Université Paul Sabatier

Systèmes Temps Réel

Compte Rendu de TP Sujet -

Auteurs :
David TOCAVEN
Lucien RAKOTOMALALA

Encadrant :
Hamid DEMMOU

17 décembre 2017

Table des matières

Introduction **1**

1 TP 1 : Iniiation a un OS temps Réel basé sur Linux **2**
1.1 Mesures sous linux 2
1.1.1 Programme *carrelinux – comedi.c* 2
1.2 Mesures sous RTAI 4

2 **5**

3 **6**

4 Conclusion **7**

Annexes **9**

TITRE **9**
TITRE 9

Annexe 2 - TITRE **10**

Introduction

Chapitre 1

TP 1 : Iniiation a un OS temps Réel basé sur Linux

1.1 Mesures sous linux

1.1.1 Programme *carrelinux − comedi.c*

Ce premier programme est un générateur de signal carré. Il va nous permettre d'analyser les réponses temps réel de en étant basé sur Linux. Notre première analyse du programme donne :

- Fonction *Void out* : envoie un signal inverse celui qu'elle envoyait précédemment. La fréquence semble être défini ailleurs dans le programme. Le signal est envoyé vers le port 0 de la carte E/S initialisé dans la fonction *main*.
- *Carte E/S* : la librairie *Comedio* permet d'ouvrir la connexion série avec la carte d'entrée sortie puis d'initialiser le sens du port.
- *main* : Initialisation d'une structure de temps dans le main qui sera utilisé dans la boucle infinie du programme principal.

Le programme entre ensuite dans ue bucle infinie dans laquelle il attend un temps *TIMER_ABSTIME* pour ensuite appeler la fonction *out*. Après, le programme calcule le *next shot*, i.e le prochain déclenchement, qu'il devra attendre lors du recommencement de la boucle.

Avec ce recueil d'informations, nous sommes capable de définir la fréquence du signal du signal carré que nous allons observer : elle est égale à 2 fois le *next shot* calculé dans la boucle infinie : ce calcul est : $next\_shot = 50000 + t$, la variable t appartient à la structure de temps et elle est défini en nanosecondes donc :

$$f = 2 \times 50000ns \Leftrightarrow f = 100\mu s \tag{1.1}$$

Mesures des période du signal Pour mesurer les modifications de période, nous avons crée deux variables de type *timespec* : une qui mesure le temps précédent le sleep, une qui mesure a la fin de l'instance *while*(1). La mesure de la demi-période du signal carré δ est alors la différence entre le temps du début de la boucle et le temps en fin de boucle.

Pour permettre un affichage correct, nous avons introduit dans notre programme une fonction qui permet d'écrire dans un fichier les 5000 dernières δ mesurées et qui sera appelé dès que le signal **Ctrl+C** grâce à l'instruction :

```
signal(SIGINT, IntHandler)
```

Nous utilisons ensuite le fichier de mesure en .res pour un affiche avec un script *gnuplot*. Nous observons les résultats suivants :

