Institut Supérieur des Études Technologiques de Kairouan

Département : Génie Électrique

” Système temps réel ”

TP2 :Ordonnancement RM et EDF et implémentation sur carte Arduino

Introduction :

Lors de cette séance nous allons :

- Le comportement de différents algorithmes d'ordonnancement à l'aide du simulateur **Cheddar.**
- Implémenter le système temps réel sur la carte Arduino UNO

Ce simulateur nous permettra de comparer les performances des algorithmes d'ordonnancement et d'évaluer l'impact des différents paramétrés de simulation.

1.Présentation de de l'outil de simulation Cheddar :

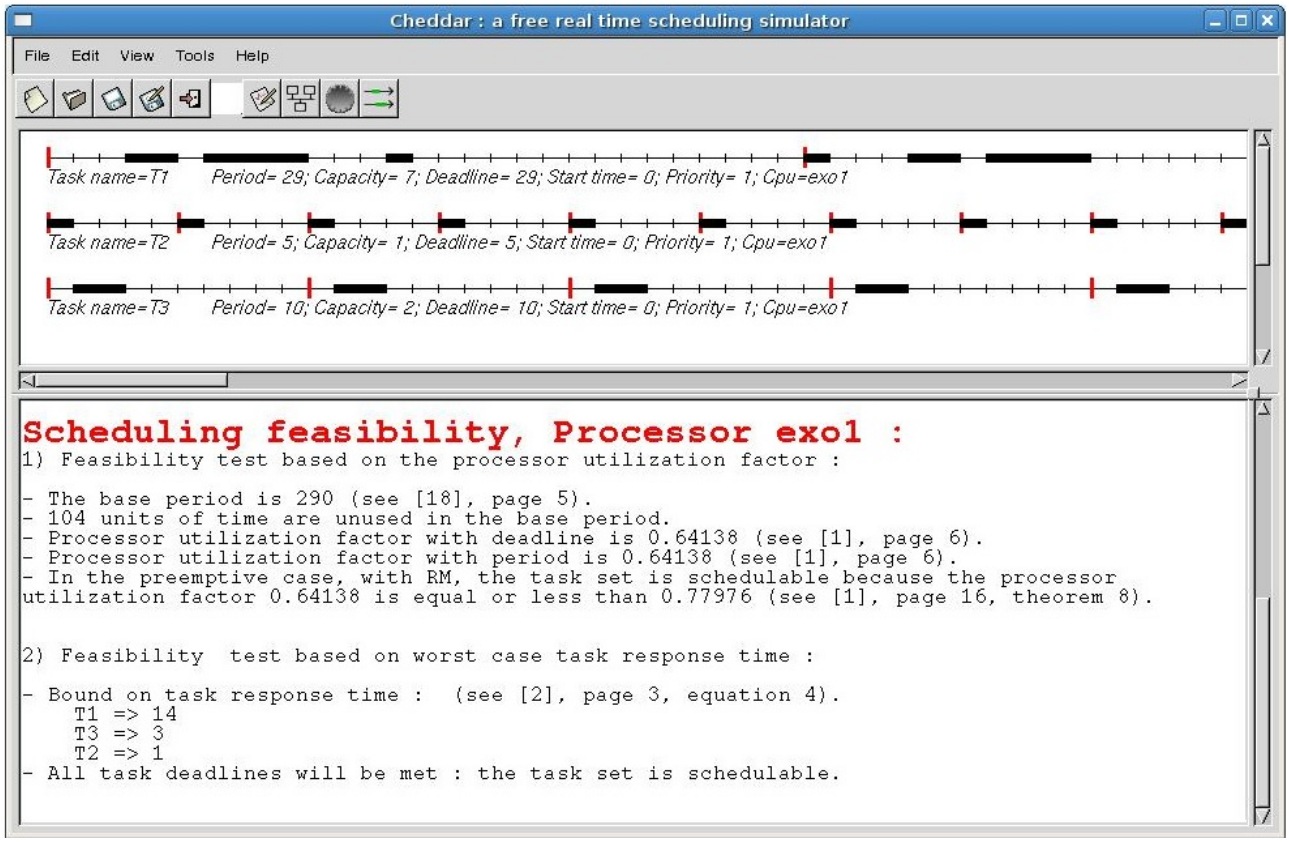
Cheddar est un outil de simulation permettant de calculer différents critères de performance (contraintes temporelles, dimensionnement de ressources). L'outil permet, entre autre, de tester le respect des contraintes temporelles d'un jeu de tâches modélisant une application/un système temps réel. Les principales fonctionnalités de l'outil sont :

