

# Amélioration de l'Asservissement de Visée Laser

Soutenance de fin de projet : 20 janvier 2025

Client : Pierre Chambert (JTL-Electronique)

Tuteur industriel : François Kersulec

Professeur Référent : Jacques Laffont



# Plan de la présentation

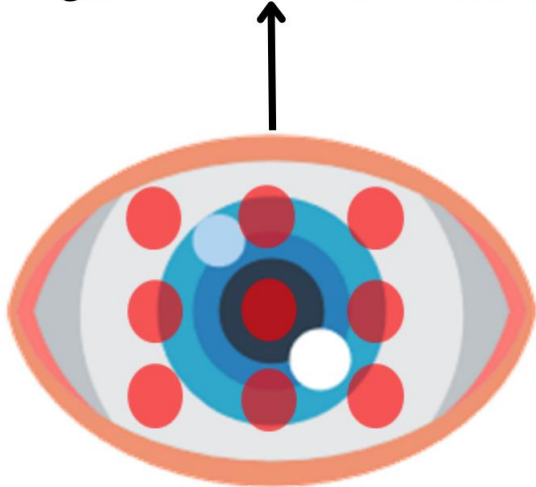
- Contexte
- Présentation du système
- Cahier des charges
- Objectifs
- Travail réalisé et résultats obtenus
- Conclusion

# Contexte

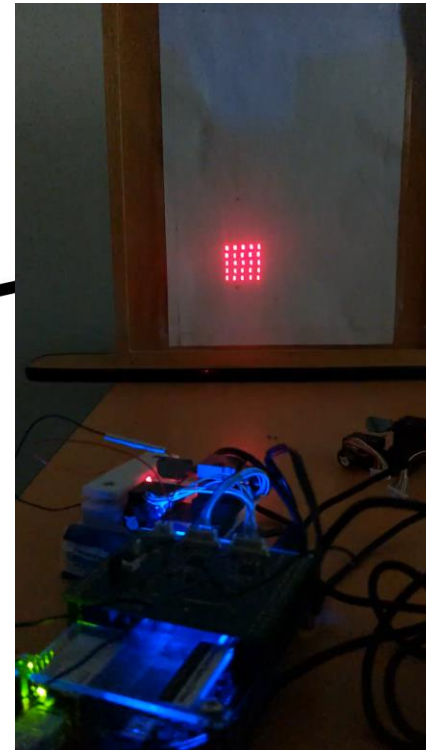
Les personnes diabétiques souffrent de lésions oculaires.

Afin d'éviter d'éventuelles complications, les chirurgiens viennent blesser l'œil à l'aide d'un laser afin de provoquer une cautérisation d'une zone souffrante de la rétine.

Patient diabétique dont les vaisseaux sanguins oculaires sont abimés



Matrice de points utilisée pour effectuer les opérations



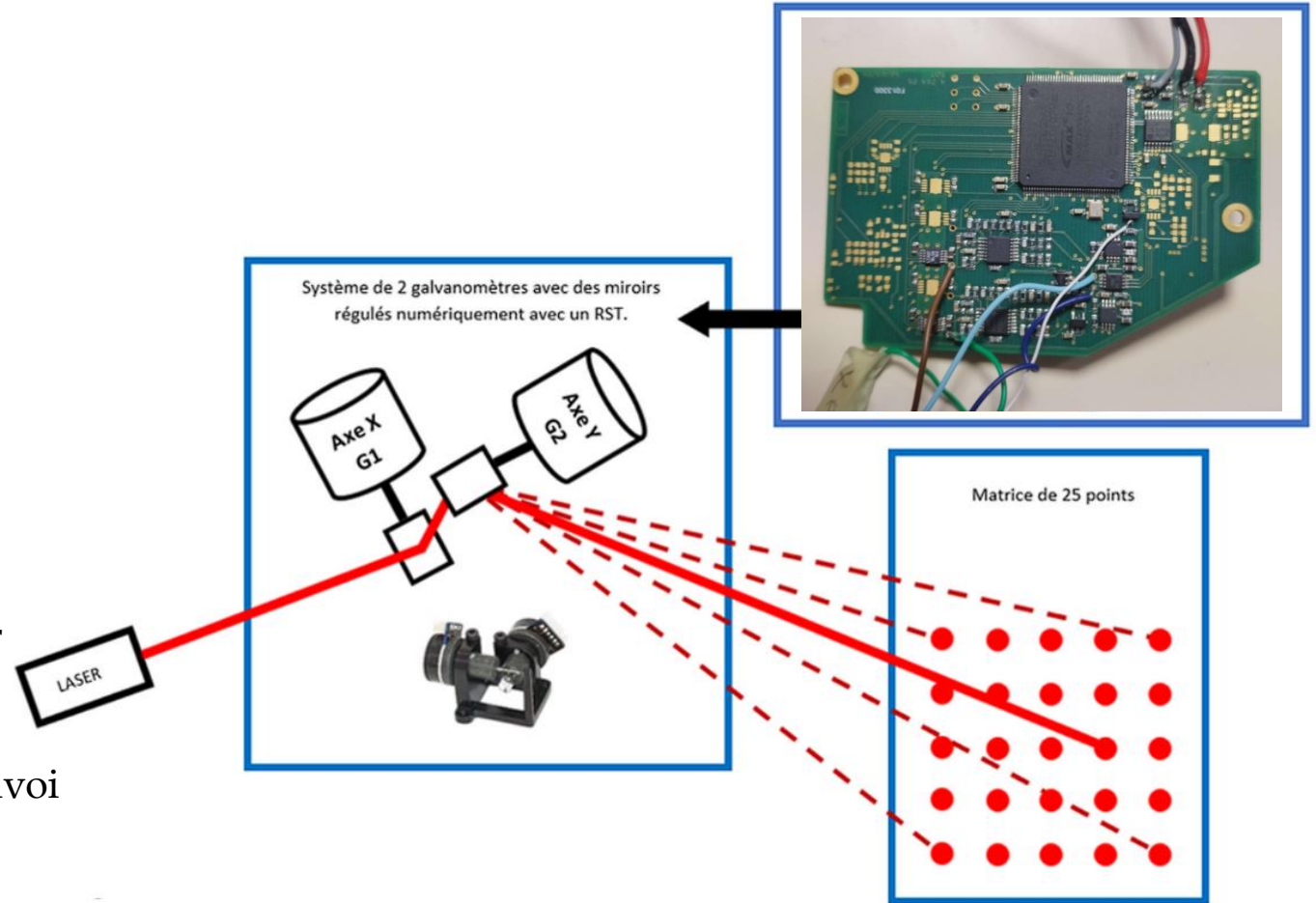
Notre système

# Présentation du système

- Une carte FPGA intégrant un Nios II
- Un ensemble de deux galvanomètres utilisés pour diriger le laser

## Solution retenue les années précédentes :

- **Utilisation d'un correcteur RST par placement de pôles.**
- Communication entre la carte et un PC pour réaliser l'identification du système.
- Calcul des coefficients RST sur le PC puis envoi de ces derniers ainsi que de la consigne à la carte.
- Calcul des commandes au sein de la carte grâce à la programmation du correcteur en VHDL.



# Schéma synoptique du système

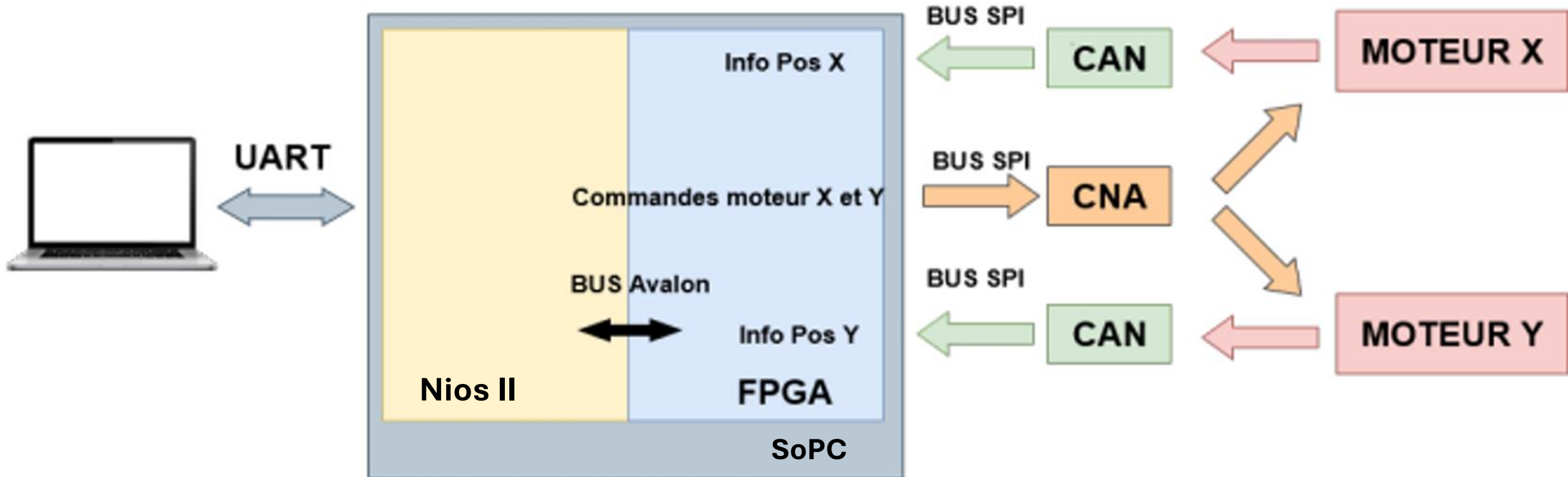


Schéma synoptique du système de visée laser

# Cahier des charges

Type	Nature	Détails
Fonction	Affichage d'une matrice de 25 points	<ul style="list-style-type: none"> <li>Distance entre deux points consécutifs : 11mm pour une distance de 30cm entre les miroirs et la cible.</li> <li>Sans scintillements : affichage de la matrice complète en 20ms.</li> </ul>
Contrainte	Temps de déplacement et de stabilisation d'un point	800µs par point : <ul style="list-style-type: none"> <li>Temps de montée : 400µs</li> <li>Temps de stabilisation : 400µs</li> </ul>
Contrainte	Contrôle de deux galvanomètres	Alimentation en -5V/+5V
Contrainte	Récupération des données sur la position des galvanomètres	Récupération de la position sur un CAN 14 bits
Contrainte	Calculs des commandes au sein de la carte	<ul style="list-style-type: none"> <li>Utilisation du FPGA pour implémenter le correcteur RST</li> <li>Envoi des données sur un CNA 16 bits</li> </ul>
Contrainte	Communication entre Scilab et la carte via le port série	Les caractéristiques du galvanomètre doivent être redéterminés : <ul style="list-style-type: none"> <li>Identification du système</li> <li>calcul des coefficients du RST</li> <li>envoi des coefficients dans le FPGA</li> </ul>

# Objectifs

## Deux parties :

- Une partie automatique
- Une partie programmation de la carte

## Deux objectifs complémentaires :

- Trouver une identification valide et calculer les coefficients du correcteur RST à envoyer à la carte avec Scilab.
- Reproduire les calculs du correcteur RST en VHDL

# Identification du système



# Identification SBPA

- Envoi en entrée d'un signal binaire pseudo aléatoire afin de déterminer les caractéristiques du système
- Détermination du temps de montée et du temps d'échantillonnage
- Comparaison des deux méthodes (Pseudo inverse et variables instrumentales)
- Conversion des coefficients discrets en paramètres continus du système

# Modèle du système à identifier

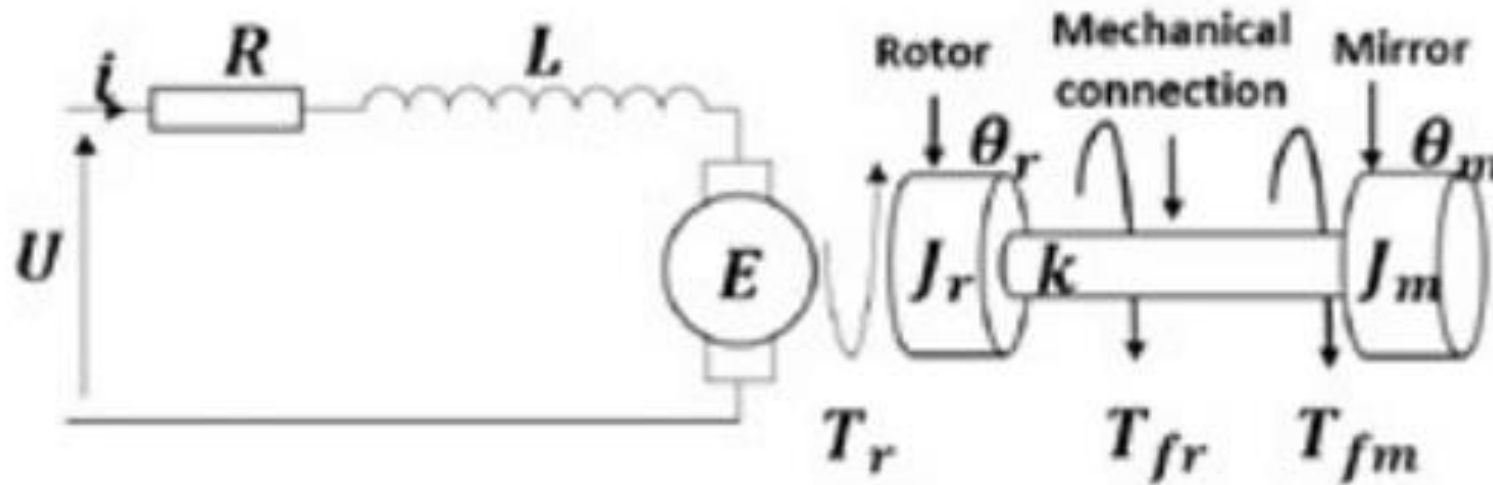


Schéma électrique et mécanique d'un galvanomètre

$$G(p) = \frac{\theta_r(p)}{U(p)} = \frac{K}{(R + Lp)(J_r p^2 + Bp + K_\tau)}$$

# Détermination du temps de montée et du temps d'échantillonnage

- Temps de montée  $\approx 2 \text{ ms}$
- Temps d'échantillonnage compris entre 3 à 12 fois plus petit au temps de montée
- $T_e = 120 \mu s$

# Comparaison des deux méthodes d'identification

Pseudo inverse :

$$\underbrace{\begin{bmatrix} S_k \\ S_{k-1} \\ S_{k-2} \\ \vdots \end{bmatrix}}_V = \underbrace{\begin{bmatrix} -S_{k-1} & \cdots & -S_{k-n} & E_{k-1} & \cdots & E_{k-m} \\ -S_{k-2} & \cdots & -S_{k-n-1} & E_{k-2} & \cdots & E_{k-m-1} \\ -S_{k-3} & \cdots & -S_{k-n-2} & E_{k-3} & \cdots & E_{k-m-2} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \end{bmatrix}}_M \times \underbrace{\begin{bmatrix} a_1 \\ \vdots \\ a_n \\ b_1 \\ \vdots \\ b_m \end{bmatrix}}_{\Theta}$$

