce que je suis intéressé par le nombre (en watts) W plutôt que kW (kilowatts).

Rattaché ici: <http://community.pachube.com/files/officeSensorsExample.zip> est l'original Arduino 11 Code. Cela rend l'utilisation de <http://www.nuelectronics.com/estore/?p=14> le bouclier NuElectronics ethernet, bien que n'importe quel autre écran peut également être utilisé, pour servir ces valeurs à Pachube et comprend des valeurs pour un capteur de température, capteur d'humidité et le capteur de lumière.

Attachement	Taille
officeSensorsExample.zip	

4. Compteur Electrique (électronique)

Voir documents

Notice d'installation et de programmation des: Shield téléinfo 2 compteurs pour Arduino

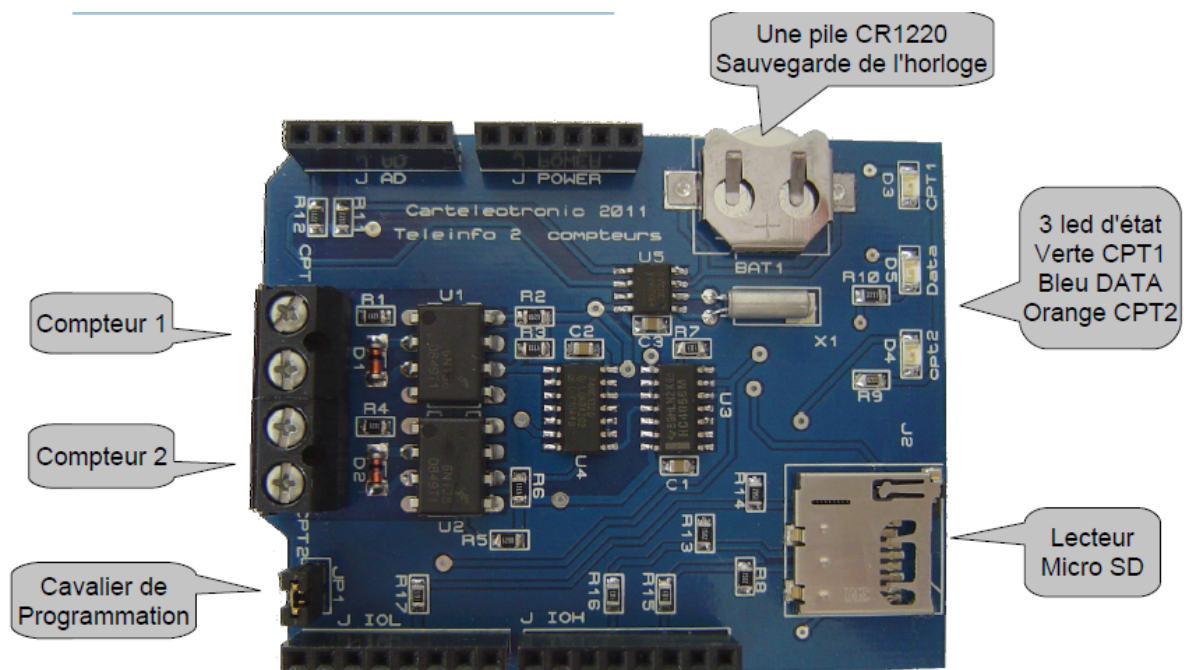
document révision 1.0

1- Présentation :

Ce shield, ou carte d'extension, permet d'interfacer rapidement et facilement votre Arduino à votre (vos) compteur(s) électriques.

Elle permet en plus de lui adjoindre un shield Ethernet ou Wifi (ou autre) pour transmettre les

- fichiers ou la lecture en direct de vos compteurs. LEINFO 2CPT ARDUINO VA1.pdf



Les borniers **Compteur 1** et **Compteur 2** servent à relier la carte aux compteurs électriques sur les bornes teleinfo de ces derniers.

Le **cavalier de programmation** permet de programmer la carte Arduino avec le Shield teleinfo

sans avoir à enlever ce dernier (car il utilise les lignes RX et TX de la carte Arduino les mêmes

que pour la programmation en USB)

Le lecteur de **Micro SD** permet la sauvegarde des valeurs lues sur vos compteurs et ainsi de

les traiter par la suite.