- Calcul d'ordonnancement pour la plupart des algorithmes "standards" :
  - Rate Monotonic (ou RM)
  - Earliest Deadline First (ou EDF)
  - Deadline Monotonic (ou DM, Inverse Deadline).
  - Least Laxity First (ou LLF).

  A partir d'un ordonnancement, l'environnement de simulation permet d'obtenir :
- Les temps de blocage sur ressources partagées (bornes maximales, minimales, délais moyens)
- Les temps de réponse des tâches (bornes maximales, minimales, délais moyens).
- Recherche d'interblocage, d'inversion de priorité.
- Echéances manquées.

2.Prise en main de l'outil de simulation Cheddar :

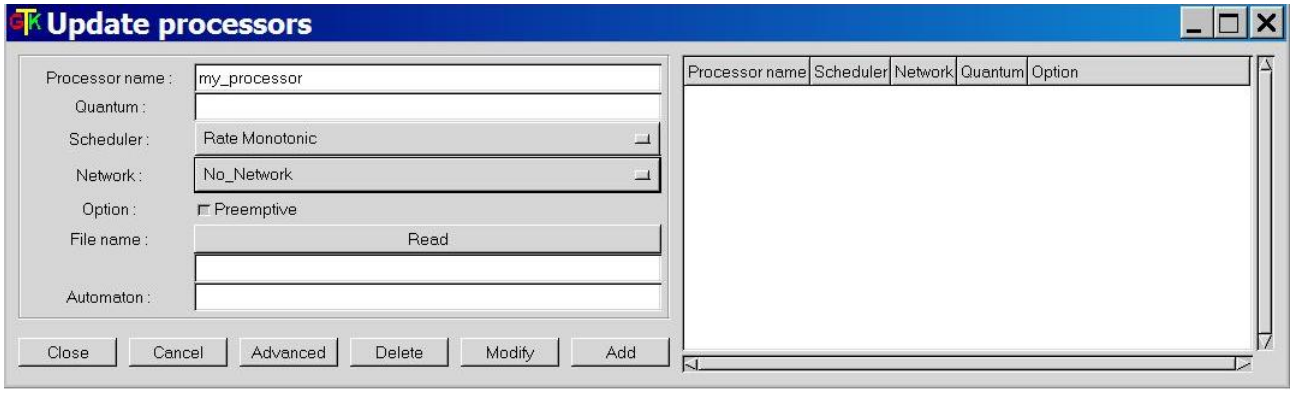
L'image ci-dessous est une copie d'écran de l'outil. On y voit dans la partie haute les chronogrammes issues de l'ordonnancement des tâches, puis, dans la partie basse, les résultats concernant l'analyse de l'ordonnancement ainsi que les tests de faisabilité.



