

Rapport de stage Ingénieur

-

Implémentation d'un ordonnanceur temps réel sur
plateforme multi-cœur hétérogène

-

BELPOIS Vincent

2023



Table des matières

Présentation du stage	3
0.1 Le L.I.A.S.	3
0.2 Le sujet du stage	3
1 LITMUS^{RT}	3
1.1 Présentation de LITMUS ^{RT}	3
1.2 Présentation de <i>feather-trace</i>	3
1.3 Implémentation d'un ordonnanceur EDF partitionné	3
1.3.1 Algorithme considéré	3
1.3.2 Implémentation	3

Présentation du stage

0.1 Le L.I.A.S.

0.2 Le sujet du stage

1 LITMUS^{RT}

1.1 Présentation de LITMUS^{RT}

1.2 Présentation de *feather-trace*

1.3 Implémentation d'un ordonnanceur EDF partitionné

1.3.1 Algorithme considéré

On cherche alors pour commencer à implémenter un algorithme d'ordonnancement simple afin de se familiariser avec les méthodes et fonctions fournis par LITMUS^{RT}. J'ai donc choisi un algorithme partitionné pour la simplicité d'ordonnancement par processeur que cela offre. Un algorithme EDF (*Earliest Deadline First*) est alors choisi pour la simplicité du choix de la tâche à exécuter. Comme son nom l'indique, on choisit à chaque instant la tâche ayant l'échéance la plus proche. On nommera par la suite cet algorithme P-EDF (*Partitionned Earliest Deadline First*).

Pour montrer le fonctionnement de cet algorithme, si l'on se place sur un même processeur, on peut visualiser l'exécution de deux tâches périodiques :

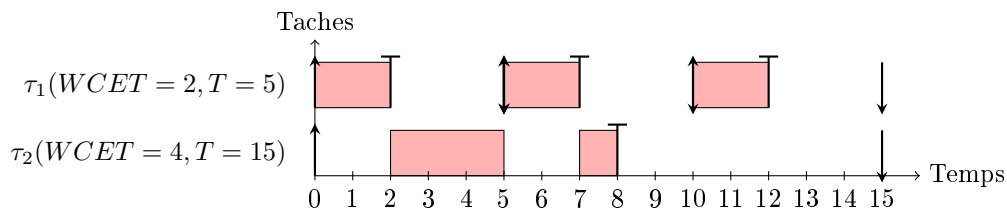


FIGURE 1 – Exemple de EDF à 2 tâches

On a ici une première tâche τ_1 avec un pire temps d'exécution (*Worst Case Execution Time*) de 2 et une période de 5, et une seconde tâche τ_2 avec un pire temps d'exécution de 4 et une période de 15. On a alors la préemption de la τ_2 à $t = 5$ afin d'exécuter τ_1 . Cela est dû au réveil de la tâche τ_1 (représenté par la flèche montante) et à la date d'échéance plus proche de cette dernière.

1.3.2 Implémentation

Expliquer ce qu'est un module dans le noyau Linux.

- Montrer ce qui est propre à la définition du module
- Montrer l'emplacement des fichiers que l'on va créer dans le noyau
- Montrer les modifications du make file

```
1 #include <stdio.h>
2
3 int main() {
4     printf("Hello, world!");
5     return 0;
6 }
```

Glossaire

processeur Ca c'est la définition. 3