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 :

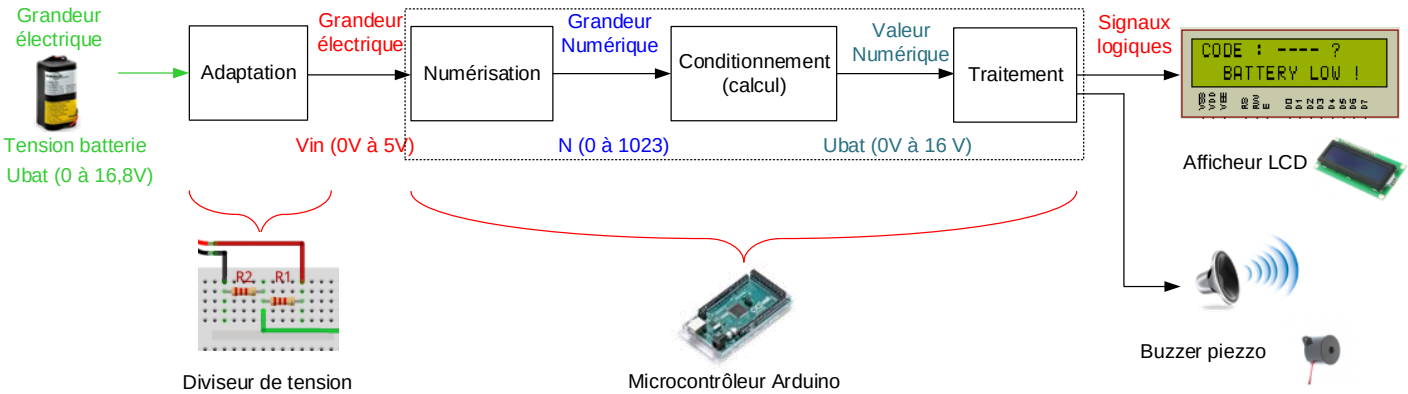
- De dimensionner les composants nécessaires à l'acquisition et conversion en grandeur numérique d'une information électrique (principe de l'adaptation, démarches et méthodes, notions requises),
- De conditionner la grandeur numérique obtenue en valeur numérique image de la grandeur convertie,
- De programmer 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 :

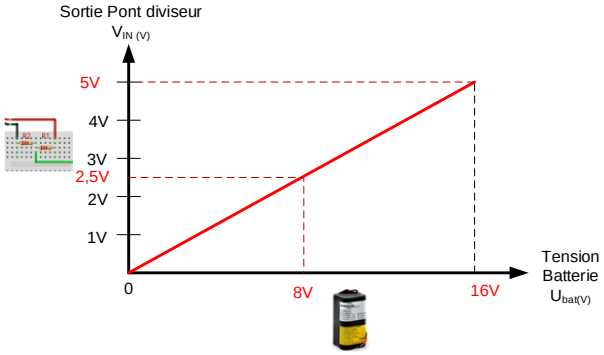
.....

.....

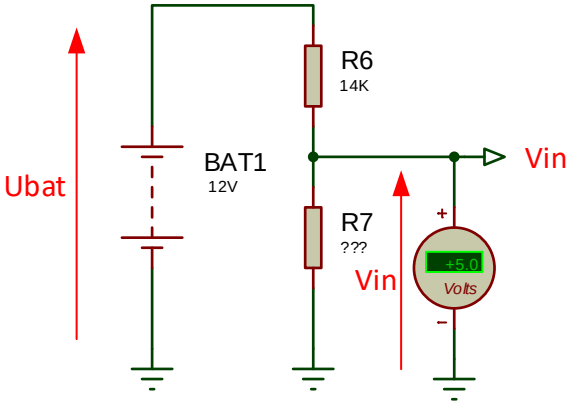
.....

.....

.....



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 :



.....

.....

.....

.....

.....

.....

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)						

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

.....

.....

.....

.....

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** :

……

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

……

……

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

……

……

……

……

……

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						
N						
Ubat $_{calculé}$						

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

……

……

……

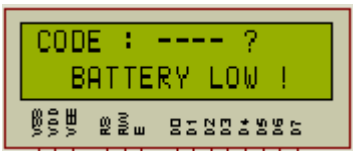
♦PARTIE 3 : PROGRAMMATION ET VALIDATION DE LA SOLUTION

♦ 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.



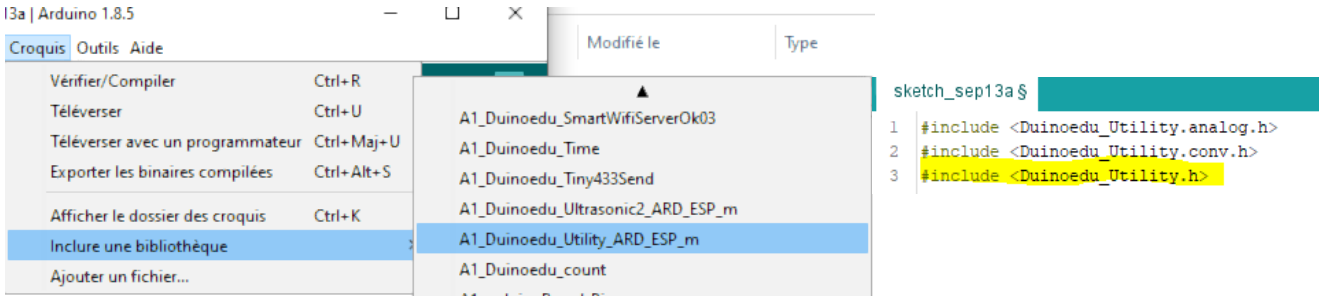
Pour exécuter un code tous les ***n*** milliseconde, nous utiliserons la librairie "***Duinoedu_Utility.h***" qui permettra d'intégrer la fonction **TIMER** suivante :

```
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 ALORS
        Générer SonErreur
        Afficher sur la 2ème ligne de l'afficheur "BATTERY LOW !"
    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 !