La pile sert quant à elle à sauvegarder l'heure et la date du circuit DS1307 en cas de coupure

de l'alimentation, cette pile est livrée avec la carte mais pas installée, bien respecter les

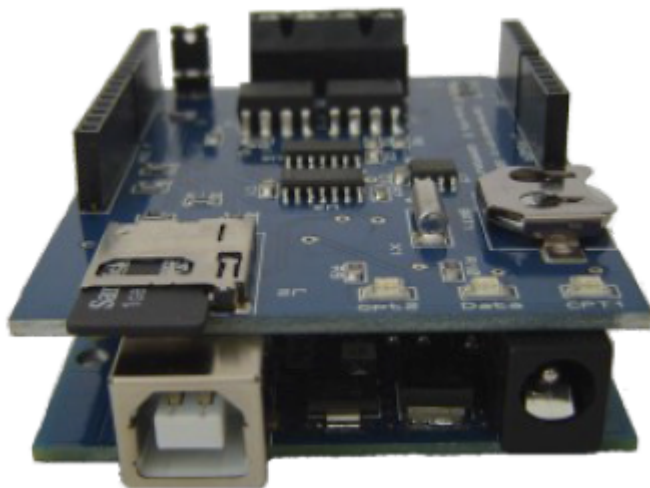
polarités :

Le + (marqué sur la pile) doit se trouver sur le dessus lors de l'insertion de cette dernière dans

son support.

2- Installation du shield :

Le shield se « plug », s'insère, directement sur la carte Arduino.



Installez votre carte mémoire (micro SD non livrée avec la carte) et la pile de sauvegarde.

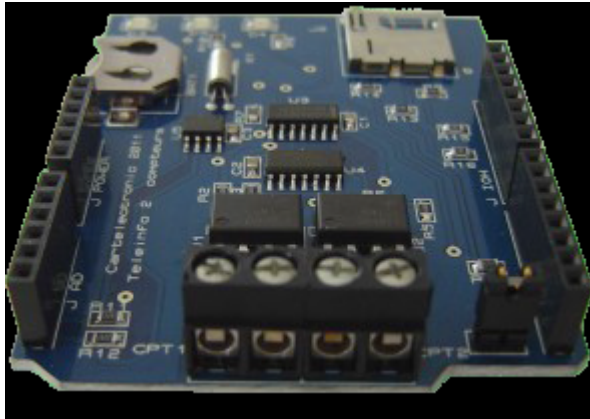
Connectez le compteur de votre fournisseur sur les bornes CPT1, et votre compteur de

production (ex solaire) sur les bornes CPT2, vous pouvez aussi brancher deux compteurs de

consommation avec deux abonnements différents, ou un sous compteur pour surveiller une

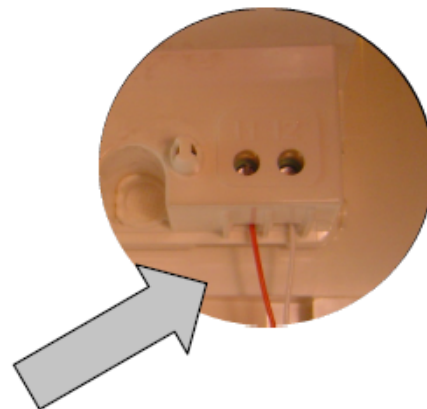
PAC ou un plancher chauffant électrique...

Voir



Voir les photos pour repérer les bornes téléinfo du compteur.
*Il n'y a pas de polarités (de sens) à respecter lors du câblage des fils pour
 relier le boîtier au*

compteur (tension alternative)



ATTENTION vous travaillez au voisinage du 240V ! Prenez vos précautions, nous vous
 conseillons de faire brancher cette carte par un électricien !

2- Installation des logiciels

Téléchargez :

<http://www.carteselectronic.fr/>

Dans la rubrique "Téléchargement" puis dans "Shield Arduino" les fichiers suivants :

Schémas TELEINFO 2CPT Arduino

Teleinfo_Arduino_pde.rar

<http://arduino.cc/fr/Main/DebuterInstallation>

Arduino-0018-fr (Pour la version Française)

<http://arduino.cc/en/Main/Software>

Arduino-0018-fr (Pour la version Anglaise)

Copier ces fichier dans : C:/Programmes / Arduino Programme perso »

Installz :

Sous « C:/Programmes » créer les répertoires :

Arduino Programme perso

Arduino-Fr (Si vous utilisez la version Française)

Arduino-A (Si vous utilisez la version Anglaise)

Installez la suite logiciel Arduino que vous préférez (version Anglaise ou Française)
Copiez la librairie de l'horloge temps réel : RTCLib qui se trouve dans le ZIP :

« Teleinfo_Arduino_pde »

L 'installation des librairies se fait en copiant le répertoire « RTCLib » dans le répertoire :

« arduino-Fr/Libraries » ou « arduino-A/Libraries »

Pour plus d'infos et des cours sur l'Arduino consulter :

