| Systèmes d'Information et Numérique | TSTI2D |
|---|---|
| Comment est structuré le produit ? | SÉANCE 2 |
| **Surveillance de la tension batterie** | Activité 2 |

**Durée : 2 H 00**

**Objectif visé :** **O3 - Analyser l'organisation fonctionnelle et structurelle d'un produit**

**O6 – Préparer une simulation et exploiter les résultats pour prédire un fonctionnement,**

**Compétences :** **CO 3.3 CO 6.2 CO 6.4**

**Connaissance visée :** **SA 2.4.3. Codage et traitement de l'information**
**SA 2.4.2 Acquisition et restitution de l'information**
**SA 3.4.1. Nature et représentation de l'information**

**Matériel nécessaire :** Poste informatique équipé de Proteus ISIS et Arduino

**Objectifs de l'activité :** L'objectif de cette activité est de dimensionner les éléments permettant la surveillance du niveau de charge de la batterie. À partir du cahier des charges et de l'étude L'élève doit être capable :

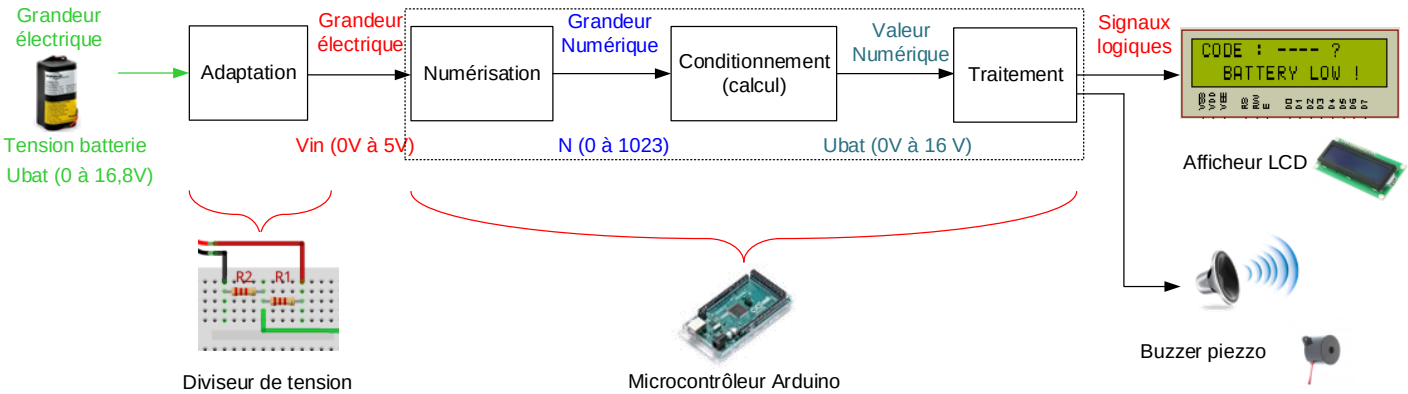
- De dimensionner les composants nécessaires à l'acquisition et conversion en grandeur numérique d'une information électrique (principe de l'adaptation, démarches et méthodes, notions requises),
- De conditionner la grandeur numérique obtenue en valeur numérique image de la grandeur convertie,
- De programmer la chaine complète de conversion en s'appuyant sur un algorithme donné, et de valider la solution retenue par simulation.

## ♦PARTIE 1 : PRÉSENTATION DU PROBLEME

La serrure codée est alimentée par une batterie Li-Ion qui délivre une tension nominale de **14,8V** pour une capacité de **2600** mAh.

Dans l'étude fonctionnelle, nous avions vu que la **tension critique** (avant décharge profonde) correspondait à **10V**. Il vous faut donc mettre en place un système de **surveillance de la tension** de la batterie, et **alerter** l'utilisateur si la tension de seuil critique est atteinte pour remplacement ou rechargement de celle-ci.

