# M2 EEA SME

Janvier 2020

---

# EIEAS3FM
# Architecture de l'électronique et conception conjointe

---

**Auteur :**
Nicolas OTAL
Antoine ROUTIER

**Encadrants :**
M. JAMMES

# Introduction

*Dans le cadre de notre UE "Architecture de l'électronique et conception conjointe", nous avons réalisés différentes séances de travaux pratiques pour étudier différents aspects du VHDL-AMS. Pour cela nous avons mis en œuvre les différentes compétences acquises lors de nos cours magistraux et travaux dirigés.*

*L'objectif de ces TP est d'élaborer et de simuler un modèle simple du système de freinage d'un véhicule, avec le logiciel AdvanceMS de Mentor-Graphics, puis de compléter ce modèle en introduisant un système d'antiblocage (ABS). Un premier aspect du travail fut l'analyse et la simulation d'un système de freinage dépourvue d'ABS. Un deuxième aspect a été l'ajout de ce dernier à notre système pour en réaliser une analyse et simulation complète.*

# Sigles et acronymes

| | |
|---|---|
| **VHDL** | *VHSIC1 Hardware Description Language* |
| **VHSIC** | *Very High Speed Integrated Circuit* |
| **AMS** | *Analog and Mixed-Signal* |

# Table des matières

# 1 Simulation d'un système de freinage sans ABS

Durant la pratique des travaux sur VHDL-AMS il est demandé aux étudiants de réaliser et d'instancier étapes par étapes les différentes partie d'un système de freinage d'une voiture. Pour la réalisation du travail nous avons travailler avec le logiciel ModelSim et un éditeur de texte pour venir rédiger, en VHDL, les différentes instances du véhicule jusqu'à l'intégration de l'ABS.