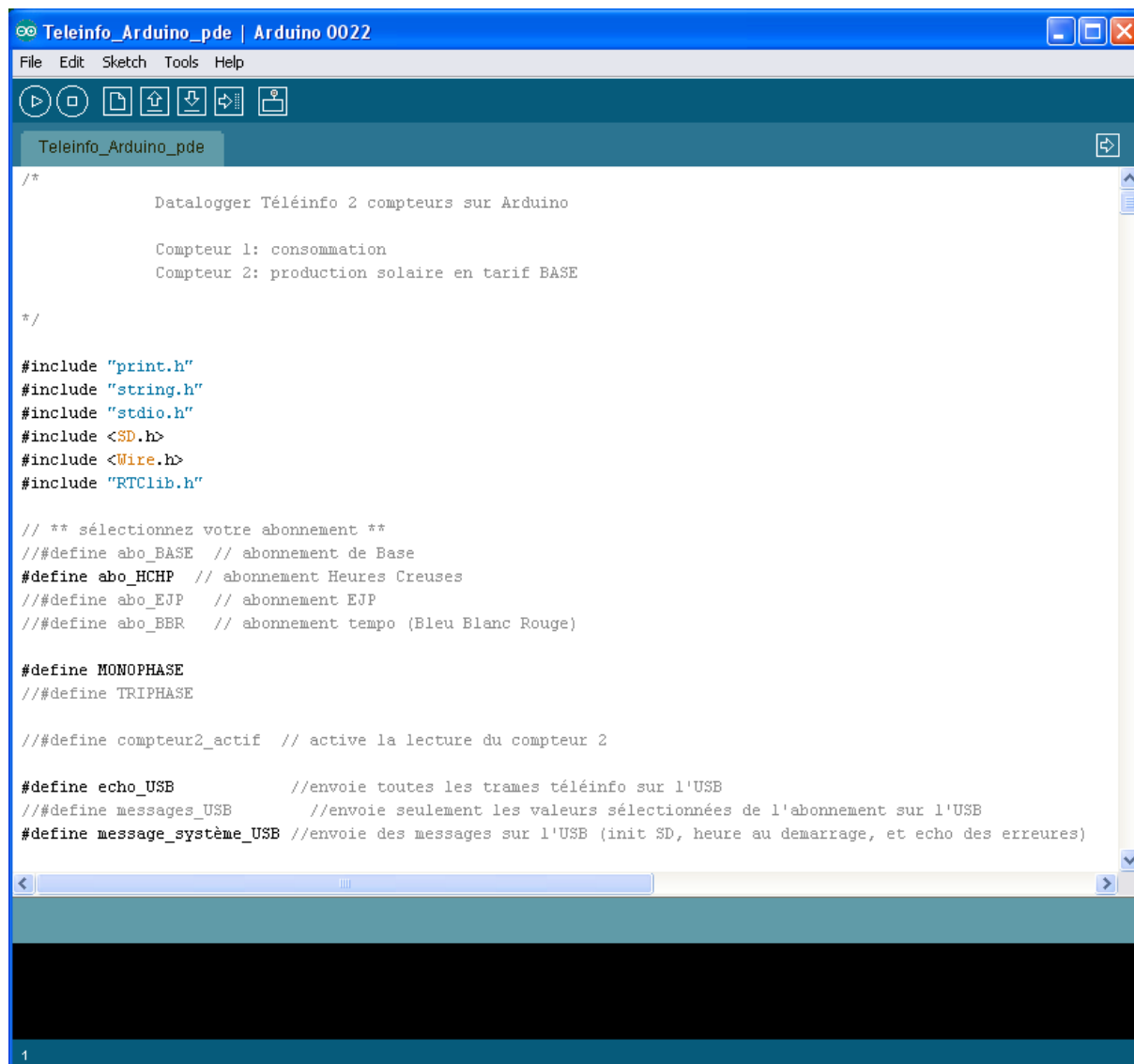
http://www.mon-club-elec.fr/pmwiki_reference_arduino/pmwiki.php?n=Main.HomePage

Programmez :

Une fois le logiciel lancé :

- Cliquer sur « Fichier » (« File »)
- Sélectionner « Mes programmes » (« Skechbook »)
- Puis Tele_arduino_pde

Le programme est chargé



```
/*
    Datalogger Téléinfo 2 compteurs sur Arduino

    Compteur 1: consommation
    Compteur 2: production solaire en tarif BASE
*/

#include "print.h"
#include "string.h"
#include "stdio.h"
#include <SD.h>
#include <Wire.h>
#include "RTCLib.h"

// ** sélectionnez votre abonnement **
// #define abo_BASE // abonnement de Base
#define abo_HCHP // abonnement Heures Creuses
// #define abo_EJP // abonnement EJP
// #define abo_BBR // abonnement tempo (Bleu Blanc Rouge)

#define MONOPHASE
// #define TRIPHASE

// #define compteur2_actif // active la lecture du compteur 2

#define echo_USB // envoie toutes les trames téléinfo sur l'USB
// #define messages_USB // envoie seulement les valeurs sélectionnées de l'abonnement sur l'USB
#define message_système_USB // envoie des messages sur l'USB (init SD, heure au démarrage, et echo des erreurs)
```

Ce programme est un '**Datalogger Téléinfo 2 compteurs sur Arduino**' de base que vous devez paramétrer suivant votre type d'abonnement et si vous possédez un compteur de production.