On observe que l'on obtient une équation  $V = M\Theta$ .

où  $\Theta$  contient les paramètres du système à identifier,  $M$  et  $V$  les échantillons d'entrée et de sortie.

$$\Theta = ((M^T \times M)^{-1} \times M^T) \times V$$

- Bruit au carré car certains termes de la matrice  $M$  vont être multiplié par eux-mêmes

Variables instrumentales :

- Même principe que la pseudo inverse

$$\Theta_{vi} = ((M_{vi}^T \times M)^{-1} \times M_{vi}^T) \times V$$

- Décalage des échantillons de sorties de un

$$M_{vi} = \begin{bmatrix} -S_{k-2} & \cdots & -S_{k-n-1} & E_{k-1} & \cdots & E_{k-m} \\ -S_{k-3} & \cdots & -S_{k-n-2} & E_{k-2} & \cdots & E_{k-m-1} \\ -S_{k-4} & \cdots & -S_{k-n-3} & E_{k-3} & \cdots & E_{k-m-2} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \end{bmatrix}$$

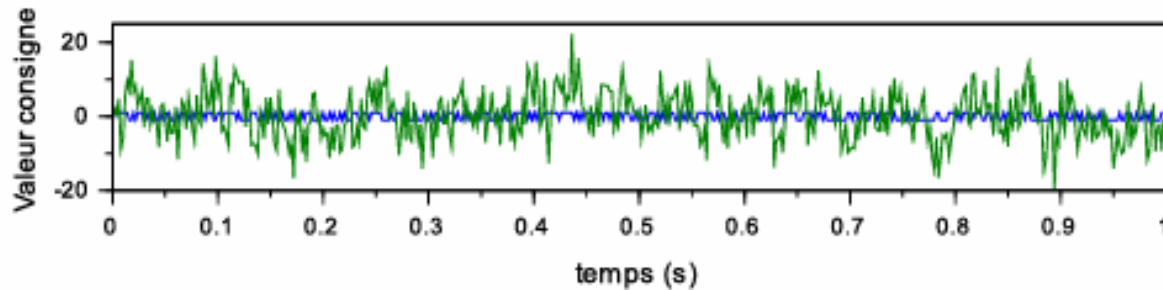
- Avantage : Pas de bruit au carré car on multiplie le bruit avec l'échantillon précédent

# Comparaison des deux méthodes d'identification

- On réalise une acquisition du système avec une première SBPA
- On applique les 2 méthodes d'identification sur cette acquisition
- On refait une acquisition du système avec une seconde SBPA
- On calcule les sorties de chaque méthode d'identification pour cette nouvelle SBPA
- On calcule l'erreur entre la sortie de chaque système identifié et la sortie réelle du système

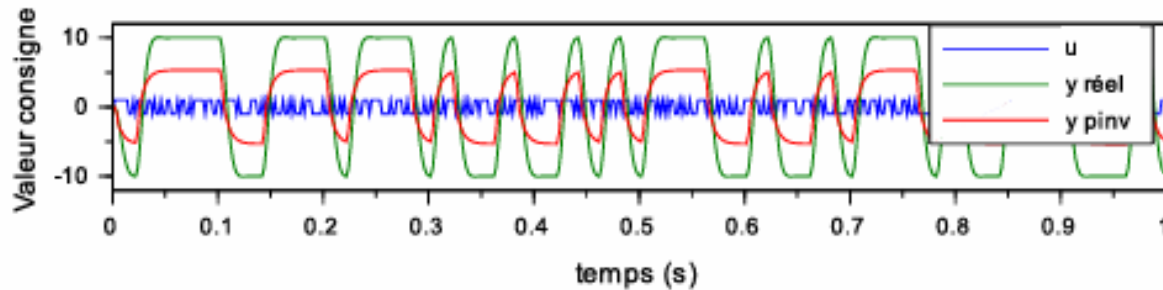
# Comparaison des deux méthodes d'identification

système réel



$$Erreur = \frac{1}{N} \sum_{k=1}^N (y(k) - y_{simu}(k))^2$$

Système avec la méthode de la pseudo-inverse



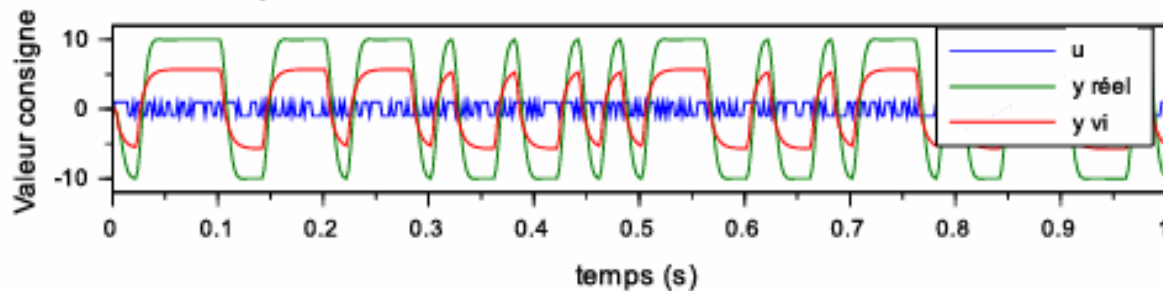
"erreur Pseudo-inverse :"

19.740293

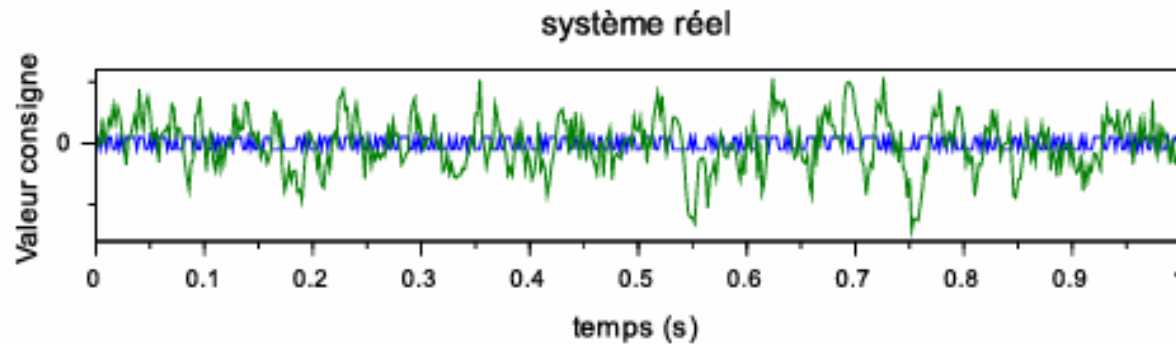
"erreur Variable instrumentale :"

17.077338

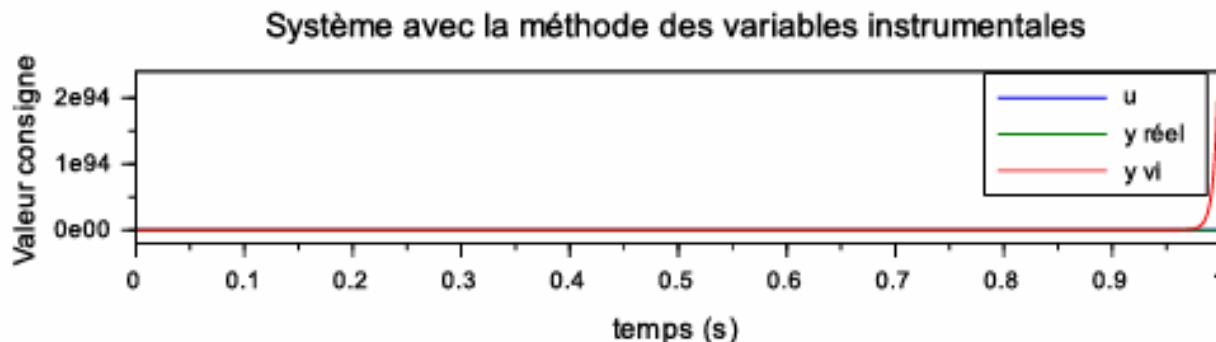
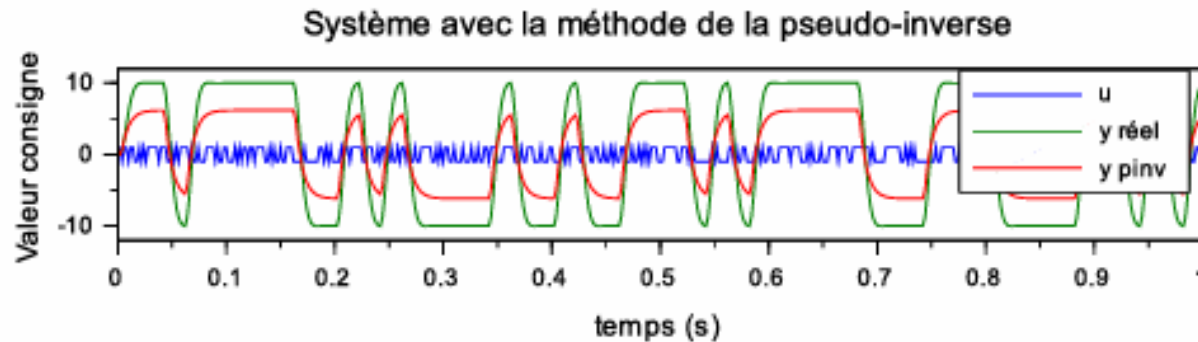
Système avec la méthode des variables instrumentales



# Comparaison des deux méthodes d'identification



```
"erreur Pseudo-inverse :"  
14.222582  
"erreur Variable instrumentale :"  
1.84D+186
```

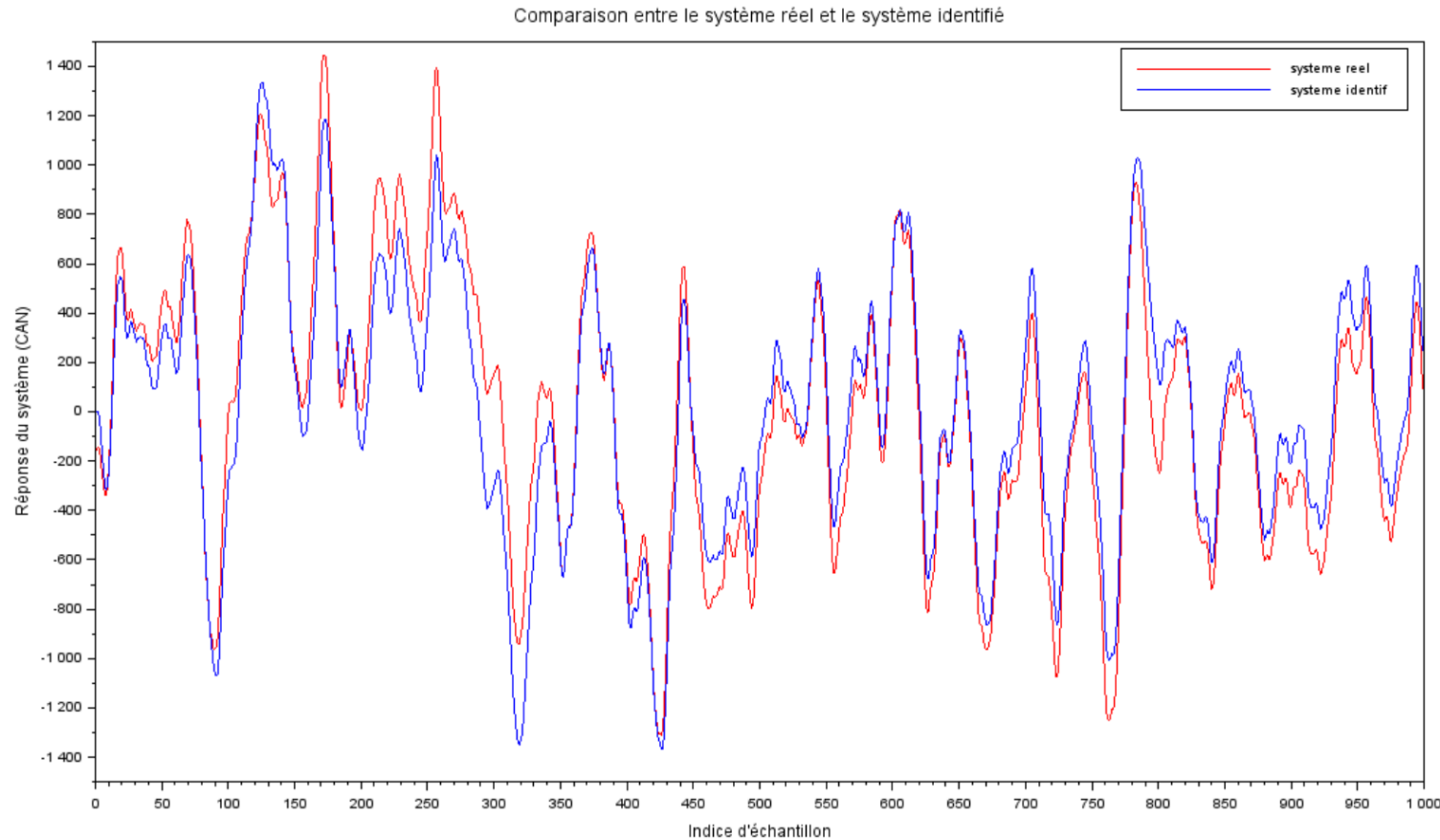


**Vainqueur :**

Méthode de la pseudo inverse



# Résultats de l'identification



```
theta = [5x1 double]
```

```
-1.9786721
```

```
1.1206457
```

```
-0.1294227
```

```
-0.0431328
```

```
-0.0226541
```

**L'identification suit l'allure du système réel, cependant nous ne sommes pas en mesure de prouver qu'elle est suffisamment fiable pour que la correction fonctionne.**



# Conversion des coefficients discrets en paramètres continus du système

## Problème :

- Les coefficients discrets dépendent de la période d'échantillonnage.
- La période d'identification diffère de celle utilisée pour l'asservissement.
- Nécessité de convertir les coefficients discrets identifiés en coefficients continus.