La carte Arduino possède un convertisseur A/N de **10 bits**, pouvant mesurer des tensions variant de **0V** à **+5V**. La chaine complète d'information à réaliser et à programmer est la suivante :



Comme la tension de la batterie est **supérieure à 5V**, nous ne pouvons pas mesurer directement celle-ci par programmation. Nous utiliserons pour cela un diviseur de tension afin d'adapter celle-ci et rendre ainsi cette grandeur exploitable par le système microprogrammé.

## ♦PARTIE 2 : ANALYSE DE LA SOLUTION

### 2.1 : Étude de l'adaptation en tension de la batterie :

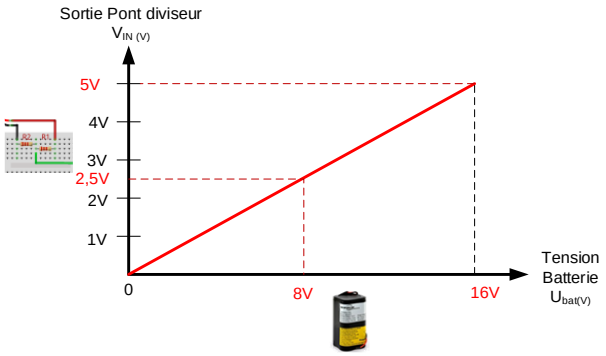
**Q1** : **Donnez** l'équation de **Vin** en fonction de **Ubat** à partir de la fonction de transfert suivante :

C'est une fonction linéaire de la forme y = a.x

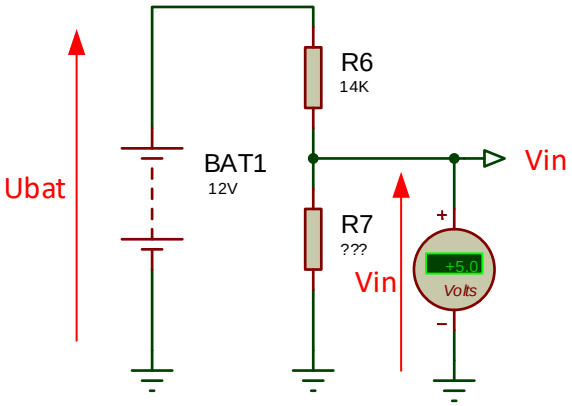
Vin = a . Ubat

5V = a . 16V  donc a = 5/16

L'équation est donc  **Vin = 5 . Ubat / 16**




**Q2** : **Exprimez** la relation de **Vin** en fonction de **Ubat**, **R6** et **R7**. **Calculez** alors la valeur de **R7** si l'on souhaite obtenir **Vin** = +5V lorsque **Ubat** = +16V avec **R6** = 22kΩ :




**Vin** = **Ubat** . R7 / (R6 + R7)

5V = 16V . **R7**/(22k + **R7**)

5.(22k + **R7**) = 16.**R7**

110k + 5.**R7** = 16.**R7** donc  11.**R7** = 110k

**R7** = 110k / 11 = **10 kΩ**

**Q3** : **Complétez** votre schéma de simulation sous Proteus ISIS en **y intégrant le diviseur de tension**, puis par simulation **relevez** dans le tableau suivant les différentes valeurs de **Vin** en fonction de la tension batterie **Ubat** :

| **Ubat (V)** | **0** (vide) | **8** | **10** (seuil critique) | **12** | **14,8** (tension nominale) | **16,8** (pleine charge) |
|---|---|---|---|---|---|---|
| Vin (V) | 0 | 2,5 V | 3,13 V | 3,75 V | 4,63 V | 5 V* |

** ne peut dépasser VFS = 5V !*

**Q4** : Ce montage permet-il d'adapter correctement la tension de la batterie pour une conversion analogique/numérique ? (**Détaillez** votre réponse) :

Oui, car le signal à mesurer (Vin) est bien compris entre 0 et 5V, même lorsque la batterie est complètement rechargée (Vbat > 14,8 V).

De plus, la valeur à mesurer pour le seuil critique de la batterie (Ubat = 10V) est assez significative (Vin = 3,13 V) pour être mesurée avec précision.