Ce code enregistre toutes les minutes les index de vos compteurs sur une carte mémoire de type micro SD.

Le logiciel crée un fichier CSV par jour ainsi qu'un fichier dit Annuel qui lui contient les index de vos compteurs enregistrés une fois par jour à 23h59.

Ces fichiers sont utilisables par le logiciel « Consult teleinfo » disponible en libre téléchargement sur notre site.

Vous n'avez pas de code à écrire, sauf si vous voulez modifier des éléments ou vous amuser à piloter des relais etc... , il suffit juste de commenter ou de décommenter des lignes de configuration !

Pour commenter une ligne ajoutez `// #define` (la ligne ne sera pas exécutée par le compilateur)

Pour décommenter une ligne enlevez `//` la ligne sera prise en compte par le compilateur.

Exemple de configuration:

Vous avez une abonnement EJP, en monophasé et des panneaux solaires,

voici à quoi devrait ressembler le début du programme :

```
// ** sélectionnez votre abonnement **  
//#define abo_BASE // abonnement de Base  
//#define abo_HCHP // abonnement Heures Creuses  
#define abo_EJP // abonnement EJP  
//#define abo_BBR // abonnement tempo (Bleu Blanc Rouge)  
#define MONOPHASE  
//#define TRIPHASE  
#define compteur2_actif // active la lecture du compteur 2  
#define echo_USB //envoie toutes les trames téléinfo sur l'USB  
//#define messages_USB //envoie seulement les valeurs sélectionnées de l'abonnement sur l'USB  
  
#define message_système_USB //envoie des messages sur l'USB (init SD, heure au démarrage, et  
echo des erreurs
```

Ensuite vérifiez le code en cliquant sur :



Pour lancer la programmation de la carte vous devez enlever le cavalier de programmation de la carte teleinfo puis cliquer sur :



N'oubliez pas de remettre en place le cavalier avant de rebooter la carte.

Après initialisation de la carte Arduino la led verte doit s'allumer et en même temps la led bleue, cette dernière indique la présence d'un signal de téléinfo.

Après un certain temps, dépendant de votre type d'abonnement, la led verte doit s'éteindre et la led orange doit s'allumer, elle aussi accompagnée de la led bleue.

Le basculement d'un compteur à l'autre est fait lorsque le logiciel a obtenu les valeurs qu'il doit enregistrer (avec une vérification du checksum pour être certain de la validité des données)

Une fois la lecture du compteur 2 terminée, led orange, le logiciel bascule à nouveau sur le compteur 1, led verte, et ainsi de suite.

Pour exploiter les données il vous suffit d'extraire la micro SD et de copier les fichiers sur un PC et de remettre cette carte dans l'Arduino et de remettre l'alimentation.

*PS : L'horloge RTC de la carte est mise à l'heure par la programmation de la carte, lors de la première mise en route, après insertion de la pile.
Si vous changez la pile vous devez reprogrammer la carte pour avoir l'heure du PC sinon vous vous retrouverez avec l'heure programmée précédemment et forcément dépassée.*

Vous pouvez, bien évidemment, rajouter une fonction dans le logiciel qui vous permettra de régler l'horloge.

Pannes :

- La led bleue ne s'allume pas !

Vérifiez vos connexions aux compteurs.

- La led verte est toujours allumée (avec la bleue) et ne bascule pas sur le compteur 2

Vérifiez la configuration du programme par rapport à votre abonnement électrique.

- Il n'y a pas de fichier sur la carte mémoire !

-Vérifiez que vous avez bien remis le cavalier de programmation,

-Vérifiez que la carte est bien insérée

-Vérifiez que la carte n'est pas pleine ou pas formatée

Contenus

- une carte shield « **Téléinfo 2 compteurs Arduino** »

- une pile CR1220

Détails techniques:

- interface USB

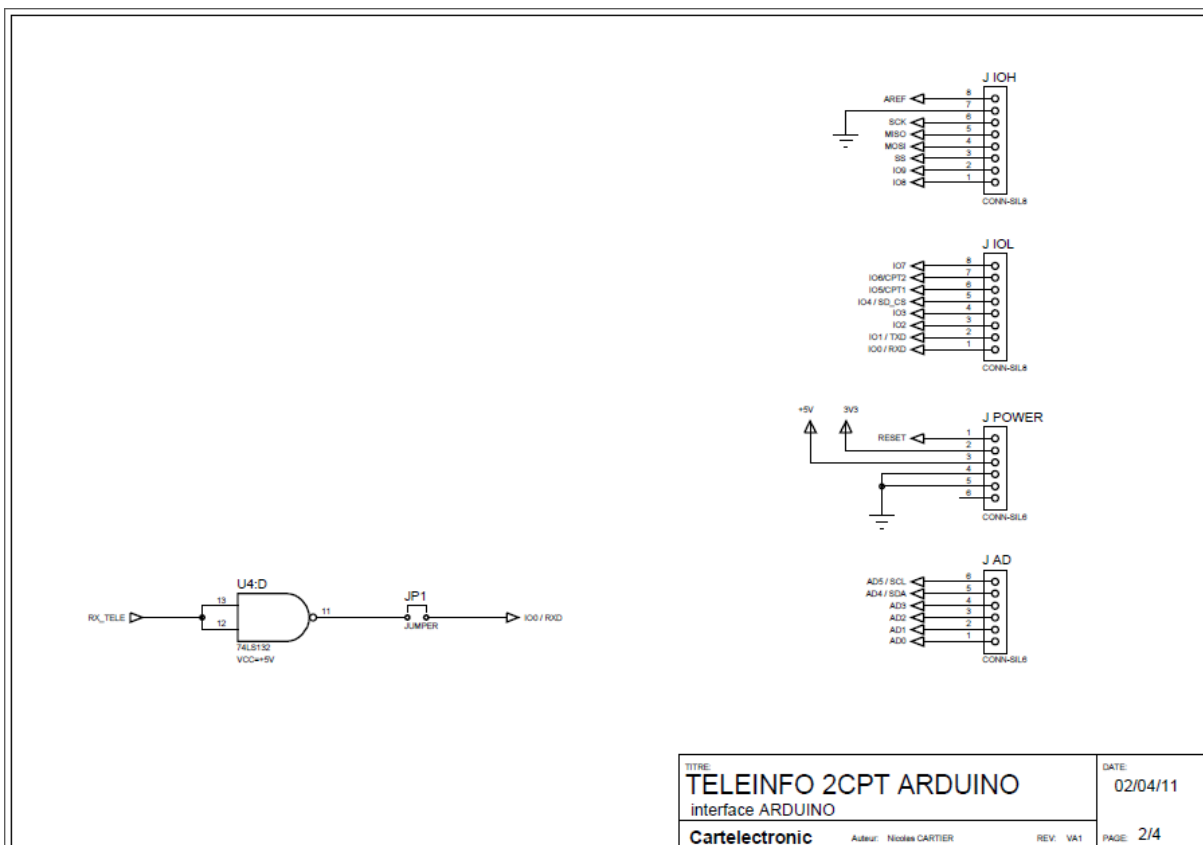
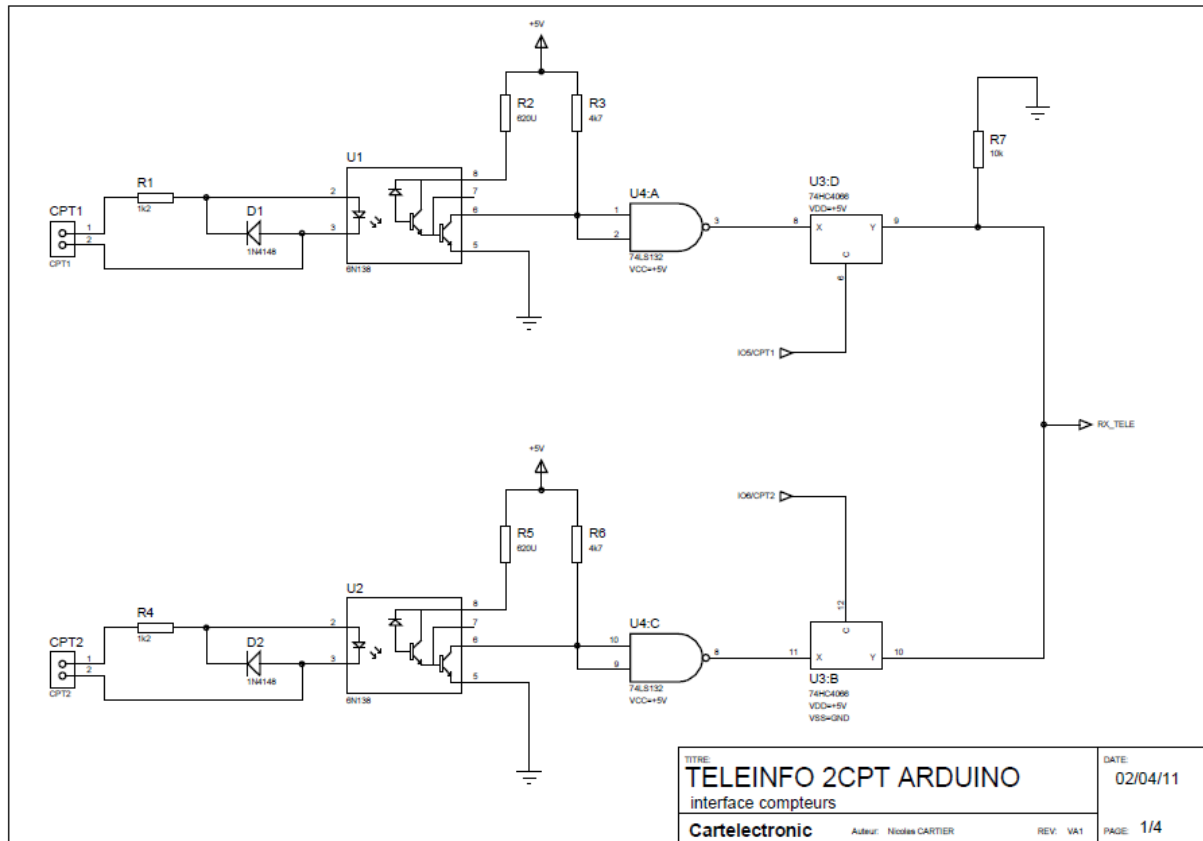
- 3 leds d'état (CPT1, Lecture, CPT2)