Première étape :Définir le (ou les) processeur :

Cette section vous montre comment appeler les fonctions de Cheddar. Cheddar fournit des outils pour vérifier les contraintes temporelles des tâches en temps réel. Ces outils sont basés sur les résultats classiques de la théorie de l'ordonnancement en temps réel. Avant d'appeler ces outils, vous **devez définir un système qui est principalement composée de plusieurs processeurs et des tâches**.
Pour définir un processeur, choisissez l'option "processeurs "Edit / Update"

La fenêtre ci-dessous apparaît alors :



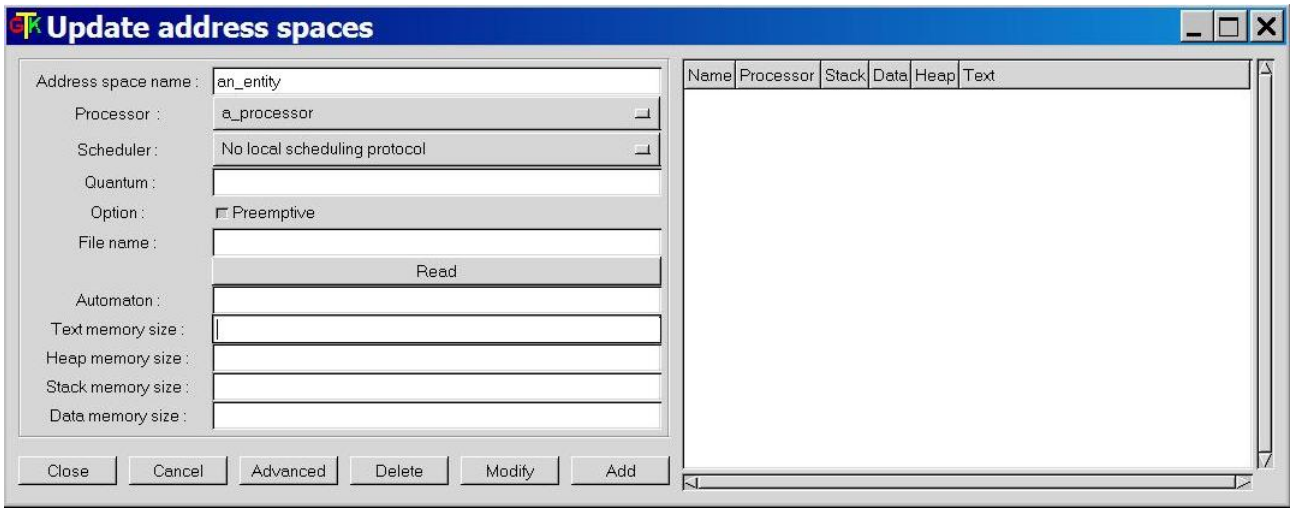
Un processeur est défini par les champs suivants (voir Figure 1.1) :

- **Processor name :** Le nom du processeur. Un nom de processeur peut être n'importe quelle combinaison de caractères littéraux y compris de soulignement. L'espace est interdite. Chaque processeur doit avoir un nom unique.
- **scheduler** : L'ordonnanceur hébergé par le processeur. Fondamentalement, vous pouvez choisir parmi une série de différents ordonnanceurs (voir cours)
- **quantum :** La valeur quantum associée à l'ordonnanceur. Cette information est utile si un programmateur doit gérer plusieurs tâches avec la même priorité statique ou dynamique. Le quantum est une borne sur le retard d'une tâche (si le montant est égal à zéro, il n'y a pas de borne sur le temps de maintien du processeur)
- **file name :** Le nom du fichier, c'est le nom d'un fichier qui contient le code source d'un ordonnanceur défini par l'utilisateur.

Attention : avec Cheddar, pour ajouter un processeur (ou un objet), vous devez appuyer sur le bouton Add avant d'appuyer sur le bouton close. Cela vous permet de définir plusieurs objets rapidement sans la fermeture de la fenêtre (vous devez ensuite appuyer sur Add pour chaque objet défini).