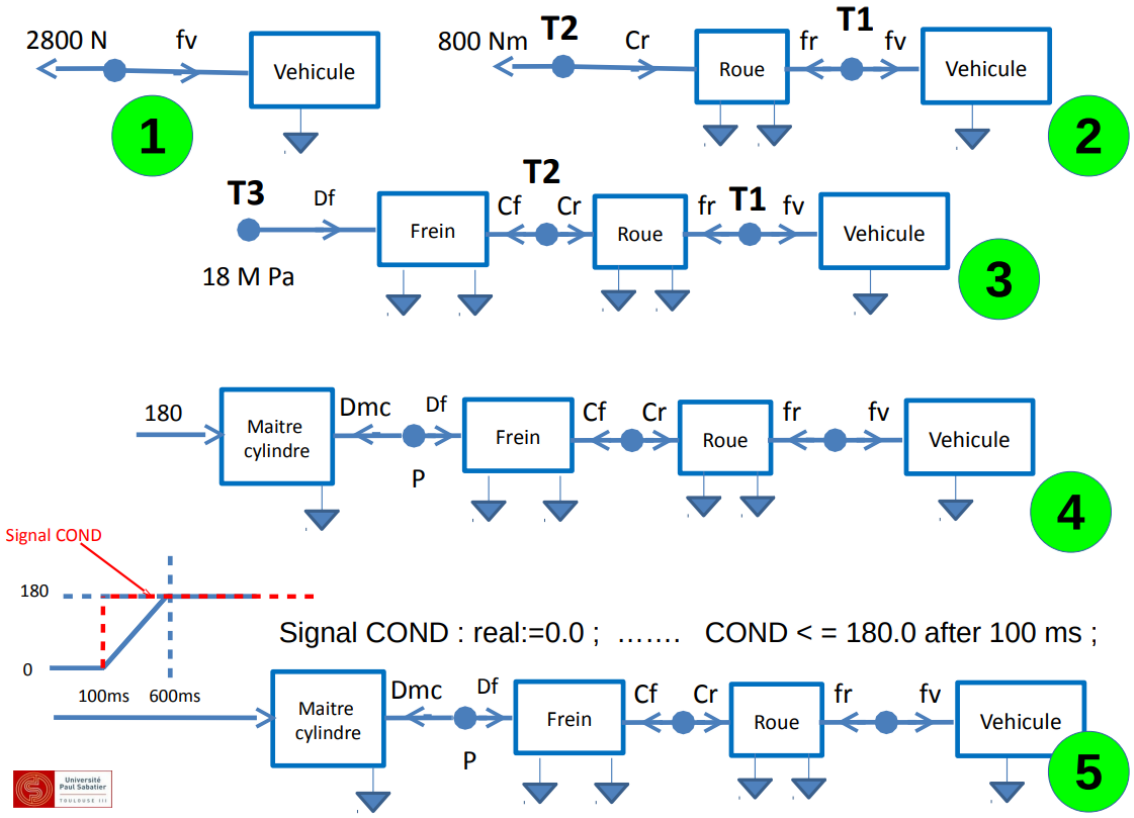


FIGURE 1 – Implémentation des différentes parties du système de freinage d'une voiture

Il est demandé d'instancier étapes par étapes différents blocs avec leurs terminaux coresspondants, comme indiqué dans le powerpoint de présentation du TP.

## 1.1 Analyse instanciation véhicule

Dans cette première partie nous devons instancié le véhicule fournit dans les fichiers du projet. Sur la Figure 2 on peut voir le schéma bloc qu'il faut instancier et décrire par son entitée en VHDL-AMS.

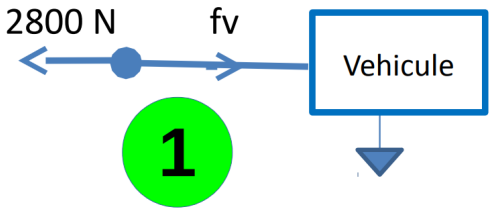


FIGURE 2 – Instanciation du véhicule avec son terminal

Dans l'entitée du véhicule nous avons donc à définir certaines quantités qui va nous permettra de résoudre certaines équations. Les paramètres d'entrées de notre première équation sont les suivants :

— $m \rightarrow$ Masse véhicule.
— $Cx \rightarrow$ Coéfficient aérodynamique.
— $S \rightarrow$ Surface Frontale.
— $V\_init \rightarrow$ Vitesse initiale du vehicule.

```
15  ----------------------------------------------------------------------------------------
16  -- Test véhicule seul
17  ----------------------------------------------------------------------------------------
18  architecture A of test is
19
20  terminal t1 : translational_velocity;
21  quantity force through t1;
22
23  begin
24
25  force==2800.0;
26  Auto:   entity vehicule(one)
27  generic map
28  (
29      m=>m_veh, cx=>0.3,
30      S=>1.8, v_init=>28.0
31  )
32  port map(roue=>t1);
33  end A;
34
```

FIGURE 3 – Code VHDL Architecture A - Véhicule

Pour la première partie, véhicule il faudra mettre en place un terminal T1 ayant pour type "Translational_velocity". Celui-ci sera plus tard rélié au Terminal "Roue", du même type, pour nous permettre la visualisation de la vitesse du véhicule.

Avec l'aide du logiciel ModelSim nous pouvons simuler le temps de freinage nécessaire à l'arrêt du véhicule lorsqu'on lui applique une force de 2800 N au terminal T1.