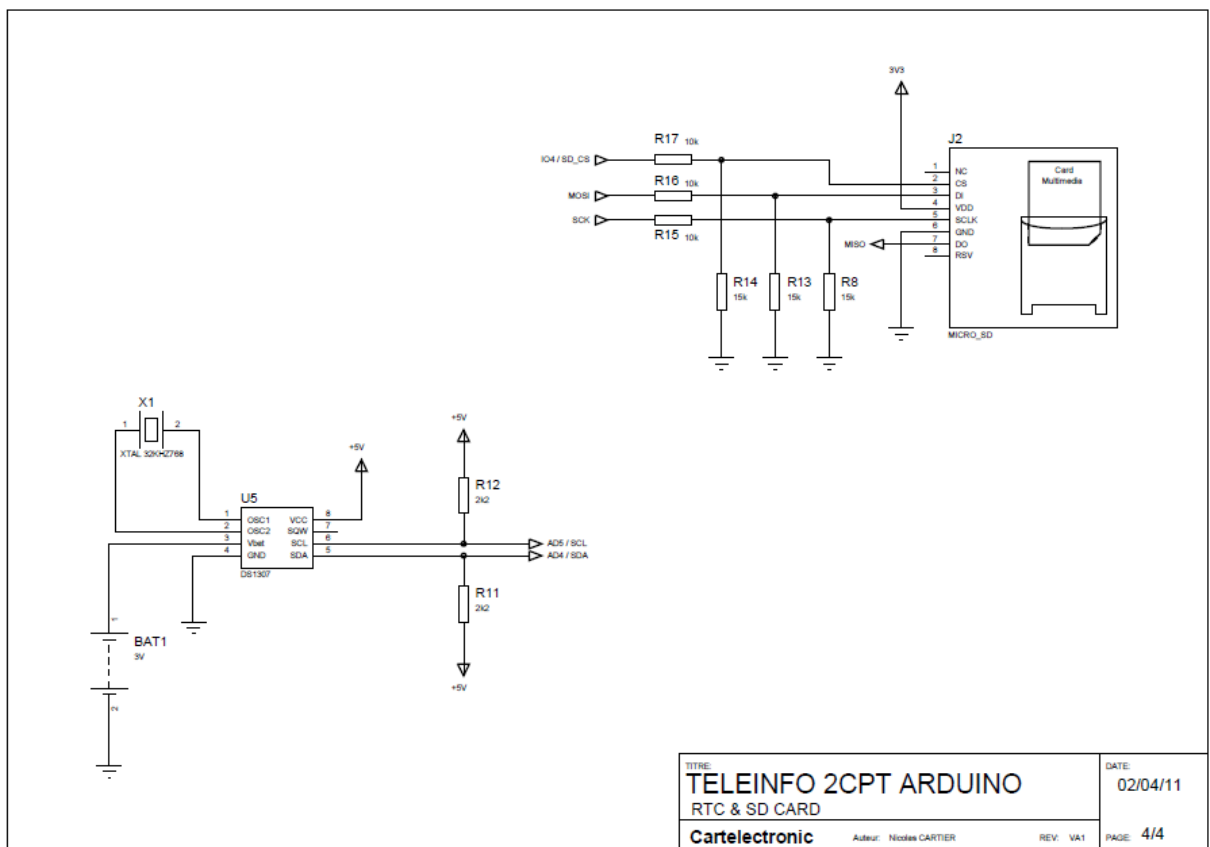