Deuxième étape :Définir un espace d'adressage :

La prochaine étape pour exécuter une simulation, consiste à définir un espace d'adressage. Choisissez le sous-menu "Edit/Update address spaces". Une adresse modèles de l'espace un morceau de mémoire qui contiennent des tâches, des tampons ou des ressources partagées. La figure 1.2 montre le widget utilisé pour définir une telle fonctionnalité. Au moment où nous parlons, les informations que vous devez fournir est :

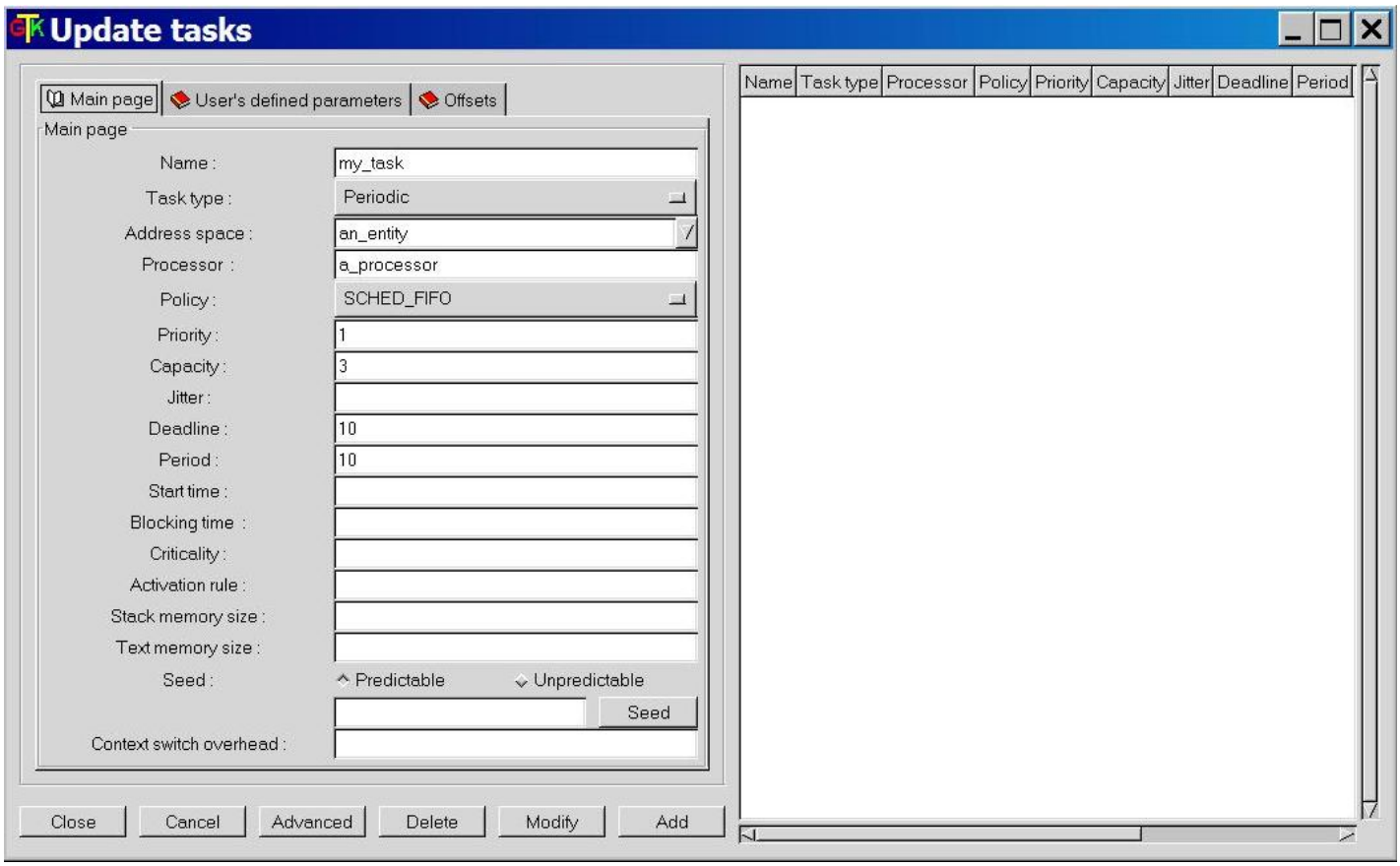


- **Name :** Une adresse nom de l'espace peut être n'importe quelle combinaison de caractères littérales, y compris de soulignement. L'espace est interdite. Chaque nom de l'espace d'adresse doit être unique.
- **processor name :** C'est le processeur qui héberge l'espace d'adressage.