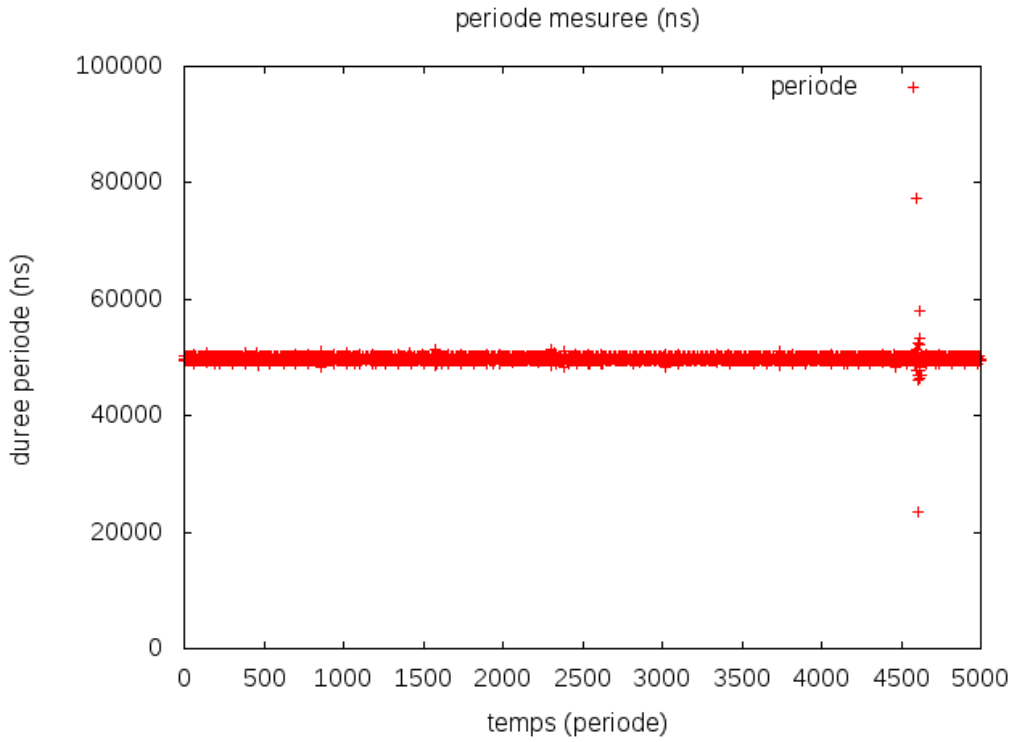

FIGURE 1.1 – Meure des périodes sans perturbation

Pour cette première observation, nous n'avons pas touché le système d'exploitation pendant la mesure des périodes et cependant, nous remarquons que celles ci ont eu out de même quelques légères perturbations. Nous allons maintenant effectué quelques perturbations pendant que le relevé s'effectue.

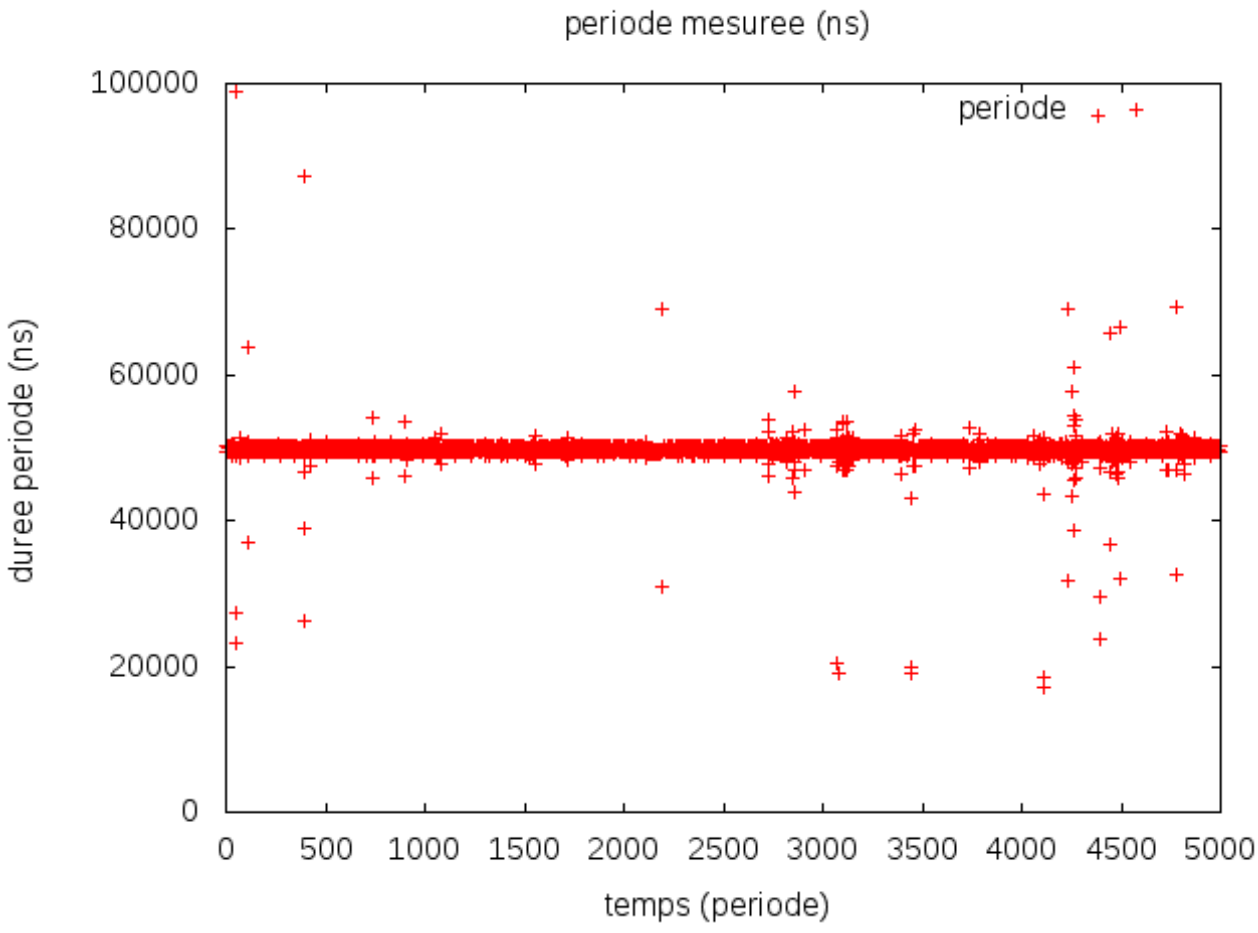

FIGURE 1.2 – Mesures des périodes avec quelques perturbations

Nous notons ici que les petites perturbations commencent à avoir de grosse conséquence sur la demi période du signal. Ces perturbations se notent aussi sur l'oscilloscope où nous observons des modifications du signal. Pour terminer cette étude, nous effectuons une dernière mesure des périodes dans laquelle nous demandons au système d'exploitation une concaténation de fichier volumineux.

