

Rapport de stage Ingénieur

-

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

-

BELPOIS Vincent

2023



## Table des matières

<b>Présentation du stage</b>	<b>3</b>
0.1 Le L.I.A.S. . . . . .	3
0.2 Le sujet du stage . . . . .	3
<b>1 Systèmes d'exploitation compatibles avec la carte ROCK960</b>	<b>4</b>
1.1 Présentation de la carte de développement . . . . .	4
1.2 Installation d'un système d'exploitation . . . . .	4
1.2.1 Installation d'une image précompilée . . . . .	4
1.2.2 Compilation de Linux depuis le code source . . . . .	4
1.2.3 Compilation croisée . . . . .	4
1.3 Etude des versions de Linux compatibles . . . . .	4
1.4 Comment Linux gère le support d'un processeur . . . . .	4
1.5 Etude . . . . .	4
<b>2 LITMUS<sup>RT</sup></b>	<b>5</b>
2.1 Présentation de LITMUS <sup>RT</sup> . . . . .	5
2.2 Présentation de <i>feather-trace</i> . . . . .	5
2.3 Implémentation d'un ordonnanceur EDF partitionné . . . . .	5
2.3.1 Algorithme considéré . . . . .	5
2.3.2 Implémentation . . . . .	5

## Présentation du stage

### 0.1 Le L.I.A.S.

### 0.2 Le sujet du stage

Mon stage s'intéresse à l'implémentation d'un ordonnanceur sur plateforme hétérogène[1]

# 1 Systèmes d'exploitation compatibles avec la carte ROCK960

## 1.1 Présentation de la carte de développement

## 1.2 Installation d'un système d'exploitation

### 1.2.1 Installation d'une image précompilée

Image Linux ubuntu fournie par 96boards

### 1.2.2 Compilation de Linux depuis le code source

Où est le code source linux ?

Comment changer de version du code cloné.

### 1.2.3 Compilation croisée

Toolchain : qu'est ce que c'est

Variables d'environnement

Réalisation de scripts linux pour accélérer le développement

Copy de l'image du noyau pour faire encore plus rapide

## 1.3 Etude des versions de Linux compatibles

## 1.4 Comment Linux gère le suport d'un processeur

## 1.5 Etude

## 2 LITMUS<sup>RT</sup>

### 2.1 Présentation de LITMUS<sup>RT</sup>

### 2.2 Présentation de *feather-trace*

### 2.3 Implémentation d'un ordonnanceur EDF partitionné

#### 2.3.1 Algorithme considéré

On cherche alors pour commencer à implémenter un algorithme d'ordonnement simple afin de se familiariser avec les méthodes et fonctions fournis par LITMUS<sup>RT</sup>. J'ai donc choisi un algorithme partitionné pour la simplicité d'ordonnement 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 :

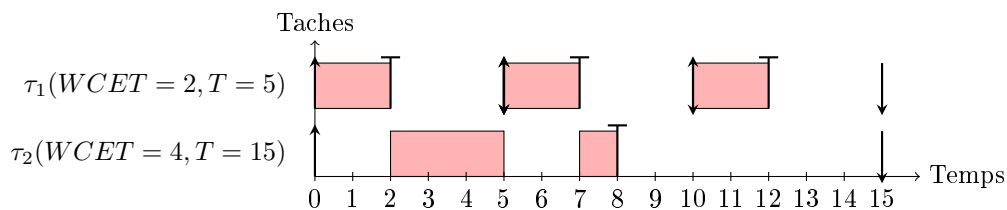


FIGURE 1 – Exemple de EDF à 2 tâches

On a ici une première tâche  $\tau_1$  avec un pire temps d'exécution (*Worst Case Execution Time*) de 2 et une période de 5, et une seconde tâche  $\tau_2$  avec un pire temps d'exécution de 4 et une période de 15. On a alors préemption de la  $\tau_2$  à  $t = 5$  afin d'exécuter  $\tau_1$ . Cela est dû au réveil de la tâche  $\tau_1$  (représenté par la flèche montante) et à la date d'échéance plus proche de cette dernière.

#### 2.3.2 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

## Références

- [1] Antoine Bertout, Joël Goossens, Emmanuel Grolleau, and Xavier Poczekajlo. Workload assignment for global real-time scheduling on unrelated multicore platforms. In *Proceedings of the 28th International Conference on Real-Time Networks and Systems*, pages 139–148, 2020.

## Table des figures

1	Exemple de EDF à 2 tâches . . . . .	5
---	-------------------------------------	---

## Glossaire

<b>plateforme hétérogène</b>	Système formé d'un ensemble de processeurs différents.	3
<b>processeur</b>	Ca c'est la définition.	5