Troisième étape :Définir les taches $\tau()$:

Choisissez "Edit/Update tasks" sous-menu. La fenêtre de la figure 1.3 est alors affiché. Cette fenêtre est composée de 3 sous-fenêtres : la «main page», le «offset page» et la user's defined parameters page". La page principale «main page» contient les informations suivantes :

- **Name :** le nom de la tâche doit être unique , une capacité (lié à son temps d'exécution) et un endroit pour le faire fonctionner (un nom de processeur et une adresse nom de l'espace) . Les autres paramètres sont facultatifs mais peuvent être nécessaires pour un ordonnanceur particulier.
- **type of task** : Un type de tâche . Il décrit la façon dont la tâche est activée. Une tâche apériodique est activé uniquement une fois . Une tâche périodique est activé plusieurs fois et le délai entre deux activations est fixe .
- **The period** La période . C'est le temps entre deux activations de la tâche.
- **start time :** A l'heure de début . C'est le moment où la tâche arrive dans le système (son premier temps d'activation) .
- **deadline :** La tâche doit terminer son activation avant son échéance.



Remarque :
La deuxième et la troisième page des informations de la tâche sont rarement utilisés par les utilisateurs.

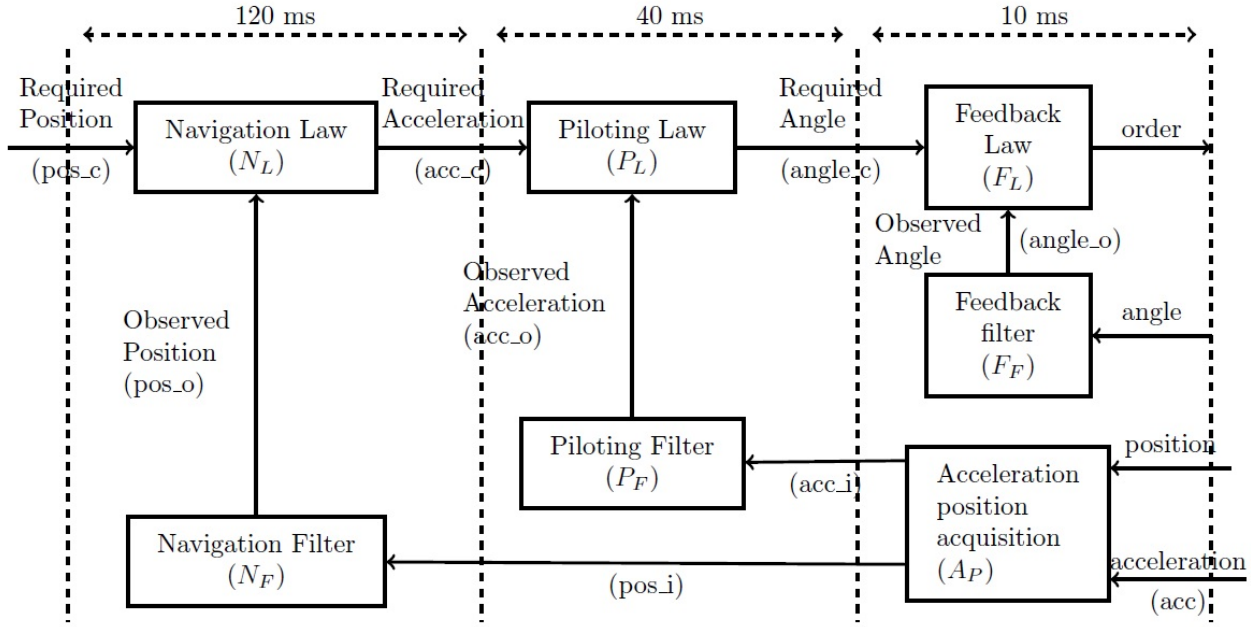
Attention :
Lorsque vous créez des tâches, dans la plupart des cas, Cheddar ne vérifie pas

