**WSN : Microsystèmes Hétérogènes**

**Document technique**

Nous avions pour objectifs de réaliser un programme, dans le langage de notre choix, simulant des cycles de fonctionnement d’un microsystème autonome permettant de faire des mesures à l’aide de différents capteurs. Ce microsystème est composé d’un microprocesseur, d’un émetteur/récepteur, d’au moins un capteur et d’une batterie. Pour ce faire on, a écrit un programme en Python avec une interface graphique.

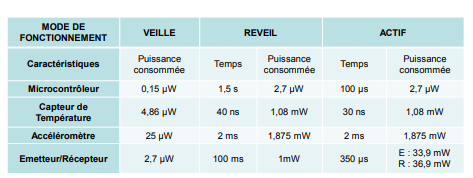
**A screenshot of a cell phone

Description automatically generated**

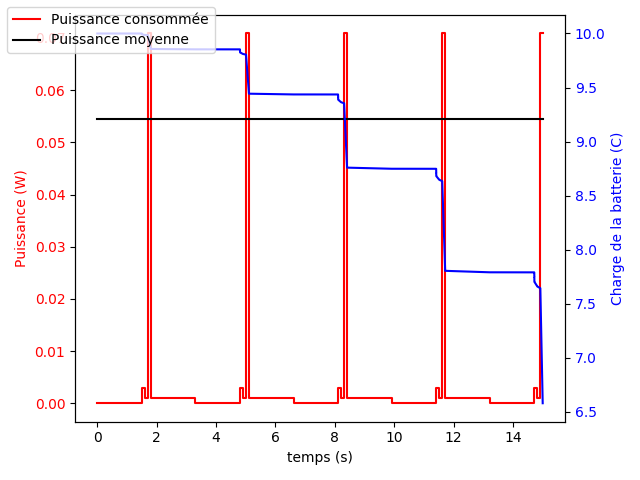
Sur cet interface graphique, on trouve des champs vides, correspondant aux valeurs de puissances et de temps pour le microprocesseur et l’émetteur/récepteur, et la charge initiale et puissance de recharge de la batterie, et de plus deux boutons dont le bouton « lancer simulation» pour lancer la simulation d’un période que l’utilisateur peut choisir (champs « temps de simulation ») , et un bouton « ajouter capteur » qui fait apparaitre les champs de valeurs de puissance et de temps pour un nouveau capteur

Ainsi pour pouvoir lancer la simulation il y’a :

Par défaut : le microsystème avec pour composants le microprocesseur, l’émetteur/récepteur, une batterie et deux capteurs et avec les valeurs de puissances et de temps par défaut contenues dans le tableau suivant :

Tableau 1 :

La batterie dans cette simulation a une puissance de recharge de 1W et une charge initiale de 10C. Ainsi, en appuyant sur « lancer simulation » dans l’interface graphique sans rajouter de valeurs, on obtient la simulation de la puissance consommée, la puissance moyenne et la charge de la batterie sur 15 secondes.



En insérant des valeurs dans l’interface graphique : Il faut remplir en intégralité toutes les valeurs de tous les champs (microprocesseur, émetteur/récepteur, batterie et au moins un capteur) et leur unité de mesure pour écraser les valeurs par défaut.

**Annexe technique**

1. **Interface graphique**

Cette interface graphique a été écrite avec la bibliothèque Tkinter. C’est la bibliothèque la plus générique pour l’utilisation de interfaces graphiques en Python, et elle nous permet de créer des boutons, des champs de texte et des listes déroulantes. **A screenshot of a cell phone

Description automatically generated**

L’utilisateur peut remplir les valeurs souhaitées dans cette interface, en choisissant les bonnes unités de mesure. Un exemple d’utilisation :

**A screenshot of a cell phone

Description automatically generated**

1. **Courbe de la simmulation**

Avec les valeurs de l’image antérieure, on peut voir le graphe de puissances suivant :

A screenshot of a cell phone

Description automatically generated

Le graphe montre l’évolution de la charge de la batterie dans le temps, et la puissance consommée du système.

1. **Simulation des composantes et de la batterie**

Selon ce qu’on a vu en classe, la simulation a sept étapes :

* 1. Le réveil du microcontrôleur
  2. Le réveil des capteurs
  3. Les mesures
  4. Le traitement et mise en veille des capteurs
  5. Le réveil de l’émetteur/récepteur
  6. L’envoi des données
  7. La mise en veille

Avant de réaliser chaque cycle, on vérifie si la batterie a de l’énergie suffisant pour réaliser le cycle. Si non, on prend du temps pour recharger la batterie avant de lancer le cycle. Par exemple, si la batterie est initialisée vide, avec la puissance de recharge de 0.1W et les valeurs des composants du tableau 1.

A screenshot of a cell phone

Description automatically generated

Avant que chaque cycle commence, la batterie est chargée jusqu’à la charge nécessaire pour le cycle.

1. **Limites et avantages du langage :**

Le langage Python a pour avantage la facilité de prise en main pour une programmation rapide, grâce à toutes les bibliothèques fournies (Numpy, Matplotlib, Tkinter) qui facilitent l’utilisation de listes, les calculs, les produits matriciels et le tracé de graphiques.

Cependant développer un programme avec une bonne interface graphique en Python reste assez difficile et avec un rendu basique. De plus l’exécution du programme nécessite la présence de l’environnement python sr la machine d’exécution. L’utilisateur doit aussi avoir toutes les bibliothèques installées dans son ordinateur et ça rends le développement et l’utilisation un peu plus difficiles.