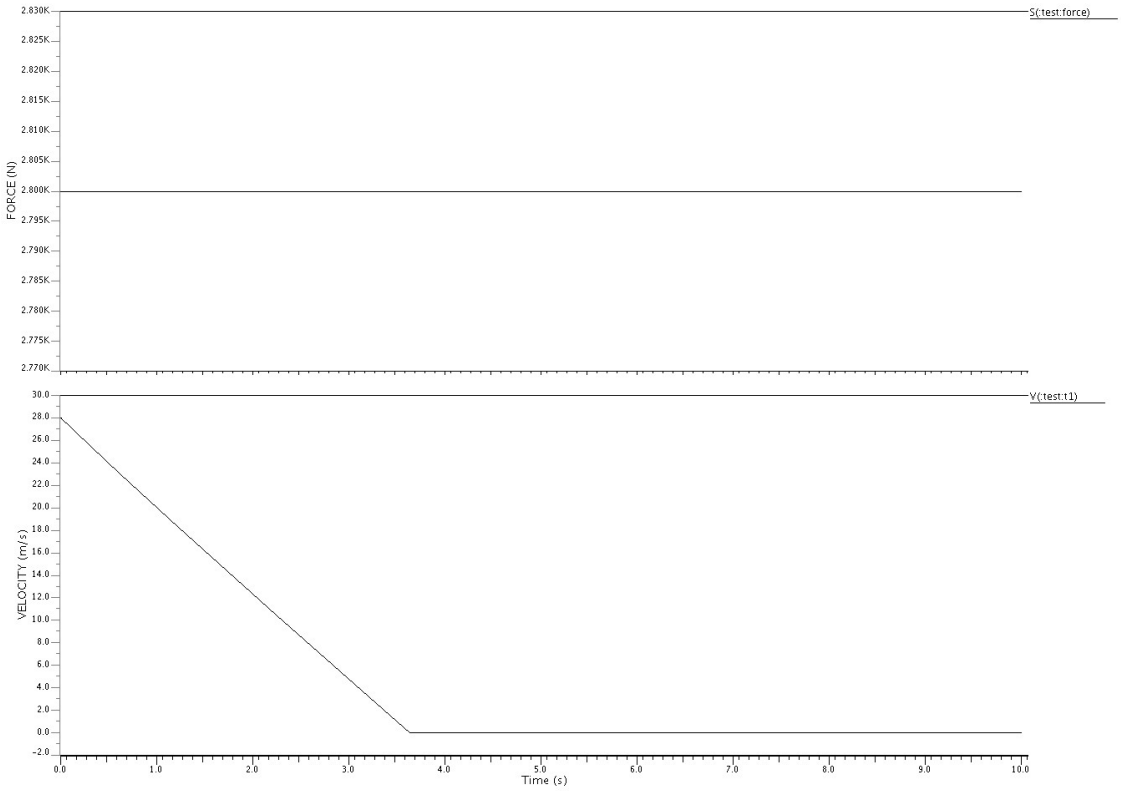


FIGURE 4 – Simulation Terminal T1 Force et Temps pour le freinage

## 1.2 Analyse instanciation roue

Dans cette seconde partie nous devons instancier la roue fournit dans les fichiers du projet. Sur la Figure 5 on peut voir le schéma bloc qu'il faut instancier et décrire par la suite son entitée en VHDL-AMS.

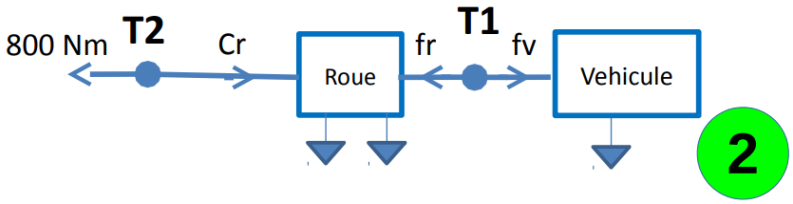


FIGURE 5 – Instanciation du véhicule avec son terminal

Une fois le véhicule instancié on nous demande d'ajouter le système "Roue". Dans cette partie on déclare en code la partie générique qui va nous permettre de résoudre l'équation suivante ayant pour entrées ces paramètres :

— $route \rightarrow$ Etat de la route (mouillée, seche).
— $m \rightarrow$ Masse vehicule.
— $rR \rightarrow$ Rayon de la roue.
— $IR \rightarrow$ Inertie de la roue.
— $mu0_D \rightarrow$ Coéfficient adhérence sans glissement sur sol sec.
— $As \rightarrow$ Facteur de decroissance de l'adherence.
— $mu0_W \rightarrow$ Coéfficient adhérence sans glissement sur sol mouillé.
— $Vc \rightarrow$ Caracteristique de la surface.
Dans cette partie du code il y aura deux terminaux à instancier au niveau de la roue :