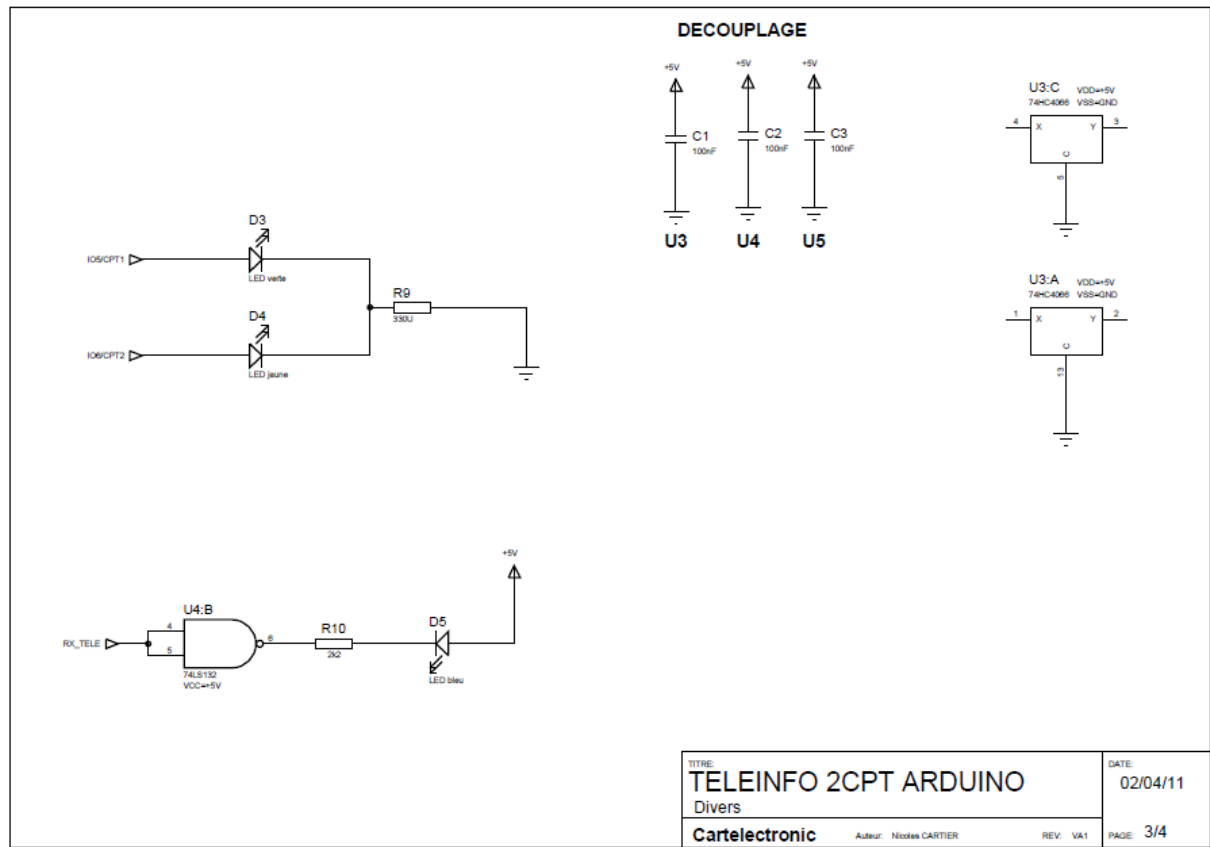
- Deux entrées compteurs optocouplées