## Solution :

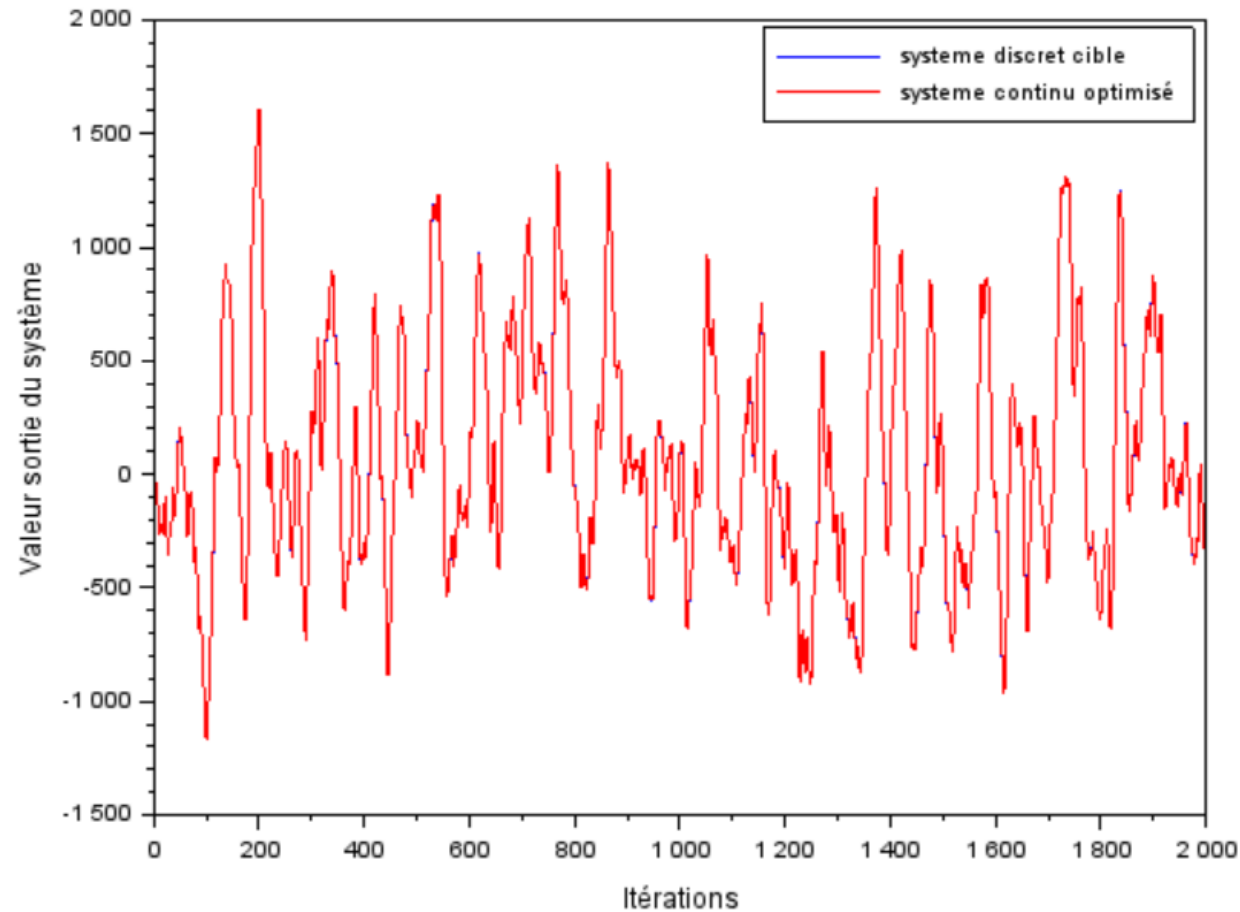
- Utilisation d'un algorithme d'optimisation : l'algorithme d'Adam.

Choix basé sur :

- Performance de l'algorithme.
- Capacité à converger rapidement vers un minimum local.

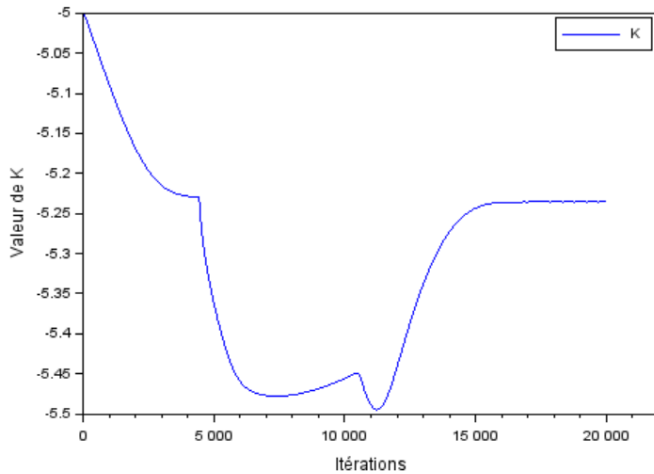
# Résultats avec le système optimisé en continu

Comparaison de la sortie du système identifié discret avec le système optimisé en continu

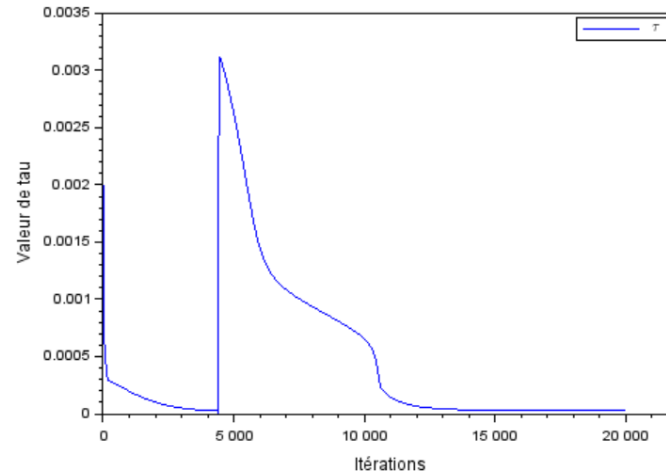


# Résultats avec le système optimisé en continu

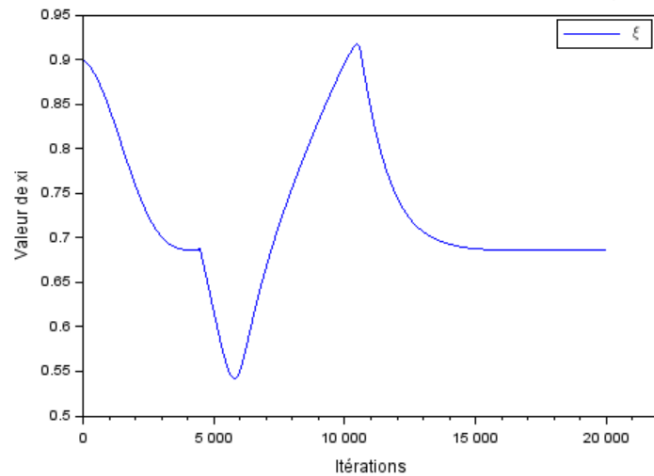
Evolution de la variable K au fur et à mesure des itérations de l'optimisation



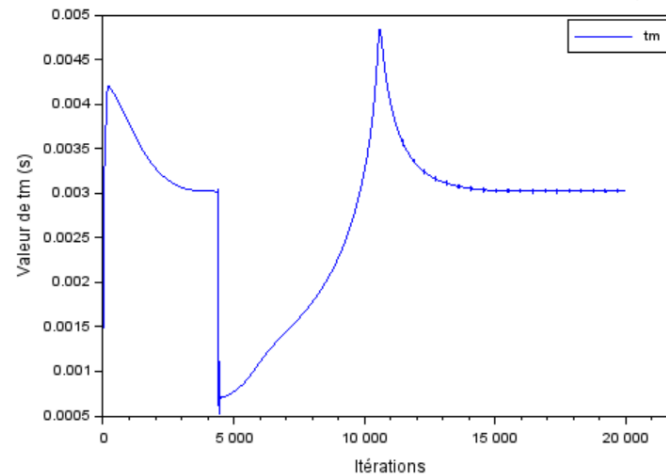
Evolution de la variable tau au fur et à mesure des itérations de l'optimisation



Evolution de la variable xi au fur et à mesure des itérations de l'optimisation



Evolution de la variable tm au fur et à mesure des itérations de l'optimisation



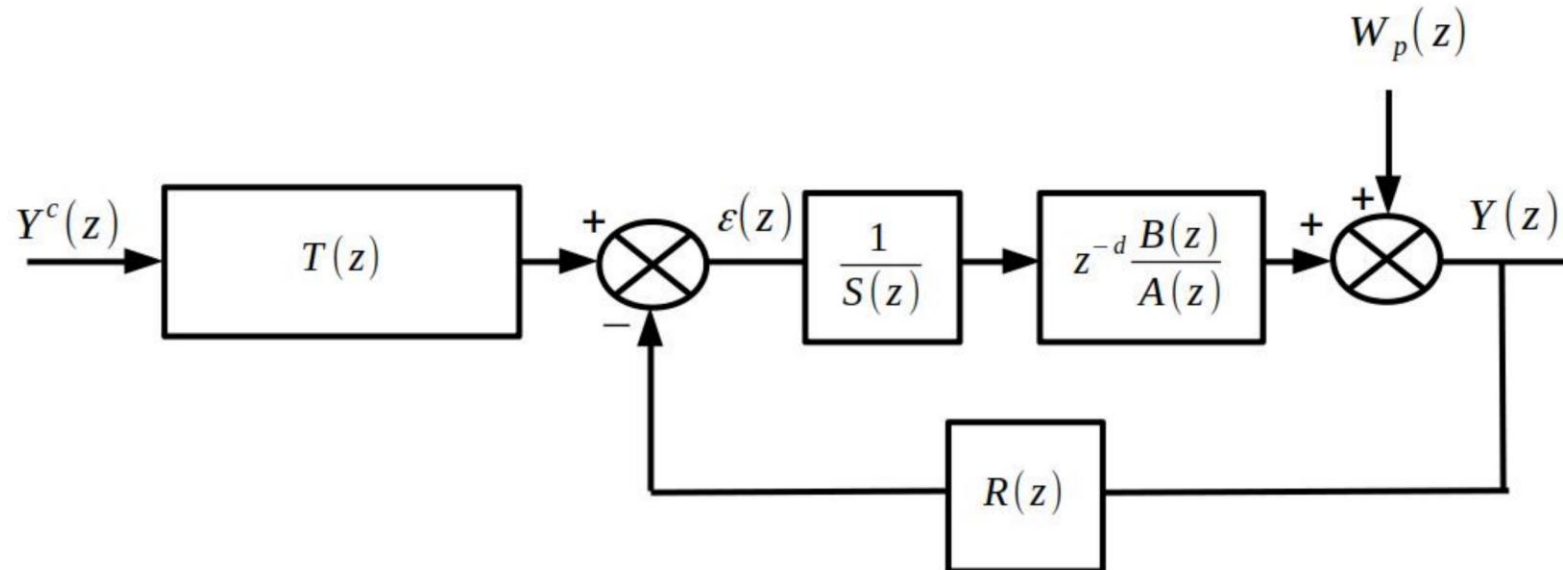
- $K = -5.2355192$
- $\tau = 0.0000307 \text{ s}$
- $\xi = 0.6864961$
- $t_m = 0.0030265 \text{ s}$

# Conception théorique du correcteur RST par placement de pôles

# Pourquoi avoir choisi ce correcteur ?

- Permet de piloter des systèmes linéaires d'ordres quelconque avec des retards qui peuvent être très élevés
- Permet de réaliser un asservissement selon deux degrés de libertés :
  - Dynamique en poursuite (par rapport à la consigne)
  - Dynamique en régulation (par rapport à une perturbation)

# Correcteur RST par placement de pôles



Avec :

$z^{-d} \frac{B(z)}{A(z)}$  : Transmittance du système à asservir,

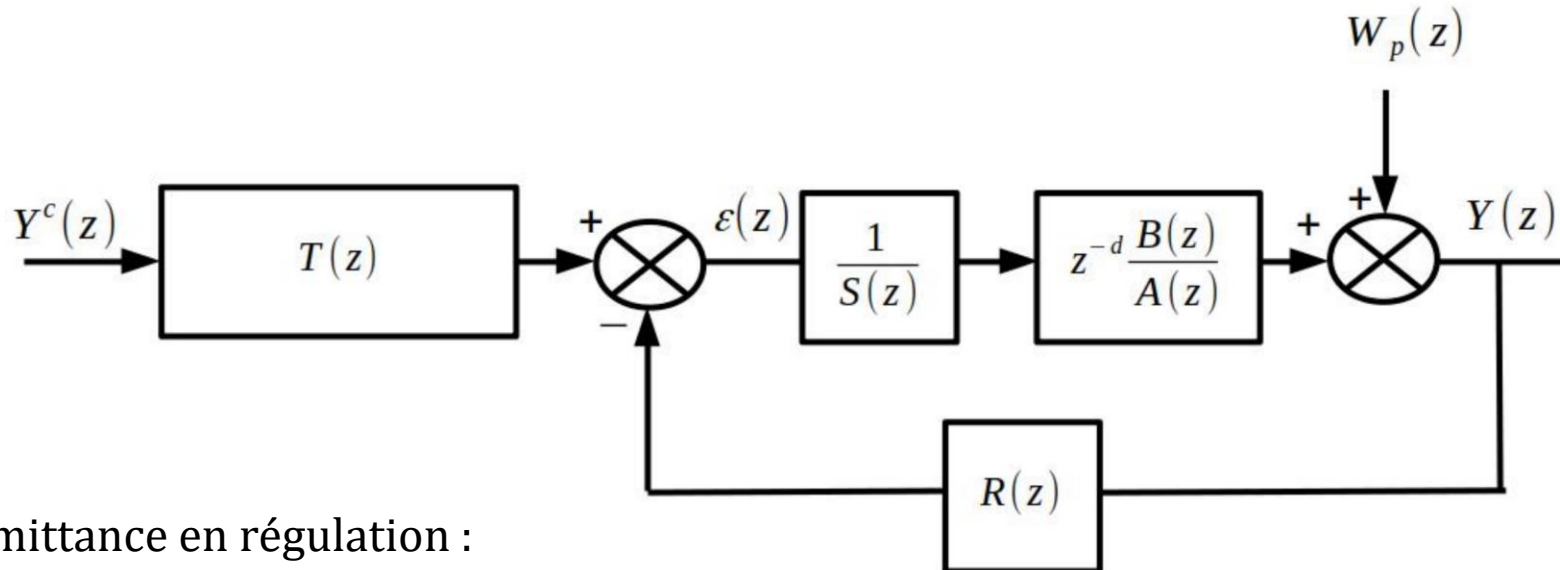
$Y^c(z)$  : Consigne du système,

$Y(z)$  : Sortie du système,

$\varepsilon(z)$  : Erreur entre la consigne et la sortie du système,

$W_p(z)$  : Bruit de mesure.