si vos paramètres de tâche sont mal definies selon l'ordonnaceur que vous avez sélectionné précédemment.
ces contrôles sont effectués à l'analyse des tâches / ordonnancement. Bien sûr, vous pouvez toujours changer tâche et les paramètres du processeur avec "Edit/Update" et "Edit/Duplicate".

3.Mise en situation :

Nous considérons un système de contrôle de vol simplifié sur la figure ci-dessous. Ce système contrôle :

- l'attitude
- la trajectoire
- la vitesse d'un avion



Il se compose de 7 tâches qui s'exécutent à plusieurs reprises à un rythme périodique.

- Le sous-système est le plus rapide et s'exécute à $10ms$, il acquiert l'état du système(angles, la position, l'accélération) et calcule la loi de retour du système.
  La commande est alors envoyée aux de commande de vol.
- Le sous-système intermédiaire est la boucle de pilotage, il s'exécute à $40ms$ et détermine l'accélération à appliquer. Le sous-système le plus lent est la boucle de navigation, il s'exécute à $120ms$ et détermine la position à atteindre.
- La position souhaitée de l'avion est acquis via le pilote.

4.Analyse d'ordonnancement mono-processeur :

Lancer l'outil de simulation Cheddar depuis votre bureau : C :\$Users$\Desktop\ Cheddar-2.1-win32-bin\ cheddar.exe. Soit :

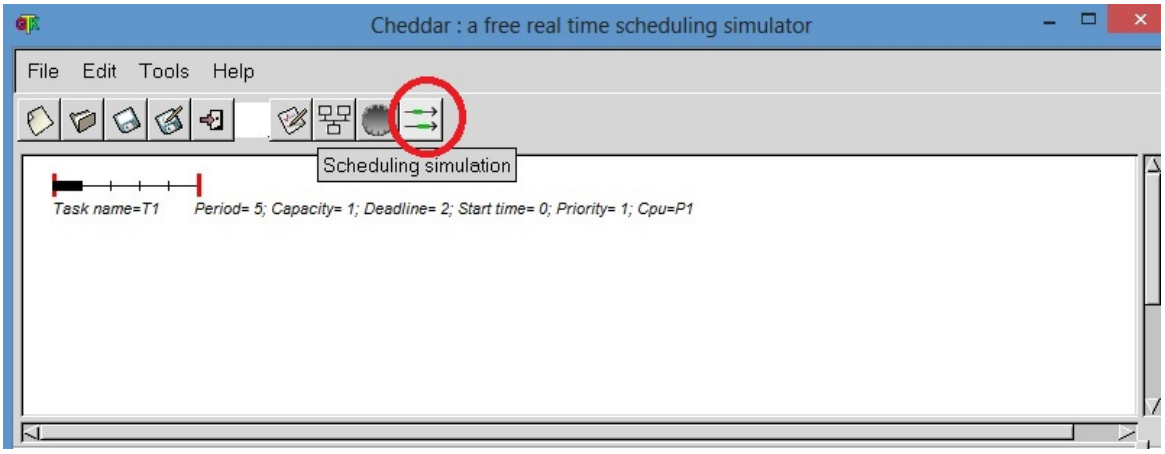
– T :Période d'exécution
– C :Durée d'exécution maximale
– r :Date de réveil
– D :Délai critique

On veut simuler une configuration donné de notre système de contrôle de vol. On suppose qu'on va associé a notre système un seul processeur et aux taches $(N_L, N_F, P_L, P_F, F_L, F_F, A_P)$ la configuration suivante :