On pourrait améliorer le diviseur de tension pour prendre en compte les tensions supérieures à 16V (pleine charge) en augmentant la valeur de R7 (27 kΩ par exemple) mais réduirait légèrement la précision de mesure sur la plage de 10V.

## 2.2 : Étude de la conversion analogique/numérique et du conditionnement de Ubat :

**Q5** : Donnez la relation de la grandeur numérique de conversion **N** par rapport au quantum **q** et à la tension à convertir **Vin**. Développez sous forme de fraction la valeur de **q** :

N = Vin / q  avec q = 5/1023

**Q6** : Donnez alors la relation de la grandeur numérique de conversion **N** par rapport à **Vin** :

N = Vin / (5 / 1023)

Soit : N = 1023. Vin / 5

**Q7** : En déduire la relation de **Ubat** en fonction de **N** (cf Q1) :

**N** = 1023.**Vin** / 5 (1)  et  **Vin** = 5.**Ubat** / 16 (2)

On remplace **Vin** dans la première équation :

**N** = 1023 . (**5**.**Ubat** /16 ) /**5**  soit **N** = 1024.(**Ubat** / 16)

Ce qui donne : 16.**N** = 1023.**Ubat**  donc : **Ubat** = 16.**N** / 1023

**Q8** : Vérifiez vos relations en complétant le tableau ci-dessous avec au maximum 2 chiffres significatifs pour les tensions (***Remarque*** *: Vous pouvez Utiliser Excel pour vous faciliter les calculs !) :*

| **Ubat** | **0 V** (vide) | **8 V** | **10 V** (seuil critique) | **12 V** | **14,8 V** | **16,8 V** (pleine charge) |
|---|---|---|---|---|---|---|
| **Vin** | 0 V | 2,5 V | 3,13 V | 3,75 V | 4,63 V | 5 V |
| **N** | 0 | 511 | 639 | 767 | 946 | 1023 |
| **Ubat** calculé | 0 | 7,99 | 9,99 | 12 | 14,8 | 16 |

**Q8** : Pourquoi utilise-t-on dans les formules **1023** ($2^{nbit}$ -1) et non pas **1024** ($2^{nbit}$) ? :

Parce que la tension à convertir Vin **peut atteindre la tension pleine échelle VFS = 5V**. La batterie fournissant au maximum de sa charge 16,8V, soit Vin > 5V, nous devons donc utiliser 1024 au lieu de 1023 pour les calculs.

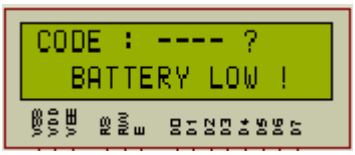
## ♦PARTIE 3 : PROGRAMMATION ET VALIDATION DE LA SOLUTION

### ♦ Cahier des charges et EXIGENCES du PROJET :

La **tension de seuil critique** sera fixée à **10,5 V** afin de se donner une marge de sécurité (puisqu'à 10V nous entrons déjà en décharge profonde…).

La **surveillance** de la tension batterie ne doit pas **empêcher le fonctionnement** normal de la **gestion de l'accès**.

Ce qui implique que la mesure et le test de la tension batterie doivent se faire **par alternance** (toutes les **5 secondes** par exemple) pour éviter d'empêcher le programme principal de fonctionner correctement.