# Correcteur RST par placement de pôles



Transmittance en régulation :

$$\frac{Y(z)}{W_p(z)} = \frac{A(z)S(z)}{A(z)S(z) + z^{-d}B(z)R(z)} = F_{\text{cible}_{\text{régulation}}}(z) = \frac{H_N(z)}{H_D(z)}$$

Transmittance en poursuite :

$$\frac{Y(z)}{Y^c(z)} = \frac{z^{-d}B(z)T(z)}{A(z)S(z) + z^{-d}B(z)R(z)} = F_{\text{cible}_{\text{poursuite}}}(z) = \frac{B_m(z)}{A_m(z)}$$

# Choix des fonctions cibles

Paramètres des systèmes du second ordre :

- Gain  $K$
- Temps de montée  $t_m$  (ou la pulsation propre  $\omega_n$ )
- Coefficient d'amortissement  $\xi$

$$F(p) = \frac{K}{1 + \frac{2\xi p}{\omega_n} + \frac{p^2}{\omega_n^2}}$$

Système cible en poursuite :

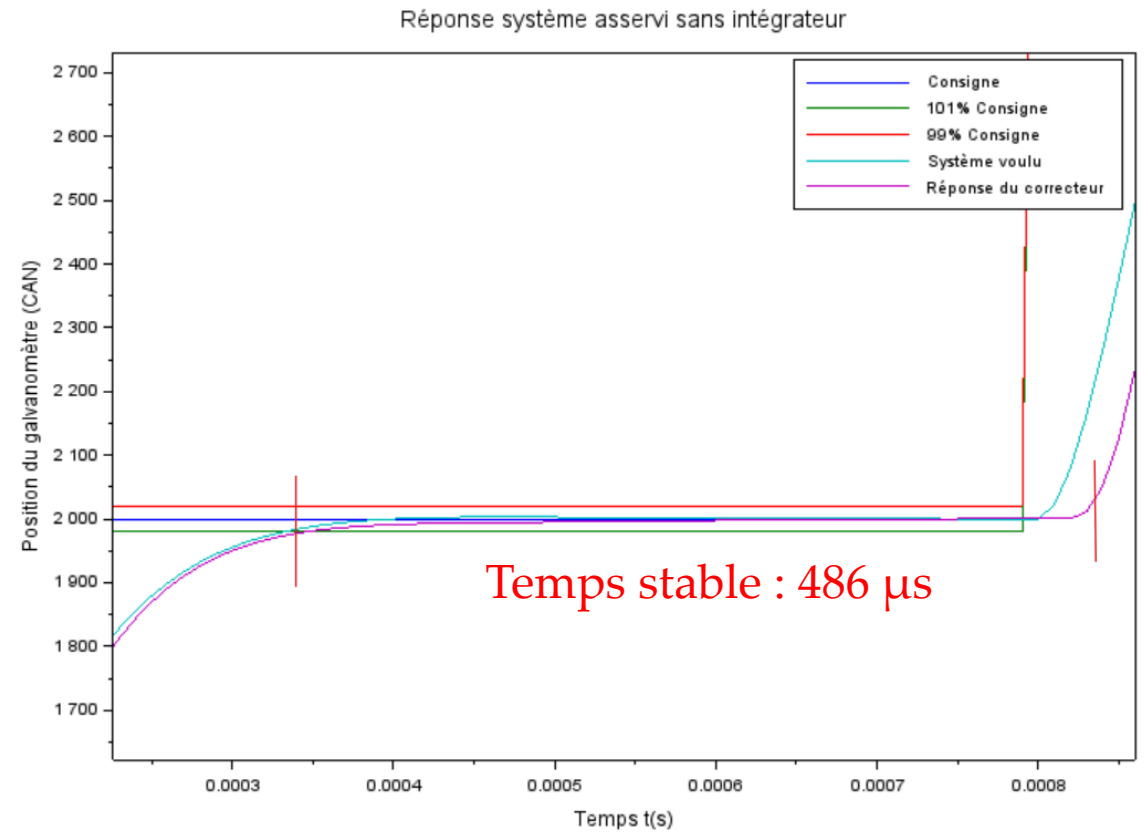
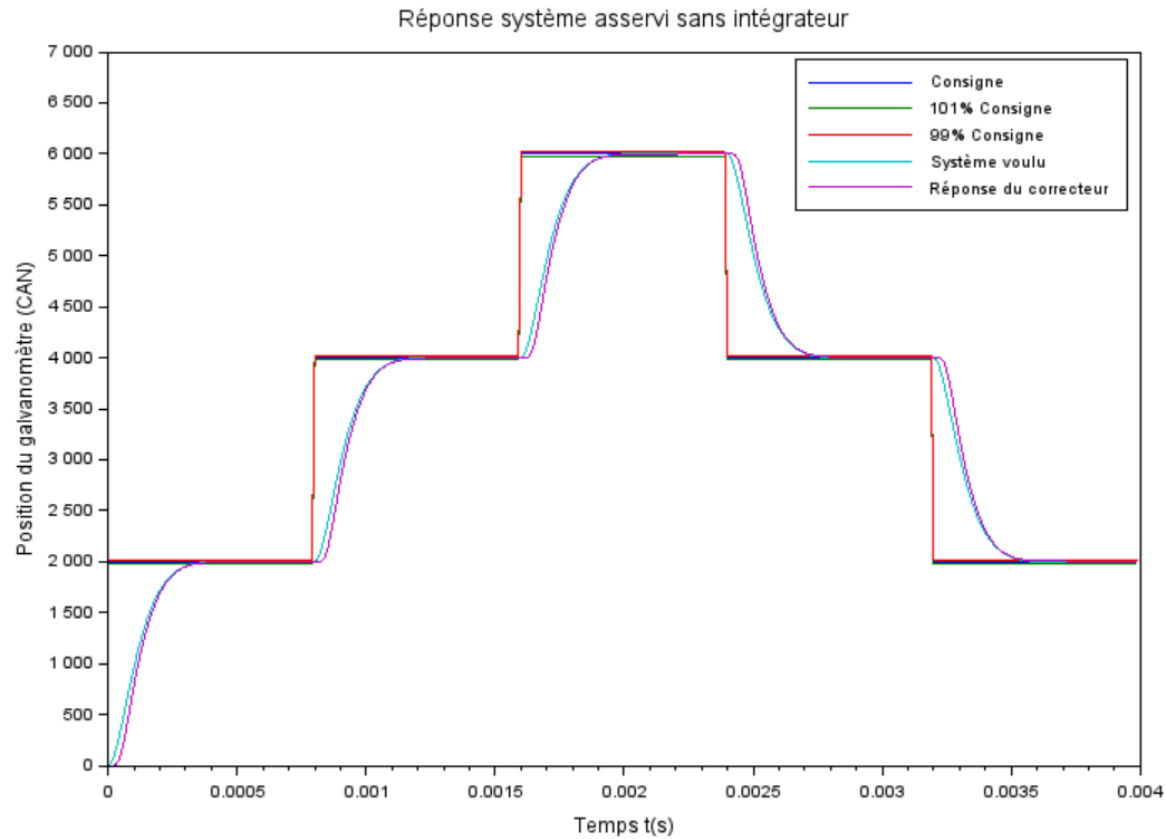
- Gain  $K = 1$
- Temps de montée  $t_m = 400\mu s$
- Coefficient d'amortissement  $\xi = 0,9$

Système cible en régulation :

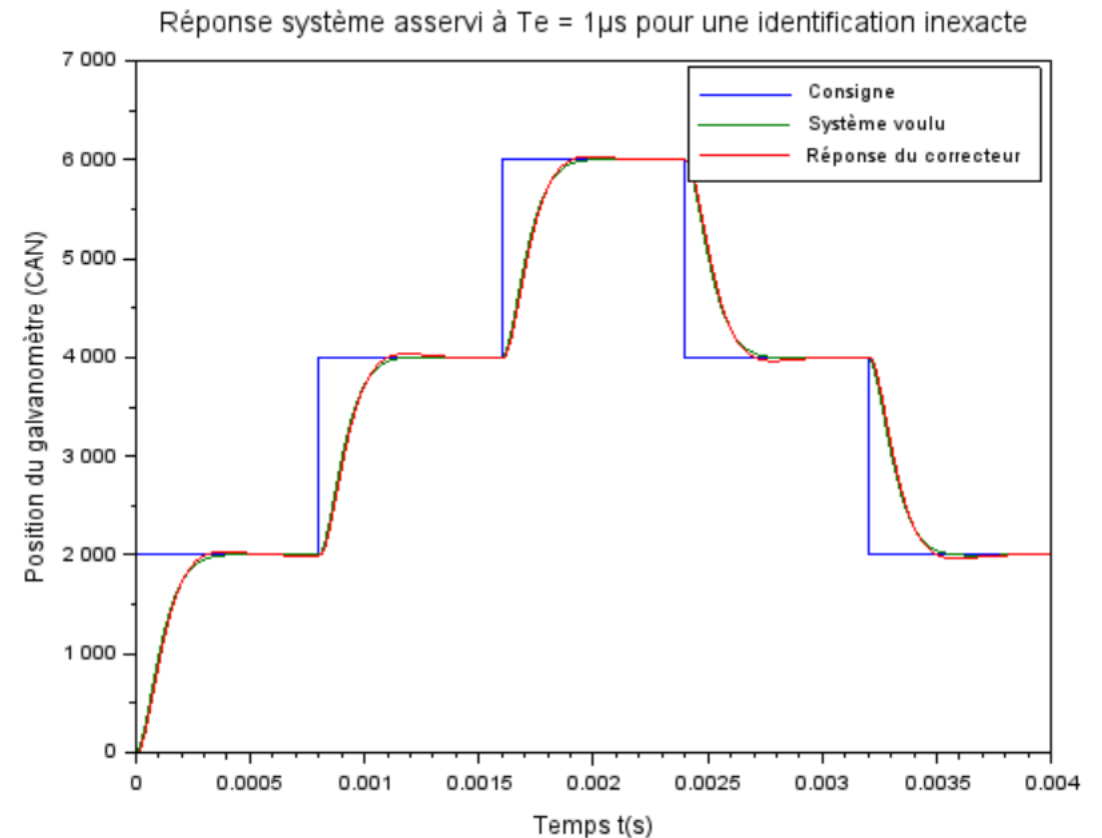
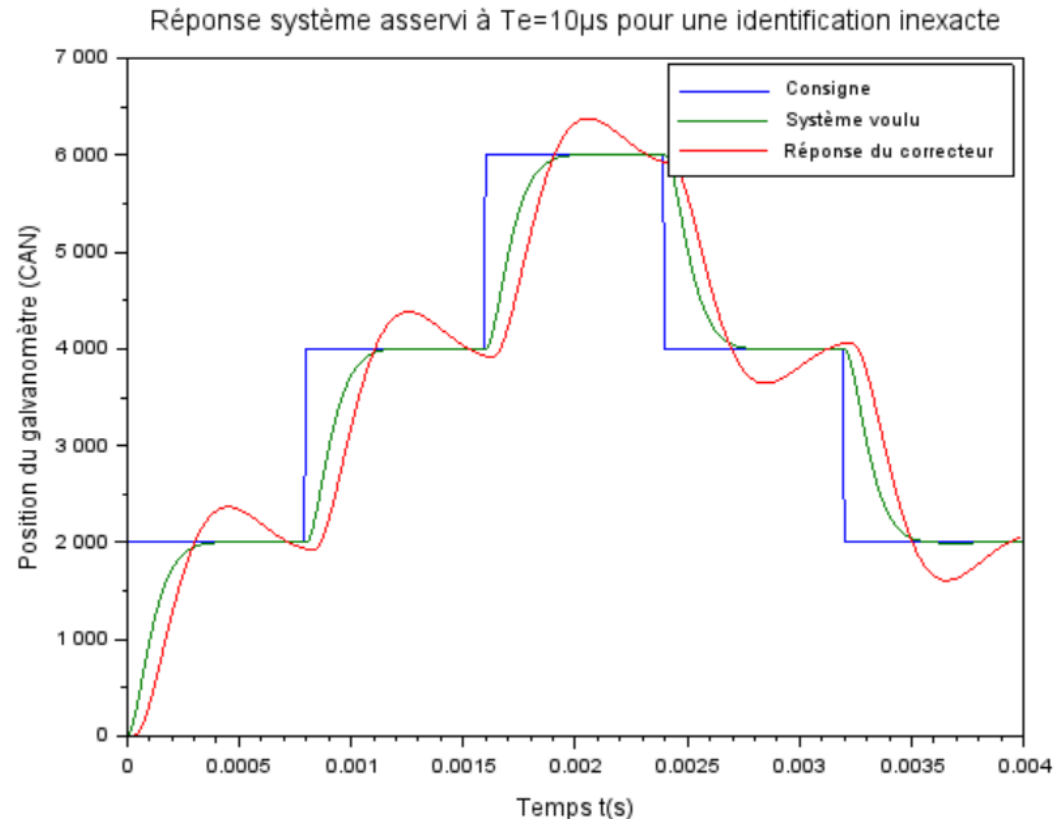
- Gain  $K$  n'importe car pas utilisé
- Temps de montée  $t_m = 300\mu s$
- Coefficient d'amortissement  $\xi = 0,7$



# Réponse du correcteur RST



# Réponse du correcteur RST avec une identification inexacte de 20% sur chaque coefficient



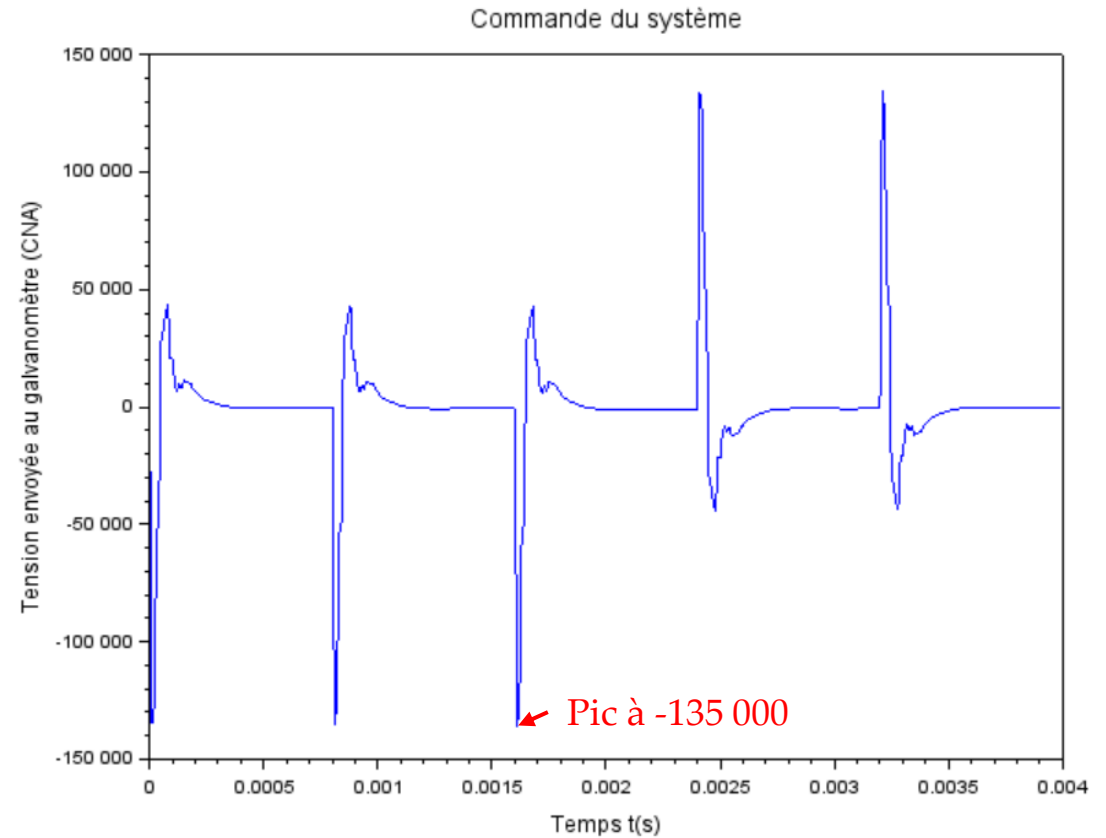
**Le choix d'une période d'échantillonnage plus faible pour l'asservissement peut palier à l'inexactitude de l'identification**