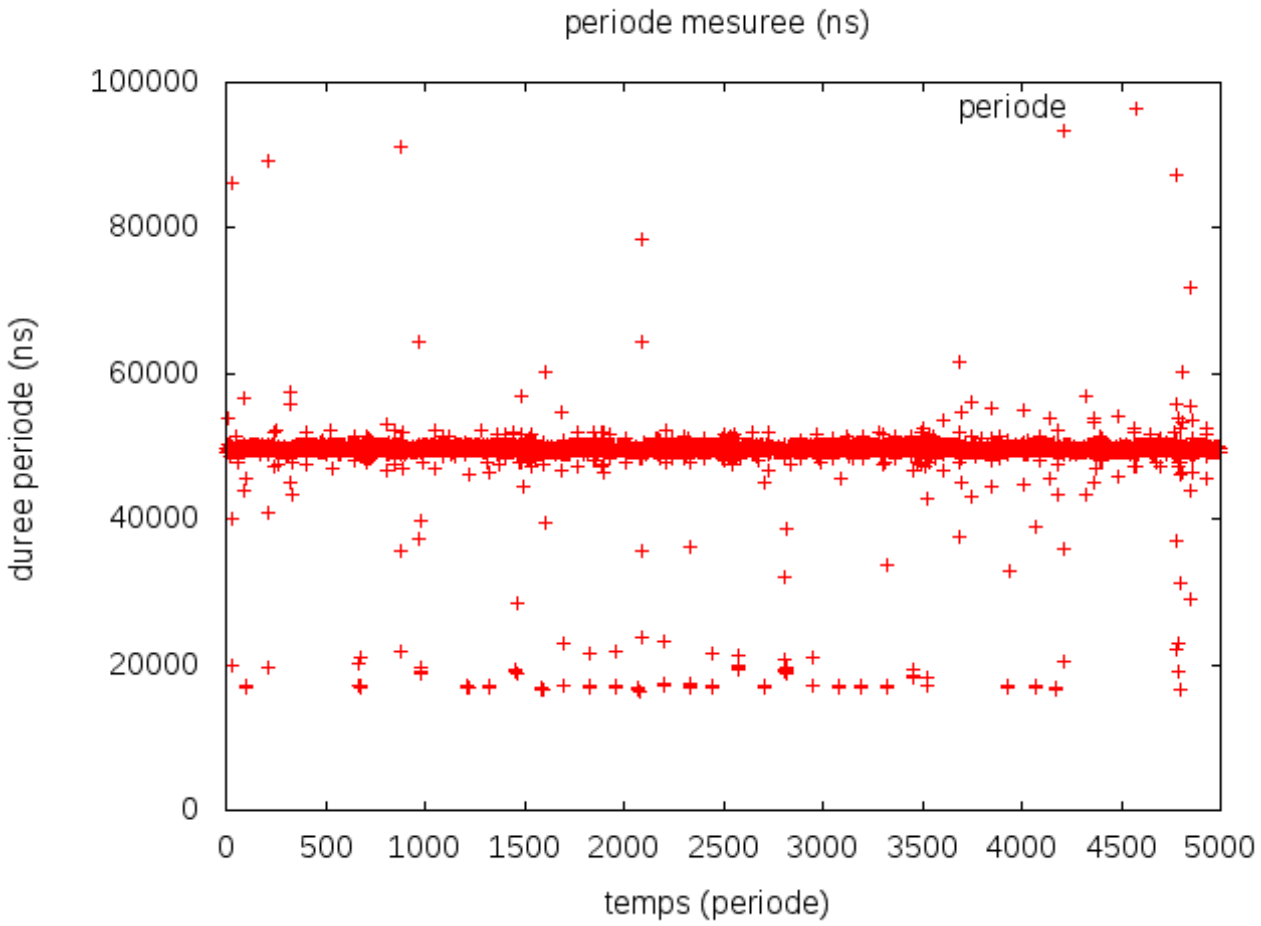

FIGURE 1.3 – Mesures des périodes avec la commande "cat /dev/zer > file2delete" en parallèle

Nous observons que la dispersion des périodes est beaucoup plus importante. Le système d'exploitation n'a pas réussi à ordonnancer notre programme avec l'instruction que nous lui avons demandé. De telles différences ne sont pas acceptables pour des applications ou des systèmes Temps Réel. Le système d'exploitation Linux seul ne peut pas être utilisé tel quel pour le Temps réel, nous devons ajouter au noyau la capacité à préempter des tâches avec l'architecture logicielle RTAI.

1.2 Mesures sous RTAI

Utilisation de la ligne :

Chapitre 2

Chapitre 3

Chapitre 4

Conclusion

Annexes

Annexe 1 - TITRE

Annexe 2 - TITRE