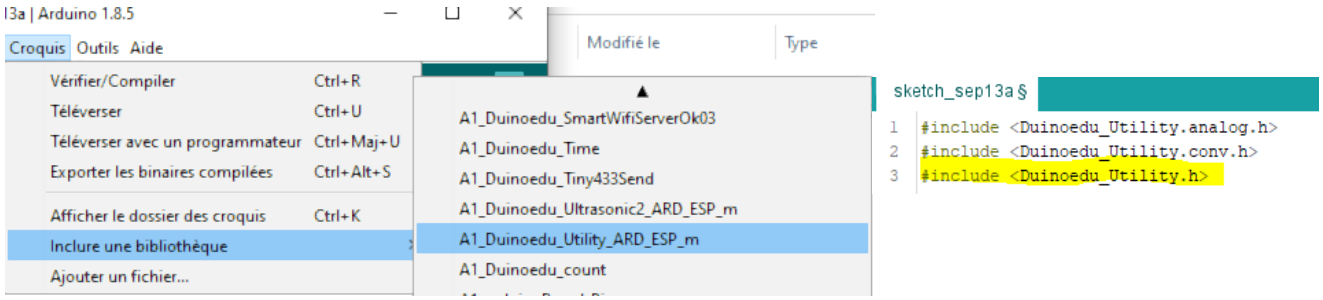

Pour exécuter un code tous les ***n*** milliseconde, nous utiliserons la librairie "***Duinoedu_Utility.h***" qui permettra d'intégrer la fonction **TIMER** suivante :

```
ONLY_EVERY(<temps en ms>,_ABVAR_1_Repeat)
  Fonction ou Code à exécuter…
END_ONLY_EVERY
```

Exemple :

```
ONLY_EVERY(2500,_ABVAR_1_Repeat)
  tone(440,100)   //génère un son toutes les 2,5 sec
END_ONLY_EVERY
```

Pour utiliser cette fonction, vous devrez intégrer la libraire "Duinoedu_Utility.h". Cette libraire fait partie des librairies d'**Arduino augmenté**, et peut être ajoutée à l'environnement de travail en cliquant sur "***Croquis***" -> "***Inclure une bibliothèque***" -> "***A1_Duinoedu_ARP_ESP_m***" :




Ou alors plus simplement en ajoutant au début du programme la ligne suivante :

```
1  #include <Duinoedu_Utility.h>
```

### ♦ Algorithme de Programmation :

Pour ajouter la surveillance de la batterie à votre programme, il vous faudra **ajouter** une fonction que l'on appellera "**`Surveillance_Batterie( )`**". Cette fonction devra être appelée toutes les 5 sec grâce à la fonction TIMER.

Algorithme en pseudo-code de la fonction **Surveillance_Batterie** :

```
N est un entier = 0
Vin est un réel = 0.0
Ubat est un réel = 0.0

Procédure Surveillance_Batterie ()
    N ← valeur de conversion A/N de la broche A0
    Vin ← calcul de Vin en fonction de N
    Ubat ← calcul de Ubat en fonction de Vin
    Débogage des informations : valeurs de N, Vin et Ubat
    SI Ubat < 10,5 ALORS
        Afficher sur la 2ème ligne de l'afficheur "BATTERY LOW !"
        Générer SonErreur
    FIN
FIN
```

La fonction C++ en arduino permettant la conversion A/N est :

```
analogRead(<broche analogique>)
```

Il vous faudra également lors de la **construction du code** (fonction LOOP) ajouter l'affichage sur la deuxième ligne "-**ACCES INTERDIT-**" en même temps que le code :

```
lcd.setCursor(0, 1);
lcd.print("-ACCES INTERDIT-") ;
```

…Sinon seul le message "**BATTERY LOW !**" sera affiché !

CORRIGÉ :

```
119   ONLY_EVERY(50000, _ABVAR_1_Repeat)
120     Serial.println("_ABVAR_1_Repeat = " + String(_ABVAR_1_Repeat));
121     SurveillanceBatterie();
122   END_ONLY_EVERY
123 }
124
125
126 void SurveillanceBatterie()
127 {
128   N = analogRead(A0);
129   Vin = 5.0 * N / 1024.0;
130   Ubat = 16 * Vin / 5;
131   Serial.println("N = " + String(N));
132   Serial.println("Vin = " + String(Vin) + " V");
133   Serial.println("Ubat = " + String(Ubat) + " V");
134   if (Ubat < 10.5) {
135     lcd.setCursor(0, 1);
136     lcd.print("  BATTERY LOW ! ");
137     SOUND(27, 6);
138   }
139 }
```