— $veh \rightarrow$ De type "translational_velocity".
— $frein \rightarrow$ De type "rotational_velocity".

```
56  ----------------------------------------------------------------------------
57  -- Test véhicule + roue
58  ----------------------------------------------------------------------------
59  architecture C of test is
60
61  terminal t1 : translational_velocity;
62  terminal t2 : rotational_velocity;
63  quantity couple through t2;
64
65  begin
66
67  couple==800.0;
68
69  Auto:  entity vehicule(one)
70  generic map
71      (
72          m=>m_veh,cx=>0.3,
73          S=>1.8,v_init=>28.0
74      )
75      port map(roue=>t1);
76      URoue: entity roue(A)
77      generic map
78          (
79              route=>seche, m=>m_veh,
80              rR=>0.275, IR=>0.4, mu0_D=>1.0,
81              As=>0.01, mu0_W=>0.5, Vc=>27.8
82          )
83  port map(veh=>t1,frein=>t2);
84  end C;
```

FIGURE 6 – Code VHDL Architecture C - Roue

Une fois la "Roue" implémentée on se retrouve sur ModelSim avec deux Terminaux, T2 étant le dernier terminal instancié qui représente le terminal d'entrée de "Roue" et T1 son Terminal de sortie. On observe grâce à ModelSim la vitesse rotationnelle de "Roue" ainsi que la vitesse du véhicule. On constate que la vitesse de "Roue" chute brusquement avant de réduire linéairement comme pour la vitesse du véhicule.

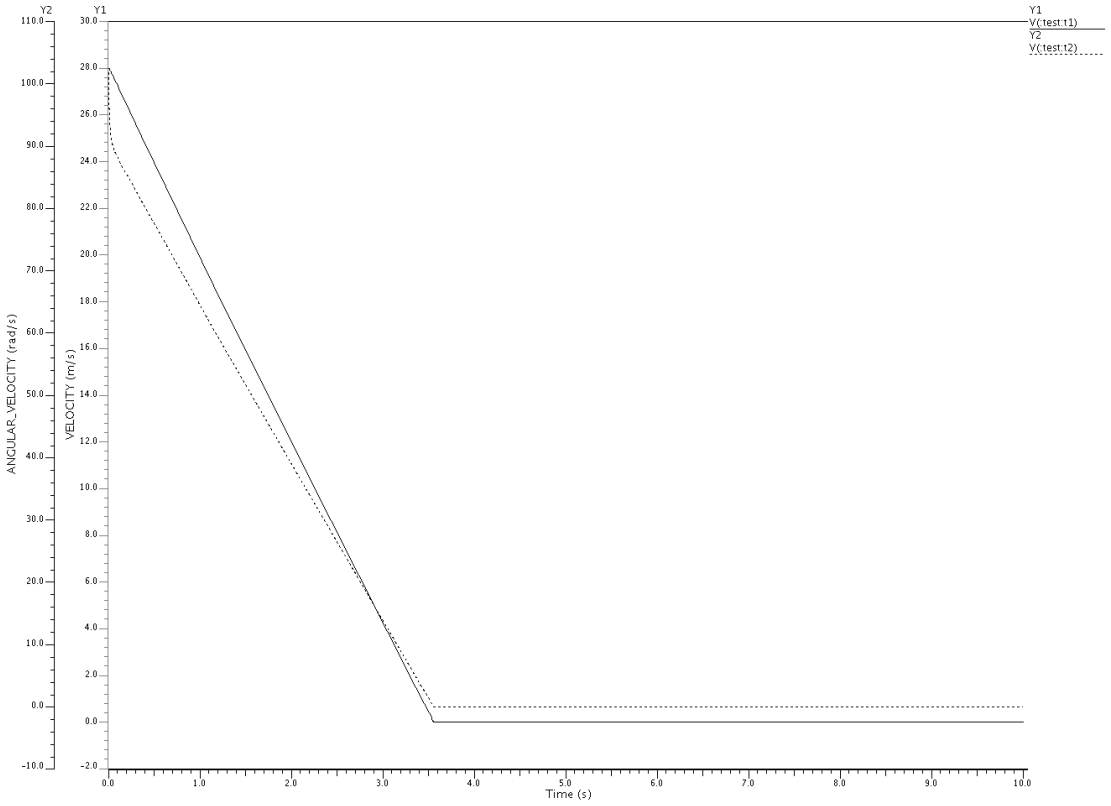


FIGURE 7 – Simulation Terminal T2 et T1

## 1.3 Analyse instanciation frein

Dans la troisème partie du tp nous devons instancier le frein au terminal T2, fournit dans les fichiers du projet. Sur la Figure 8 on peut voir le schéma bloc qu'il faut instancier et décrire par la suite son entitée en VHDL-AMS.