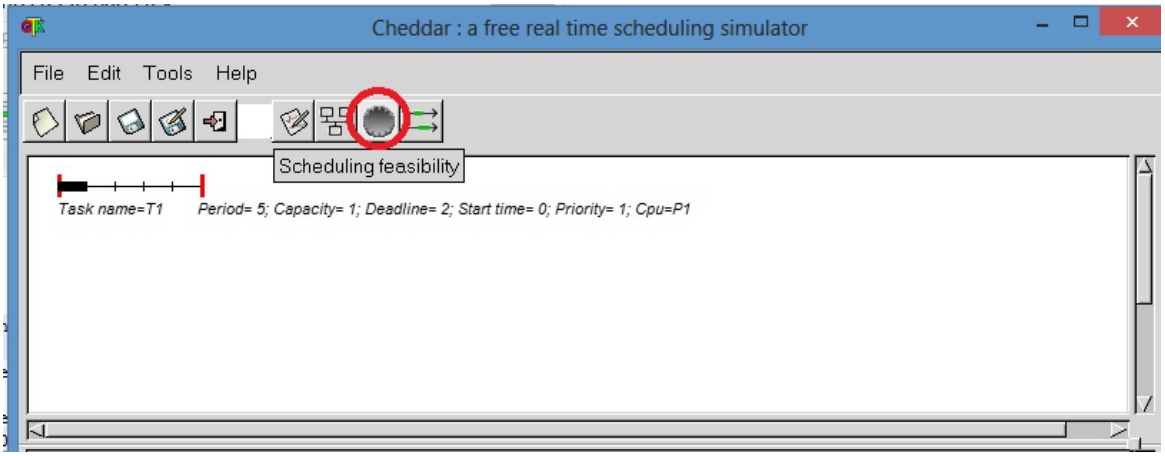
$Taches$	$T(ms)$	$C(ms)$	$r(ms)$	$D(ms)$	$Processeur$
N_L	120	20	0	120	1
N_F	120	10	0	120	1
P_L	40	5	0	40	1
P_F	40	5	0	40	1
F_L	10	2	0	10	1
F_F	10	1	0	10	1
A_P	10	1	0	10	1

On veut comparer entre l'ordonnancement RM et EDF pour choisir la meilleur solution pour notre système.

1. Rappeler le principe de chacun de ces ordonnancement(RM et EDF).
2. Faire une simulation d'ordonnancement avec RM. Pour cela cliquez sur *scheduling simulation* et Le résultat apparaît.Figure ci dessous.
3. Faire une simulation d'ordonnancement avec EDF.pour cela cliquez sur *scheduling simulation* et Le résultat apparaît.Figure ci dessous.



4. Pour chacun d'eux, faire le tests de faisabilité. Cliquez sur *scheduling feasibility*. Ce bouton nous permet de :
   – vérifiez si l'ordonnancement est faisable ou non
   – calculer le facteur d'utilisation du processeur et le temps de réponse de chaque tâche.



5. Commenter les résultats trouvés et indiquer le quel des ordonnancements (RM ou EDF) est le plus adéquat (meilleur) pour notre système..

5.Analyse de l'ordonnancement multi-processeur :

Refaire les questions de la partie 4 avec la configuration suivante :

$Taches$	$T(ms)$	$C(ms)$	$r(ms)$	$D(ms)$	$Processeur$
N_L	120	20	0	120	1
N_F	120	10	0	120	2
P_L	40	5	0	40	1
P_F	40	5	0	40	2
F_L	10	2	0	10	2
F_F	10	1	0	10	2
A_P	10	1	0	10	1

6.Implémentation sur carte Arduino UNO

Nous allons simuler chaque tache par une diode LED "de differents couleurs"

1. Implémenter les différentes taches la carte arduino à travers l'environnement de simulation Proteus ISIS et l'environnement Arduino IDE.