# Commande du correcteur RST

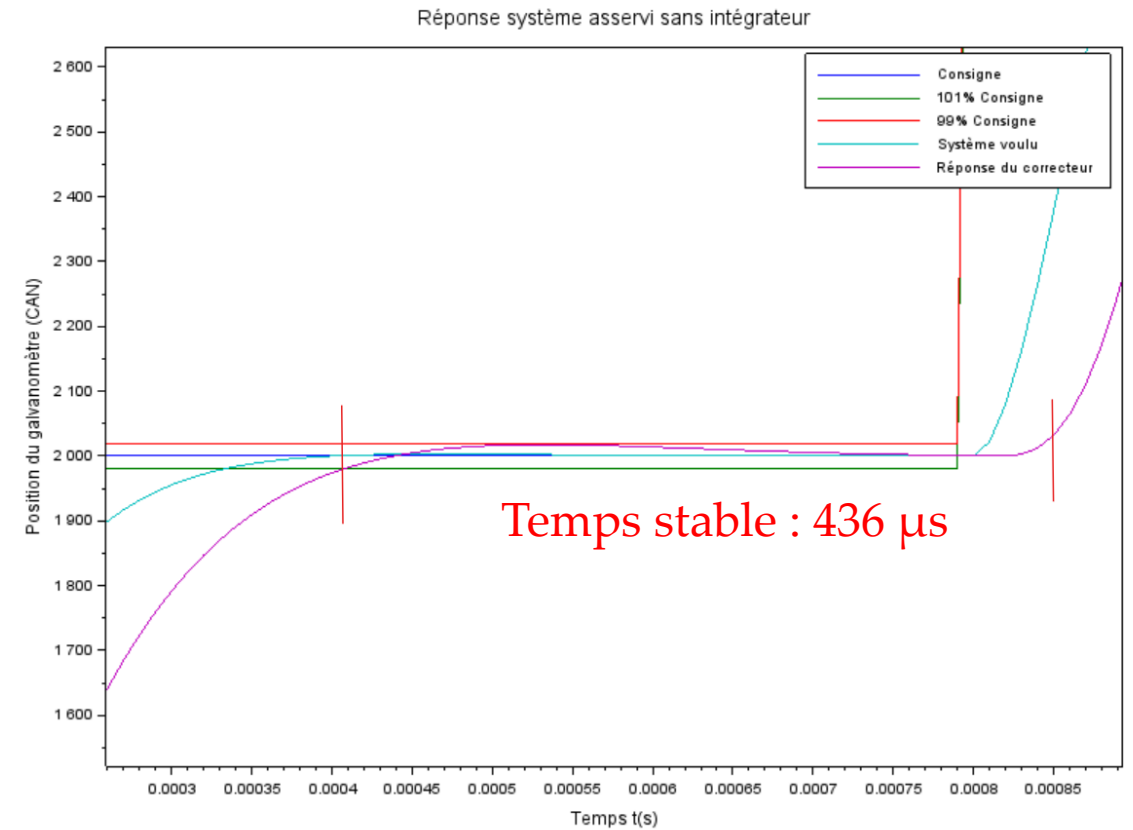
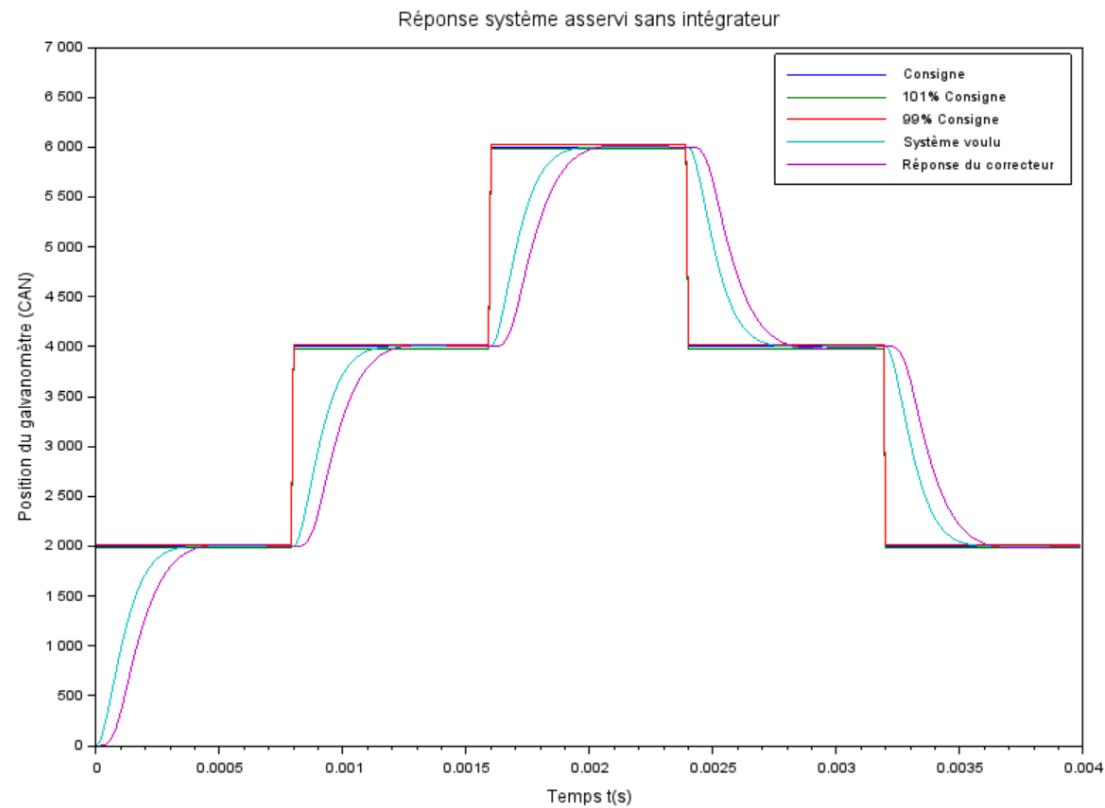
Pics de commande à  $\pm 135\,000$ .

Commande envoyée sur un CNA 16 bits :

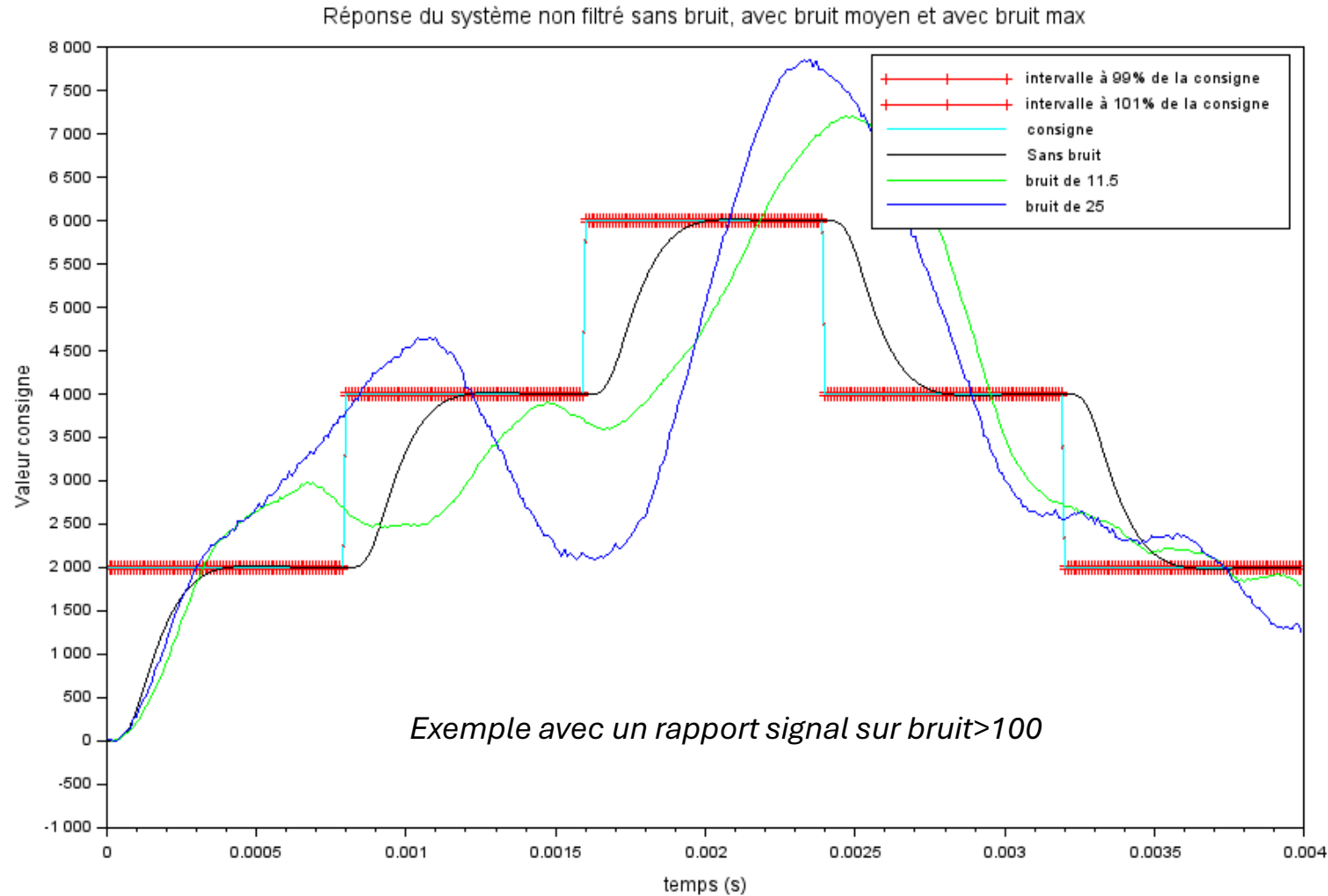
**Valeurs comprises entre  $\pm 32\,768$ .**



# Réponse du correcteur RST avec saturation de la commande

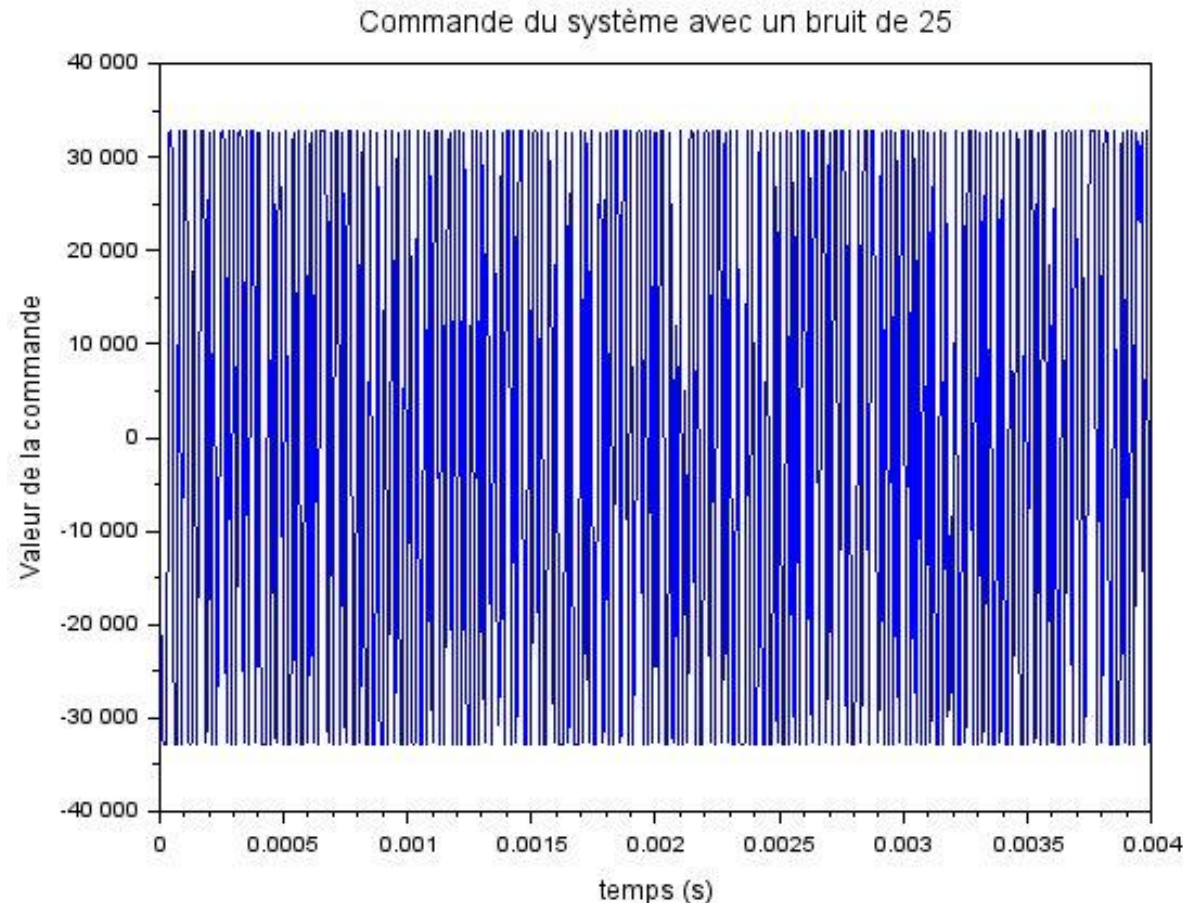


# Réponse du système avec ajout de bruit



**La présence de bruit même infime sur la sortie rend l'asservissement instable.**

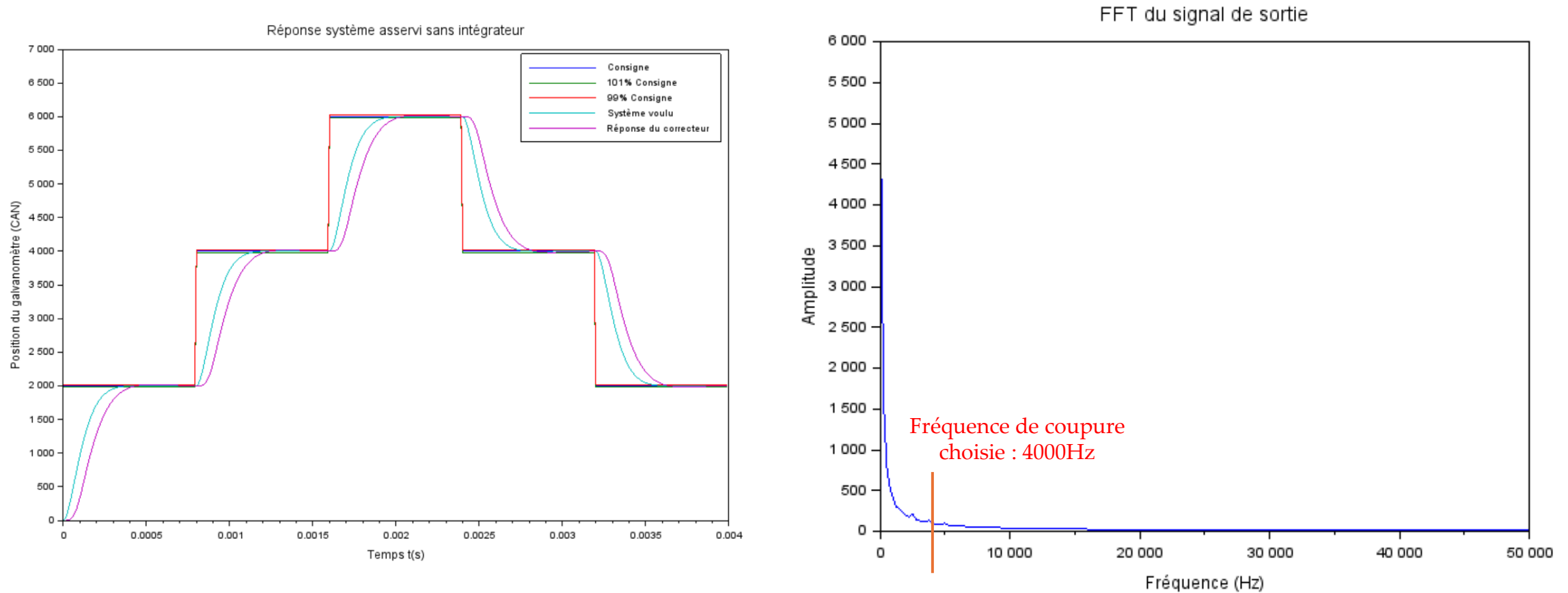
# Commande du système avec ajout de bruit