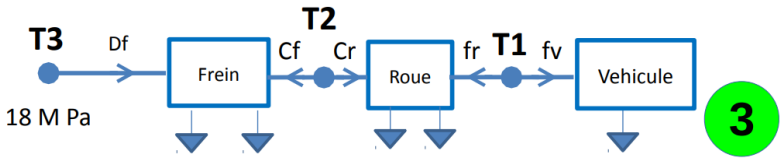


FIGURE 8 – Instanciation du véhicule avec son terminal

Une fois le véhicule et la Roue instanciés on nous demande d'ajouter le système "Frein". Dans cette partie on déclare en code la partie générique suivante :

— $coef_fric \rightarrow$ Coéfficient de friction des plaquettes.
— $S \rightarrow$ Surface piston = 10 cm2.
— $R \rightarrow$ Rayon moyen du disque.

Dans cette partie du code il y aura deux terminaux à instancier au niveau de la du Frein :

— $Roue \rightarrow$ Type "rotational_velocity".
— $MC \rightarrow$ Type "fluidic" à l'entrée du Frein.

Dans cette nouvelle instance nous venons d'implémenter le "Frein" permettant l'obtention de la vitesse de rotation du disque dans le terminal 2 en fonction de la pression appliquée à son entrée. On définit une quantité de Pression (across) et un débit (Through) pour le nouveau Terminal. On vient ensuite fixer la valeur de pression à 18 MPa pour pouvoir résoudre l'équation.

```
85  ----------------------------------------------------------------------------
86  -- Test véhicule + roue + frein
87  ----------------------------------------------------------------------------
88  architecture D of test is
89
90  terminal t1 : translational_velocity;
91  terminal t2 : rotational_velocity;
92  terminal t3 : fluidic;
93  quantity pression across debit through t3;
94  begin
95
96  pression == 18000000.0;
97
98  Auto:  entity vehicule(one)
99  generic map (m=>m_veh,cx=>0.3,S=>1.8,v_init=>28.0)
100     port map(roue=>t1);
101 URoue: entity roue(A)
102 generic map
103 (
104     route=>seche, m=>m_veh,
105     rR=>0.275, IR=>0.4, mu0_D=>1.0,
106     As=>0.01, mu0_W=>0.5, Vc=>27.8
107 )
108 port map(veh=>t1,frein=>t2);
109 Ufrein: entity frein(one)
110 generic map
111 (
112     coef_fric=>0.36, S=>1.0e-3, R=>0.12
113 )
114     port map(roue=>t2, MC=>t3);
115 end D;
```

FIGURE 9 – Code VHDL Architecture D - Roue et Frein

Sur la figure 10 on peut observer grâce à la simulation que lorsque l'on applique une pression de 18 MPa au frein nous avons en sortie la vitesse qui décroit exactement comme à l'instanciation du véhicule ce qui nous permet de conclure que le frein est implémenté de la bonne manière.