- Lecteur de carte micro-SD

- Horloge temps réel RTC à base de chip DS1307

Voir aussi :





5. Arduino

<http://arduino.cc/fr/Main/DebuterIntroduction>

5.1.1 Qu'est-ce qu'Arduino?

Le système Arduino est un outil pour fabriquer de petits ordinateurs qui peuvent capter et contrôler davantage de choses du monde matériel que votre ordinateur de bureau. C'est une plateforme open-source d'électronique programmée qui est basée sur une simple carte à microcontrôleur (de la famille AVR), et un logiciel, véritable environnement de développement intégré, pour écrire, compiler et transférer le programme vers la carte à microcontrôleur.

Arduino peut être utilisé pour développer des objets interactifs, pouvant recevoir des entrées d'une grande variété d'interrupteurs ou de capteurs, et pouvant contrôler une grande variété de lumières, moteurs ou toutes autres sorties matérielles. Les projets Arduino peuvent être autonomes, ou bien ils peuvent communiquer avec des logiciels tournant sur votre ordinateur (tels que Flash, [Processing](#) ou MaxMSP). Les cartes électroniques peuvent être fabriquées manuellement ou bien être achetées pré-assemblées; le logiciel de développement open-source peut être téléchargé gratuitement.

Le langage de programmation Arduino est une implémentation de [Wiring](#), une plateforme de développement similaire, qui est basée sur l'environnement multimédia de programmation [Processing](#).

5.1.2 Pourquoi Arduino ?

Il y a de nombreux microcontrôleurs et de nombreuses plateformes basées sur des microcontrôleurs disponibles pour l'électronique programmée. Parallax Basic Stamp, Netmedia's BX-24, Phidgets, MIT's Handyboard, et beaucoup d'autres qui offrent des fonctionnalités comparables. Tous ces outils prennent en charge les détails compliqués de la programmation des microcontrôleurs et les intègrent dans une présentation facile à utiliser. De la même façon, le système Arduino simplifie la façon de travailler avec les microcontrôleurs, tout en offrant plusieurs avantages pour les enseignants, les étudiants et les amateurs intéressés par les autres systèmes :

- **Pas cher** : les cartes Arduino sont relativement peu coûteuses comparativement aux autres plateformes. La moins chère des versions du module Arduino peut être assemblée à la main, et même les cartes Arduino pré-assemblées coûtent moins de 25 Euros (microcontrôleur inclus...) !!!
- **Multi-plateforme** : Le logiciel Arduino, écrit en Java, tourne sous les systèmes d'exploitation Windows, Macintosh et Linux. La plupart des systèmes à microcontrôleurs sont limités à Windows.
- **Un environnement de programmation clair et simple**: L'environnement de programmation Arduino (= le logiciel Arduino) est facile à utiliser pour les débutants, tout en étant assez flexible pour que les utilisateurs avancés puisse en tirer profit également. Pour les enseignants, il est basé sur l'environnement de programmation [Processing](#) : les étudiants qui apprennent à programmer dans cet environnement seront déjà familiarisés avec l'aspect du logiciel Arduino.

- **Logiciel Open Source et extensible** : Le logiciel Arduino et le langage Arduino sont publiés sous licence open source, disponible pour être complété par des programmeurs expérimentés. Le langage peut être aussi étendu à l'aide de bibliothèques C++, et les personnes qui veulent comprendre les détails techniques peuvent reconstruire le passage du langage Arduino au langage C pour microcontrôleur AVR sur lequel il est basé. De la même façon, vous pouvez ajouter du code du langage AVR-C directement dans vos programmes Arduino si vous voulez.
- **Matériel Open source et extensible** : Les cartes Arduino sont basées sur les microcontrôleurs Atmel ATMEGA8, ATMEGA168, ATMEGA 328, etc... Les schémas des modules sont publiés sous une licence Creative Commons, et les concepteurs de circuits expérimentés peuvent réaliser leur propre version des cartes Arduino, en les complétant et en les améliorant. Même les utilisateurs relativement inexpérimentés peuvent fabriquer la version sur plaque d'essai de la carte Arduino, dans le but de comprendre comment elle fonctionne et pour économiser de l'argent.

6. Arduino GSM Shield

Le Gsm shield permet l'envoi des données récoltées par l'arduino le cas échéant la consommation d'énergie de compteur à relever.

La GPRS Shield permet d'ajouter à n'importe quelle carte arduino des capacités gsm. En "gros" vous pourrez faire des appels, recevoir des appels, gérer les fréquences DTMF ou utiliser le modem gprs pour connecter à internet votre arduino partout dans le monde (dans la mesure où votre carte sim puisse le faire bien sur 😊)



7. Conclusion

J'avoue que ce document a été fait en hâte, il présente sommairement la faisabilité mais ne traite pas l'aspect rentabilité toutefois l'abonné sera facturé sur la consommation réelle et non plus en estimation et les releveurs n'auront plus à se déplacer ce qui engendrera gain en hydrocarbure et amortissement véhicules.

Ce concept pourra être utilisé pour les compteurs Gaz et les compteurs eau. Bref en utilisant des cartes SIM le secteur Télécom et Orange sera bénéficiaire vu le nombre d'abonnés ici en Tunisie seulement 3 millions d'abonnés triplé (compteur Gaz et compteur eau).

Une meilleure solution plus économique encore serait de créer une Application (Android/Iphone etc) à travers duquel l'utilisateur ici (releveurs ou abonné) flash l'index du compteur avec la CAM intégré à son Smartphone ensuite il envoie l'image au serveur du prestataire de service en énergie ou/et eau pour être traitée par OCR et les données clients seront enregistrées dans la Base de données facturation du prestataire du coup le client ne sera facturé que ce qu'il a consommé.