**Comment diminuer l'impact du bruit sur la commande ?**

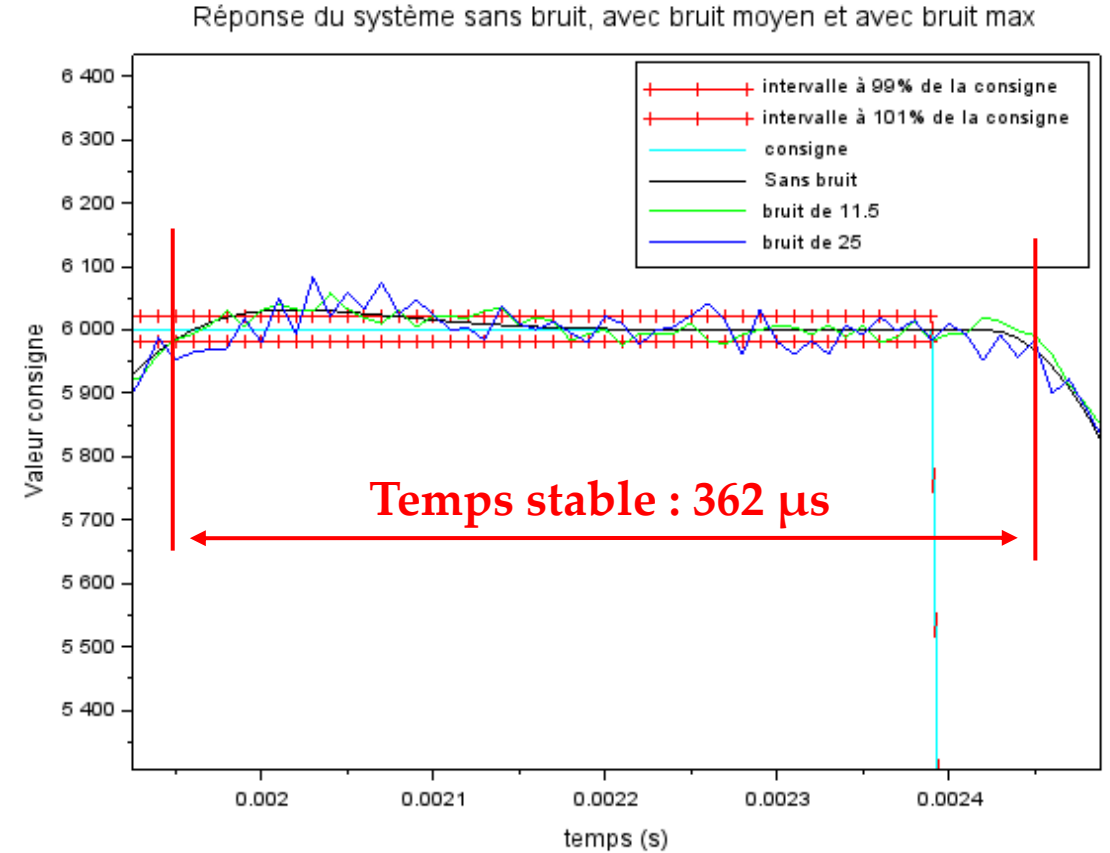
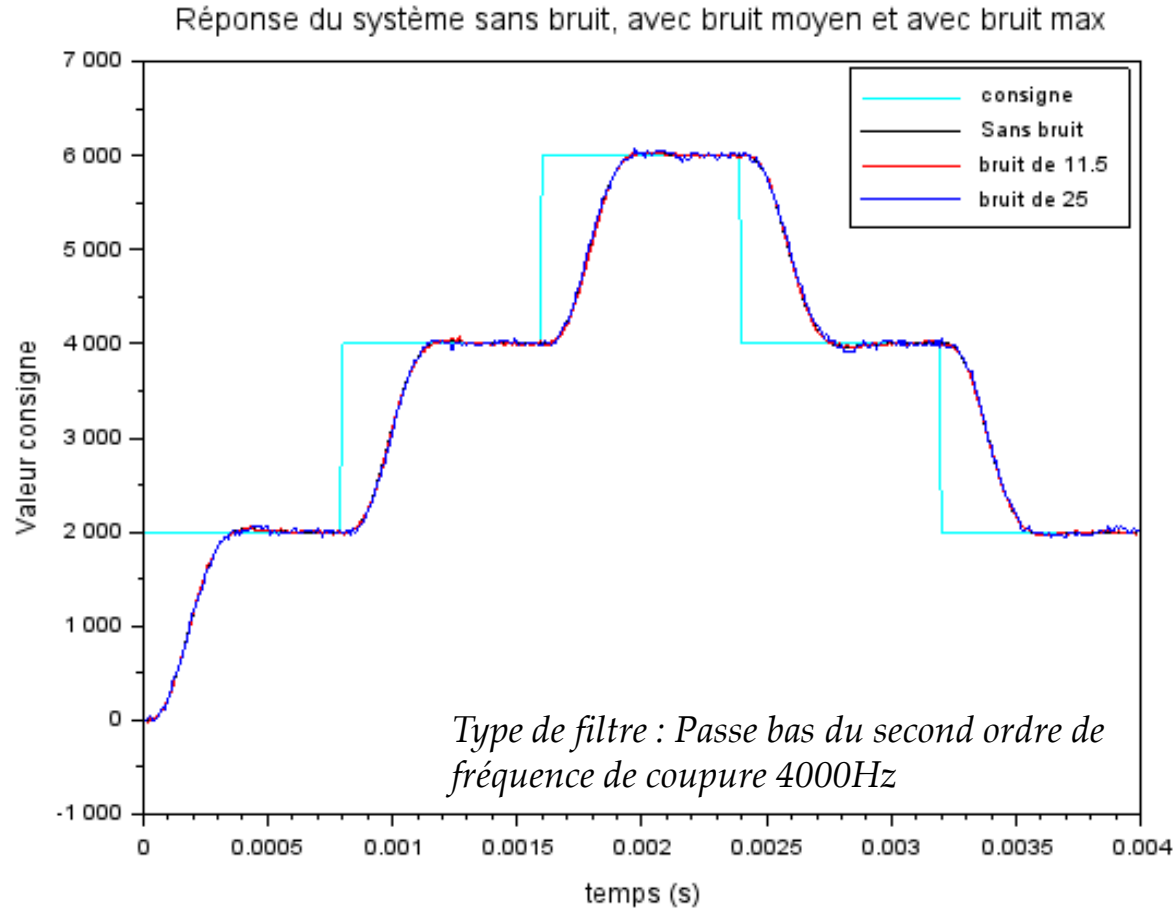
**Possibilité d'ajouter d'un filtre dans le correcteur.**

# Estimation de la fréquence de coupure à partir du signal de sortie espéré sans bruit



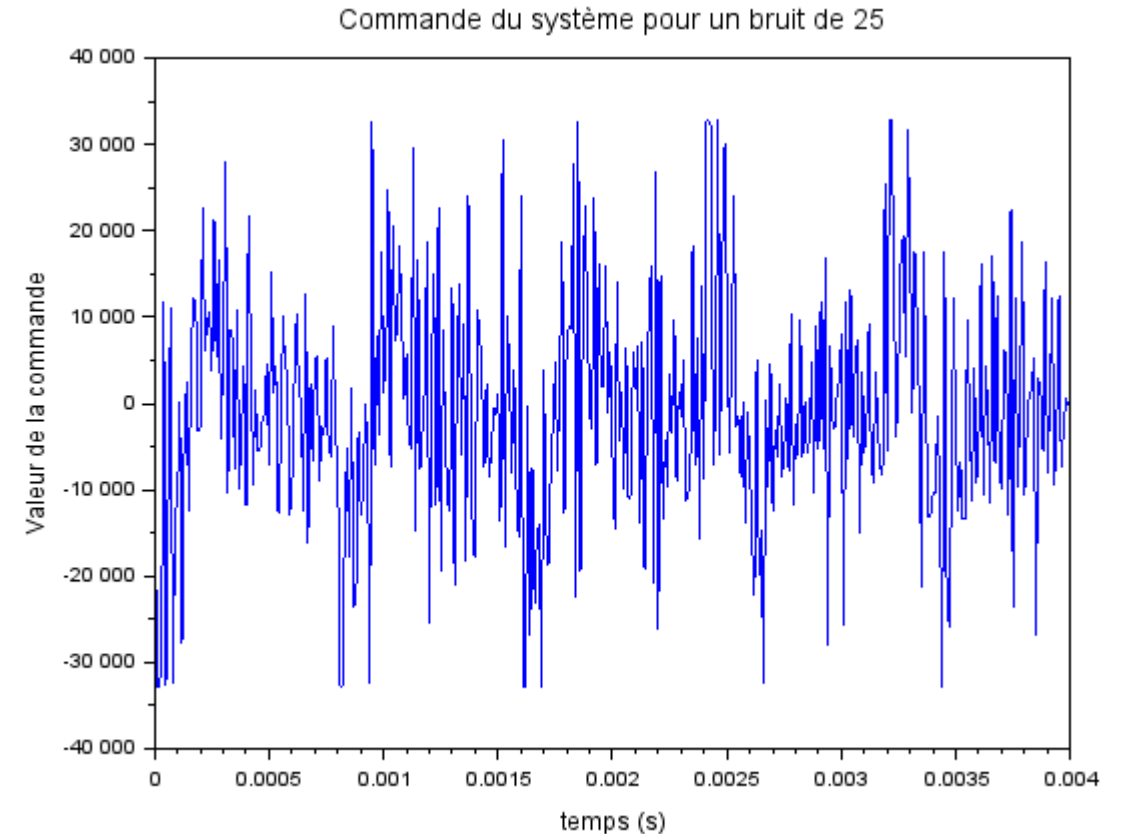
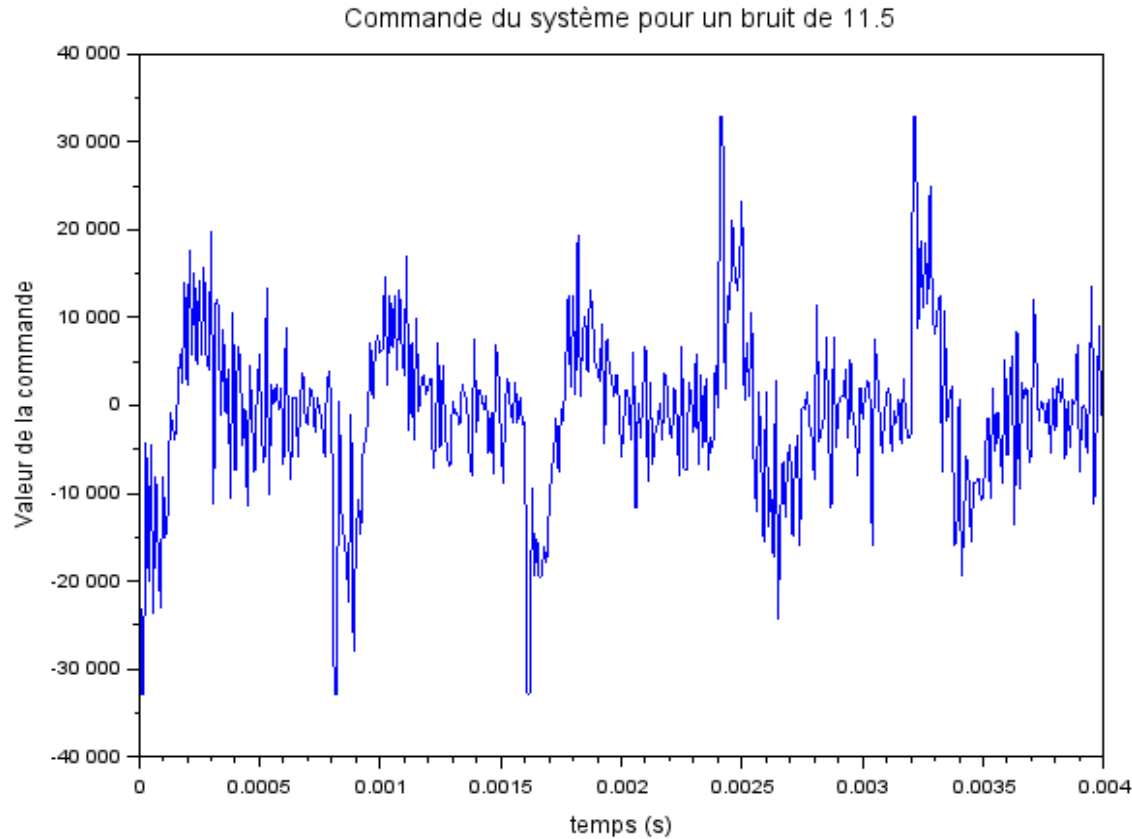
**Ajout d'un filtre passe-bas d'ordre 2 de fréquence de coupure 4000 Hz**

# Réponse du système pour une mesure bruitée filtrée





# Simulation de la commande du système pour une mesure bruitée filtrée



**Le système est bien moins sollicité après ajout du filtre.**

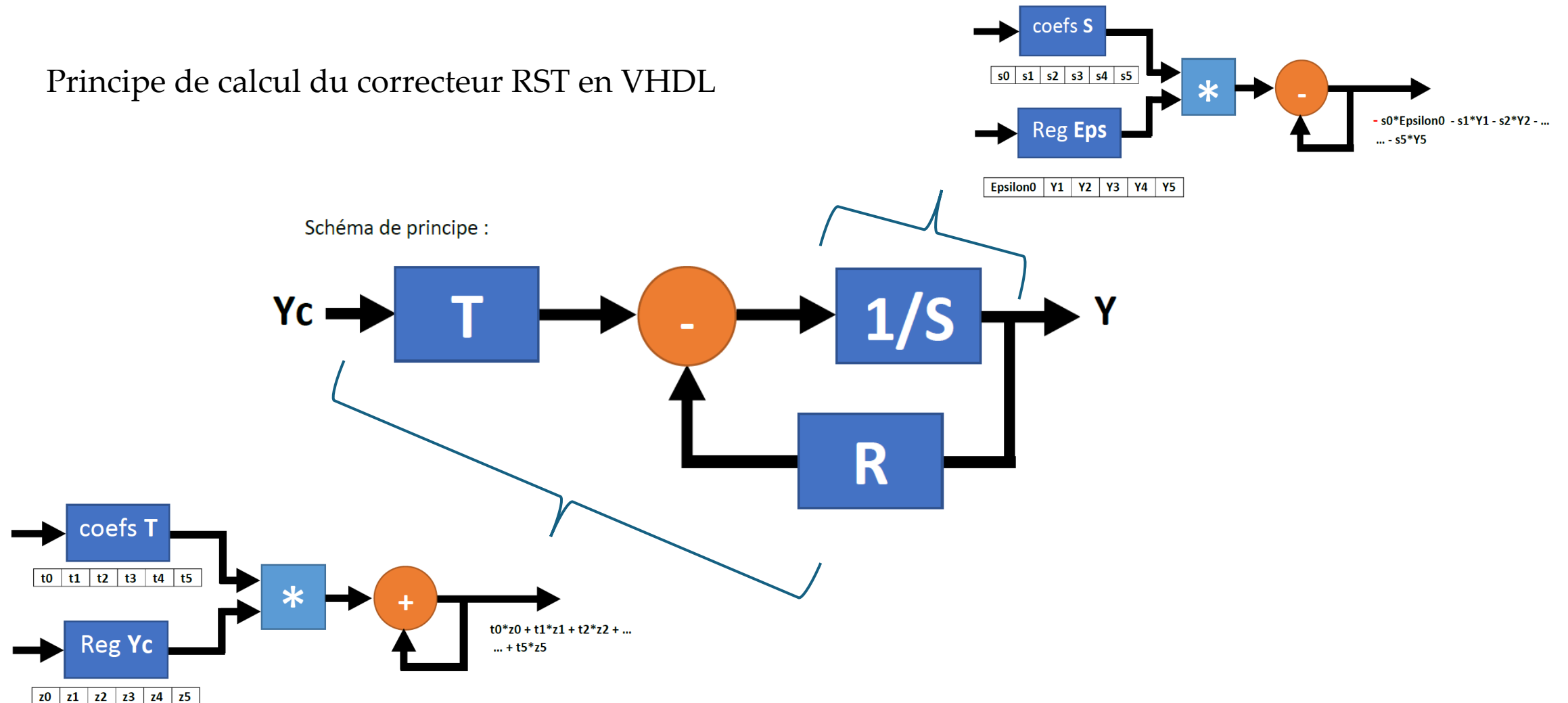
# Résumé de la partie automatique

- Deux méthodes d'identification suivant bien l'allure du système réel mais impossible de certifier qu'elles seront suffisantes pour commander le système.
- Conception d'un correcteur RST par placement de pôles fonctionnel même avec la présence de bruit et la contrainte de saturation de la commande.
- Possibilité de jouer sur la fréquence de commande au cas où l'identification ne permettrait pas d'asservir correctement le système.

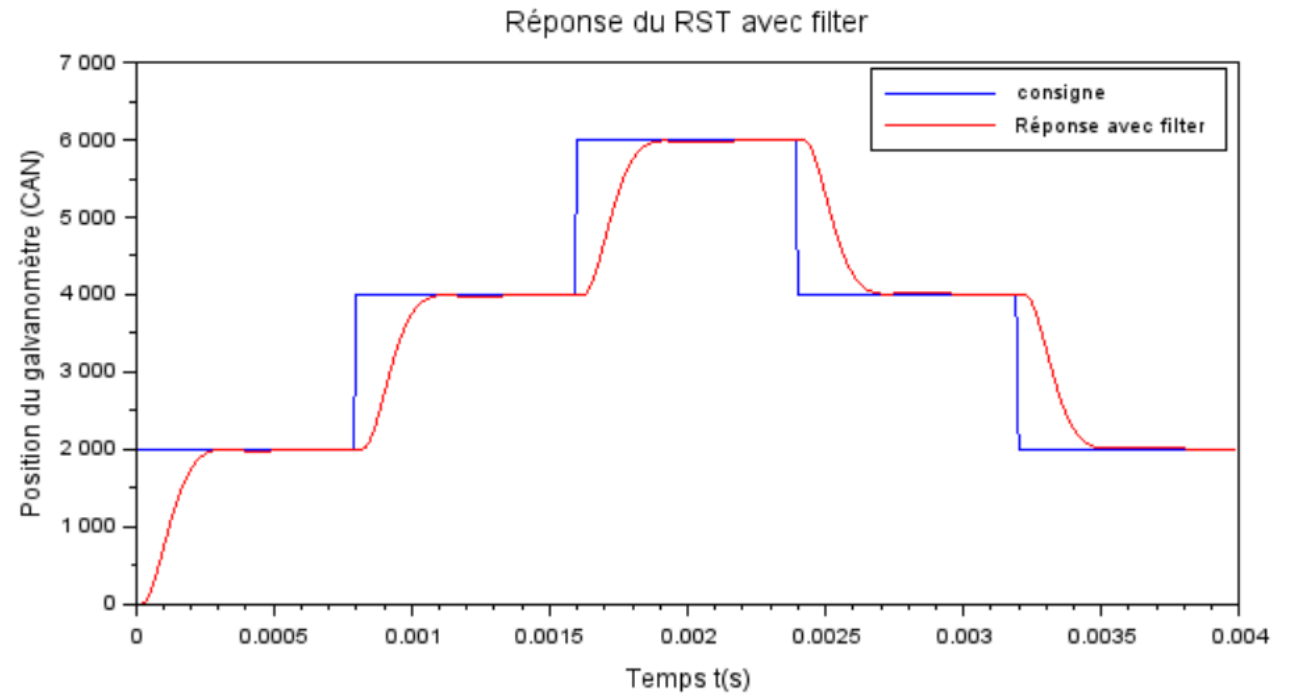
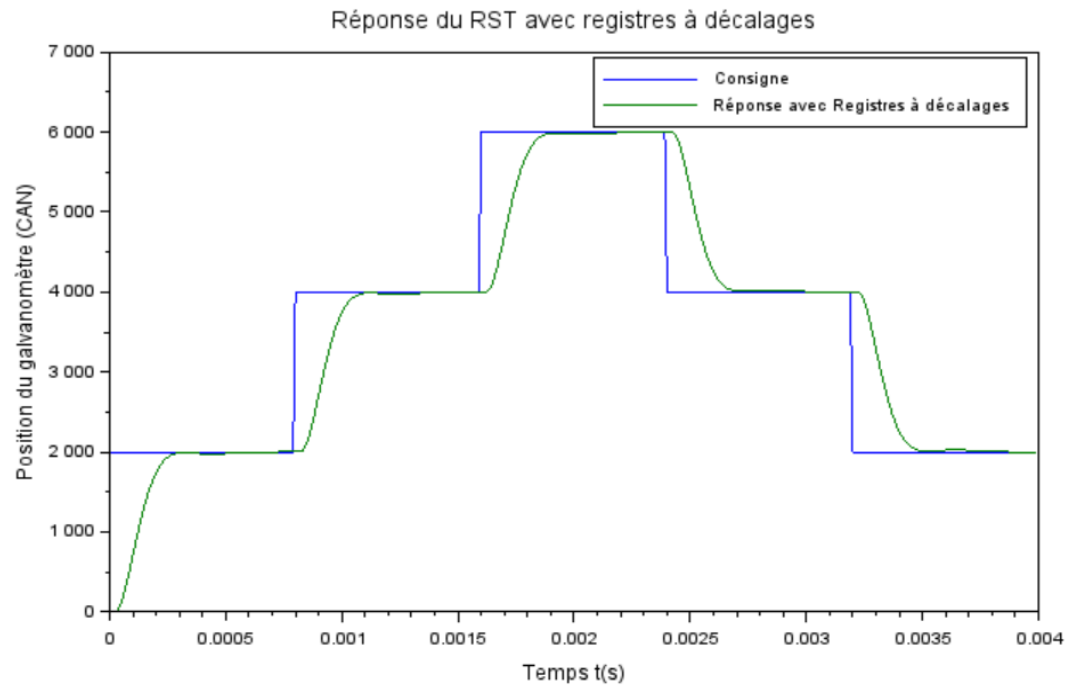
# Conception du correcteur RST en VHDL

# Méthode calcul RST

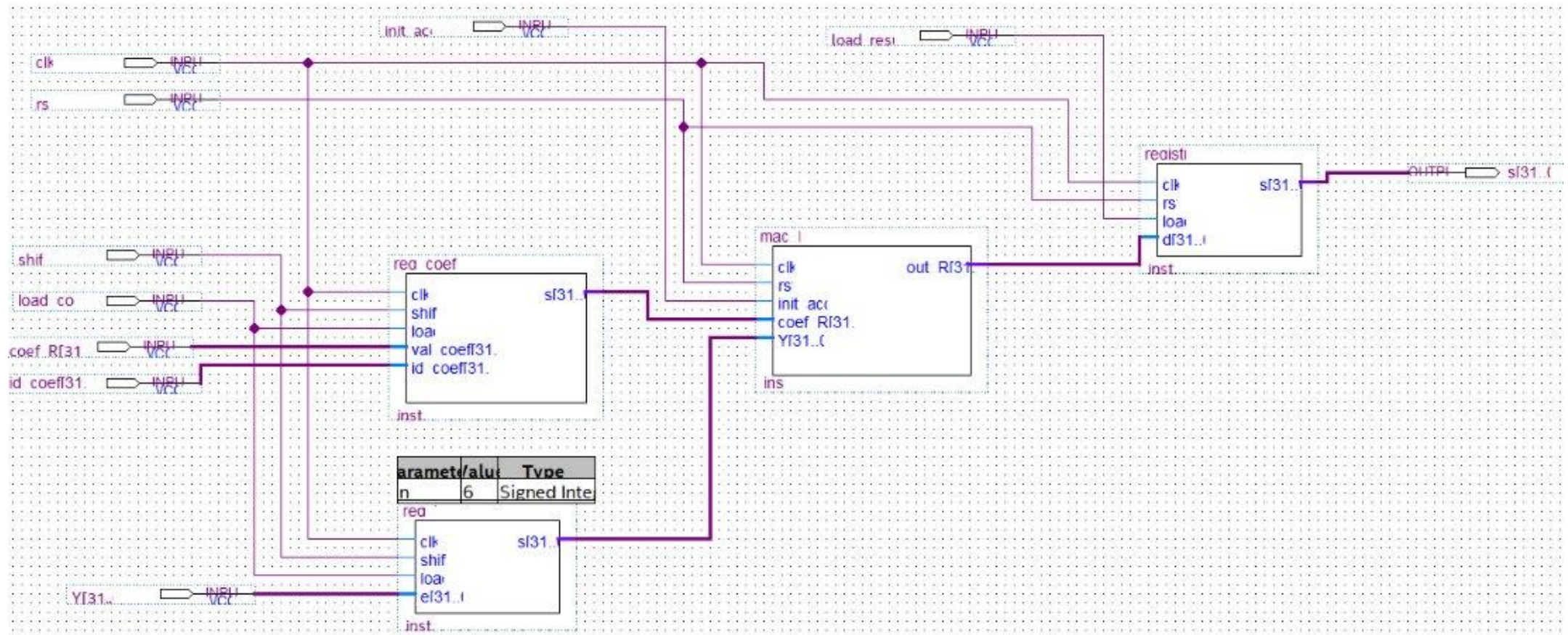
Principe de calcul du correcteur RST en VHDL



# Simulation sur Scilab

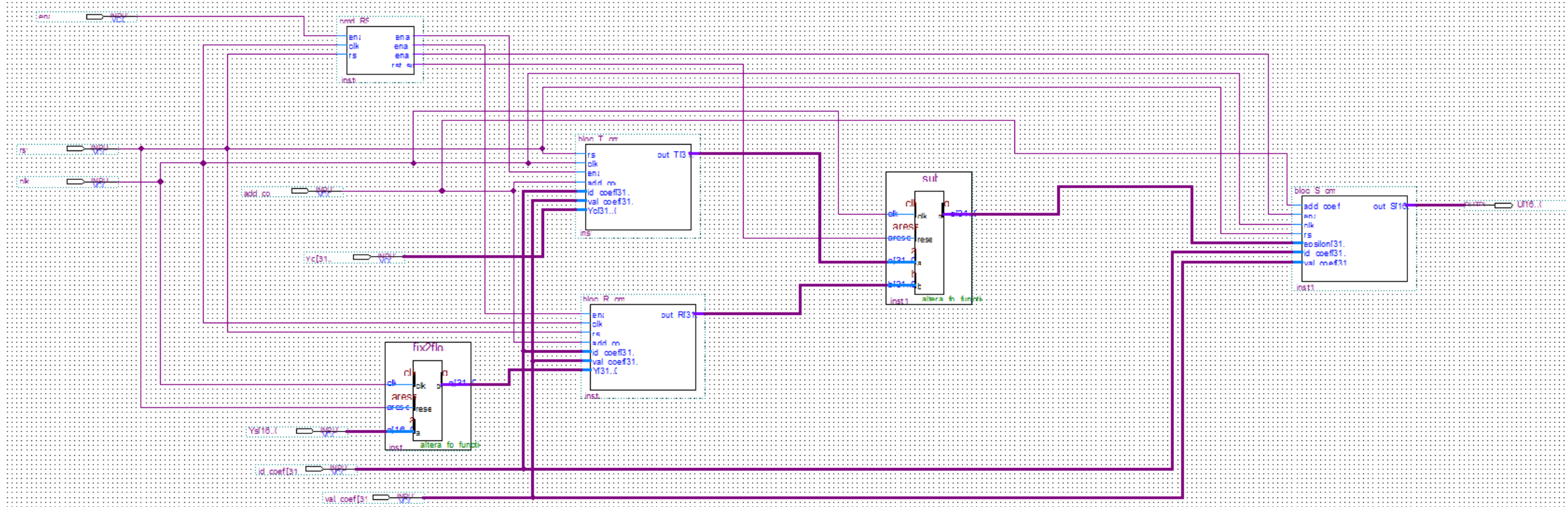


# Création premier bloc



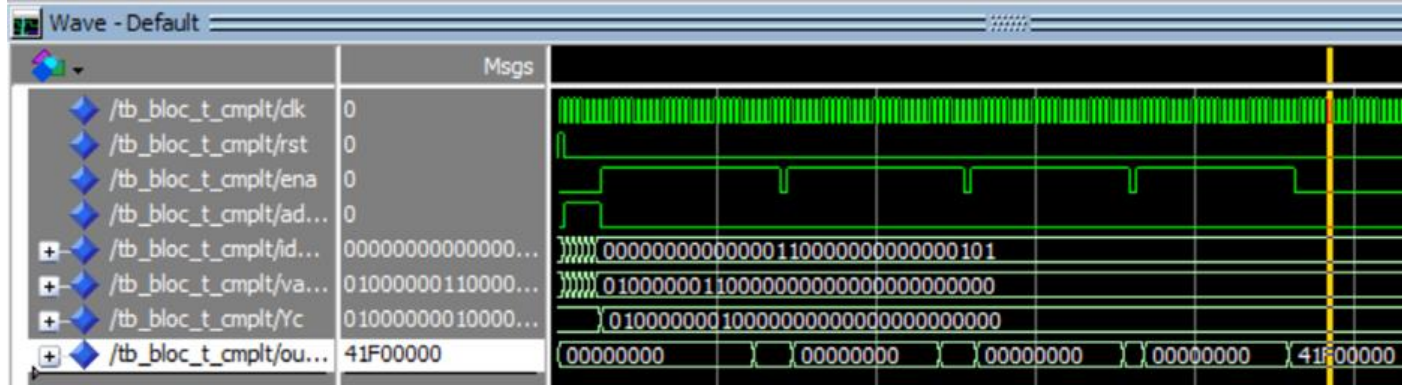
Bloc T sous Quartus

# Schéma du correcteur complet



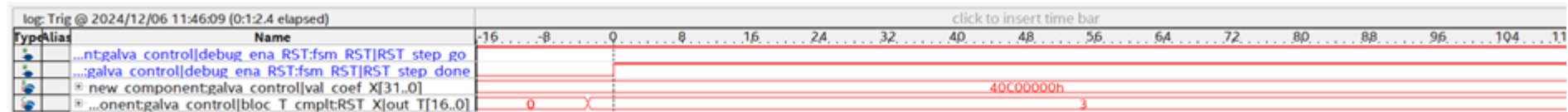
Correcteur RST

# Test du bloc T



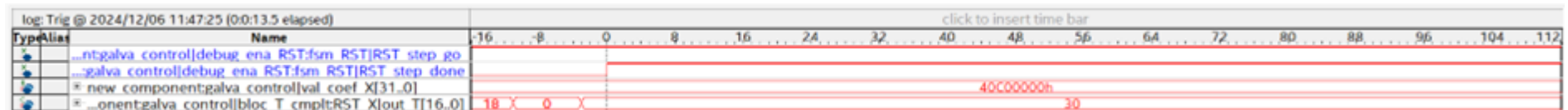
ModelSim (outil de simulation)  
0x41F0000 => 30

1er cycle



SignalTap

Après 4 cycles



Out T = 30



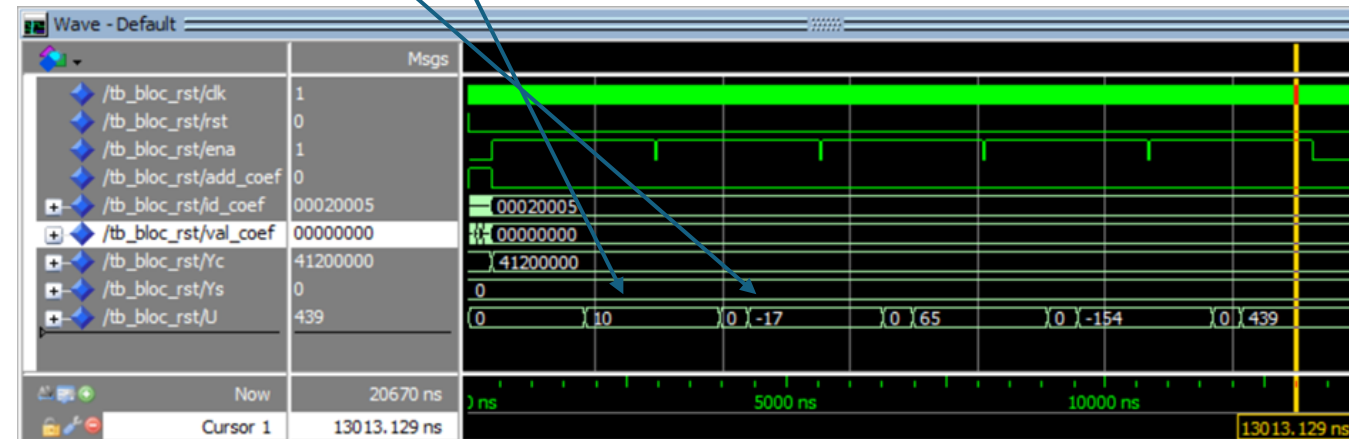
# Simulation du bloc S

yt : 10.000000	ys : 10.000000	yg : 0.000000
yt : 10.000000	ys : -16.946300	yg : 0.000000
yt : 10.000000	ys : 65.090288	yg : 0.000000
yt : 10.000000	ys : -153.848301	yg : 0.000000
yt : 10.000000	ys : 439.283957	yg : 0.000000
yt : 10.000000	ys : -1139.600972	yg : 0.000000
yt : 10.000000	ys : 3071.493800	yg : 0.000000
yt : 10.000000	ys : -8131.849674	yg : 0.000000
yt : 10.000000	ys : 21681.420272	yg : 0.000000
yt : 10.000000	ys : -57625.963765	yg : 0.000000
yt : 10.000000	ys : 153349.314264	yg : 0.000000

Simulation Théorique sur Scilab

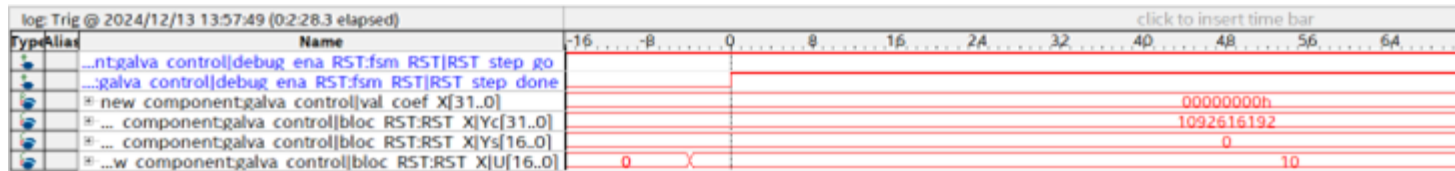
Ys : sortie du correcteur

Yg : Entrée bloc R

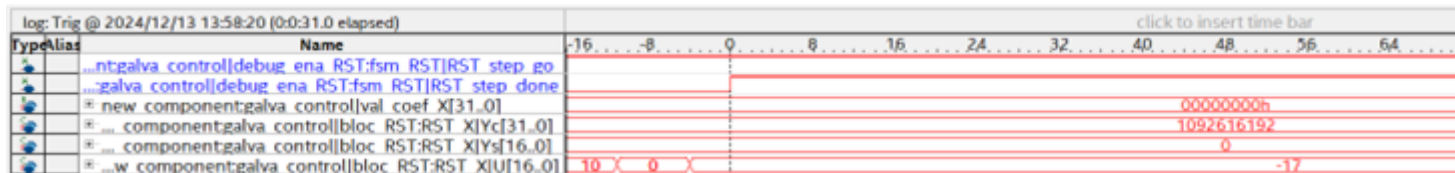


Simulation ModelSim bloc VHDL

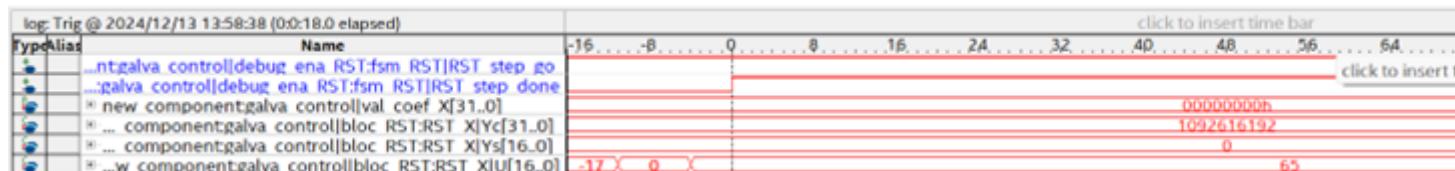
# Test du bloc S



Cycle 1



Cycle 2

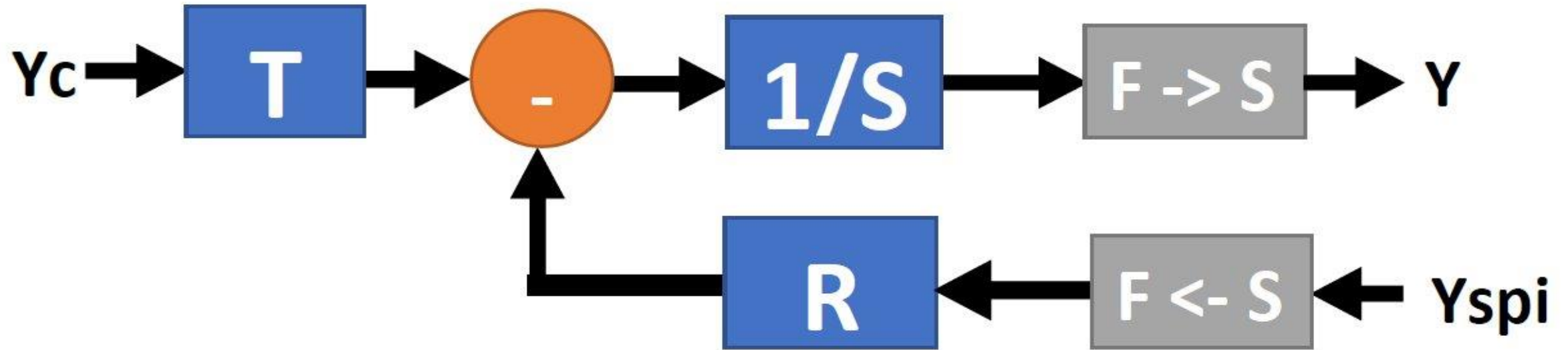


Cycle 3

ys : 10.000000  
ys : -16.946300  
ys : 65.090288  
ys : -153.848301  
ys : 439.283957  
ys : -1139.600972  
ys : 3071.493800  
ys : -8131.849674  
ys : 21681.420272  
ys : -57625.963765  
ys : 153349.314264

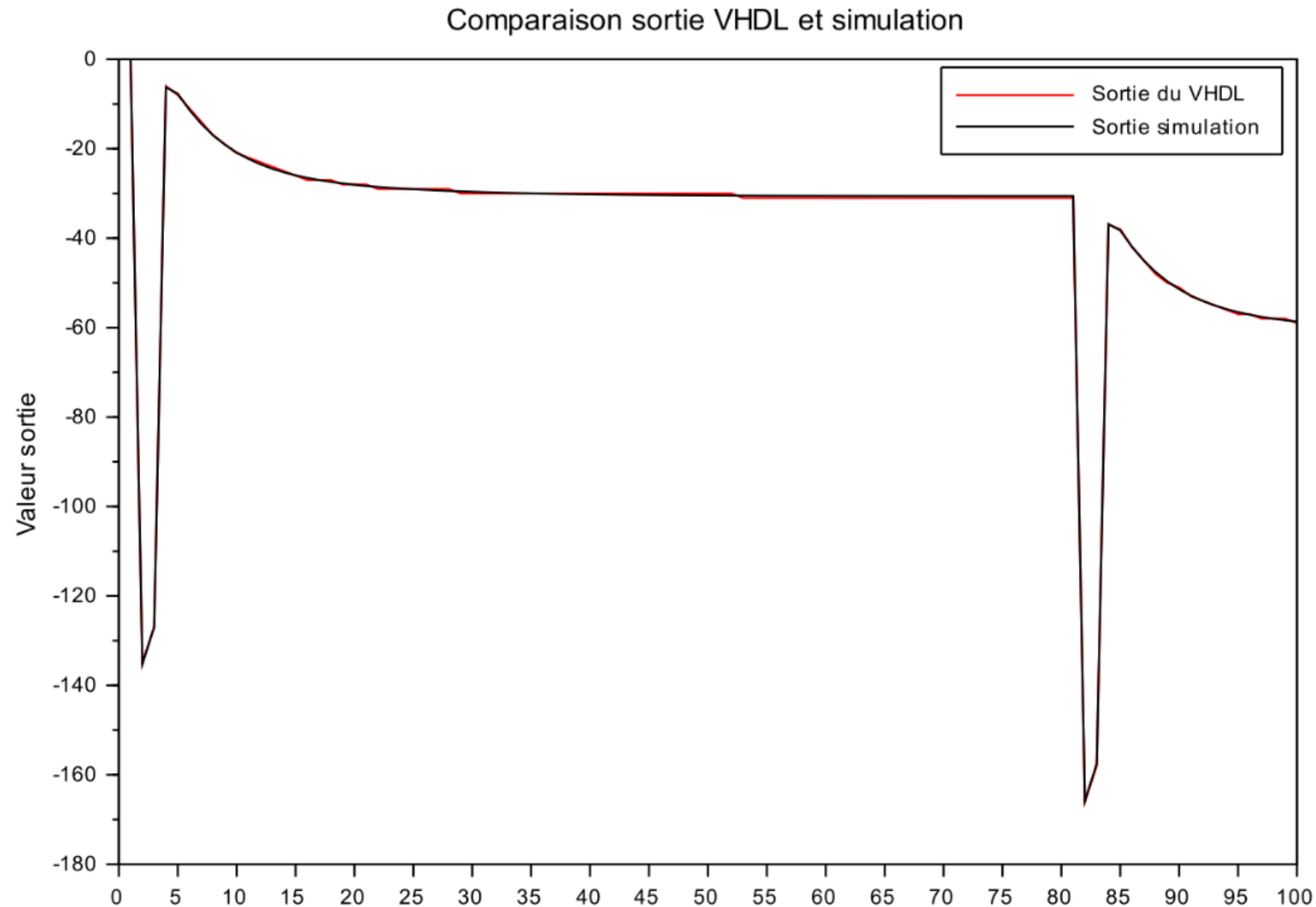
Résultats obtenus sur SignalTap

# Test de l'ensemble avec un retour $Y_{spi}$ imposé



Synoptique test réalisé

# Test de l'ensemble avec un retour Yspi imposé



# Bilan de fin de projet

Travail réalisé	Travail restant
Identification selon deux méthodes (Pseudo-inverse et Variables instrumentales) et conversion des coefficients en continus.	Tester l'aiguillage du correcteur vers le galvanomètre
Conception d'un script permettant de calculer et simuler la correction RST : <ul style="list-style-type: none"><li>• pour n'importe quel système linéaire</li><li>• avec inexactitude de l'identification</li><li>• avec saturation de la commande</li><li>• Avec du bruit sur la sortie</li></ul>	Dupliquer le correcteur pour calculer les commandes des deux voies en parallèle
Création du correcteur RST en VHDL	Créer le script de commande pour afficher la matrice
Tests blocs R, S et T séparément	
Test du correcteur RST complet	

# Perspectives d'amélioration

- Basculer les codes Scilab en C pour pouvoir les implémenter directement sur la carte et ne plus avoir besoin d'un ordinateur.
- Augmenter la plage de tension de la carte ou prendre des galvanomètres plus rapides pour assurer le bon fonctionnement de l'asservissement sur de plus grands niveaux de bruit.

Merci pour votre attention !