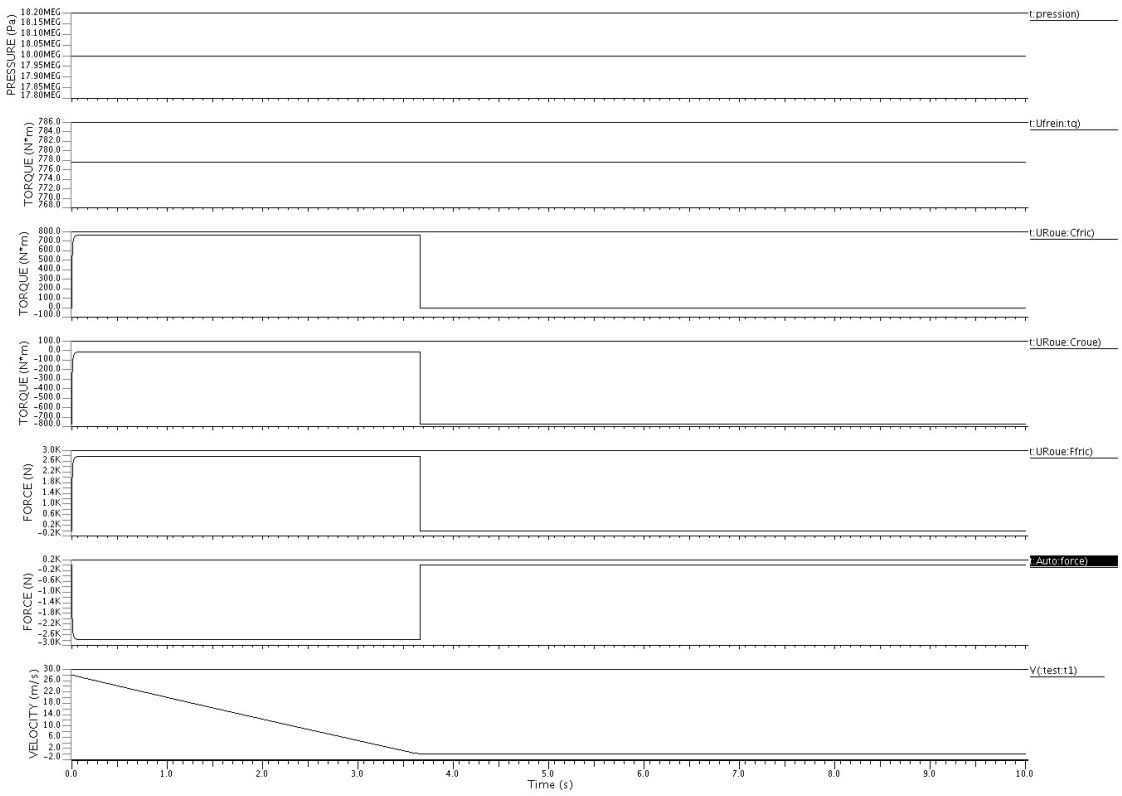


FIGURE 10 – Simulation Terminaux T3 T2 et T1

## 1.4 Analyse instanciation maître cylindre

Dans la quatrième partie nous allons instancier le maître cylindre qui va permettre de transmettre la pression vers le terminal T3. Sur la Figure 11 on peut voir le schéma bloc qu'il faut instancier et décrire par la suite son entitée en VHDL-AMS.

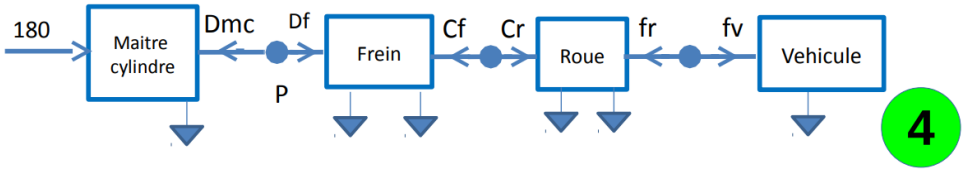


FIGURE 11 – Instanciation du véhicule avec son terminal

Pour l'instanciation et la résolution du système maître cylindre il sera nécessaire de compléter et de modifier le fichier fournit en TP. Dans cette partie on déclare en code la partie générique suivante :

— $S \rightarrow$ Coéfficient de friction des plaquettes.
— $S \rightarrow$ Surface piston = 10 cm2.

Dans cette partie du code il y aura deux terminaux à instancier au niveau de la du Frein :

— $Hyd \rightarrow$ Type "fluidic" qui représente la pression en entrée du frein.
— $Force \rightarrow$ Qui est une quantité réel exercée par le conducteur sur la pédale de frein.

```
117  ------------------------------------------------------------------------------------------------
118  -- Test véhicule + roue + frein + MC
119  ------------------------------------------------------------------------------------------------
120  architecture E of test is
121
122  terminal t1 : translational_velocity;
123  terminal t2 : rotational_velocity;
124  terminal t3 : fluidic;
125  quantity s_force : real;
126  signal COND : real:=0.0;
127
128  begin
129
130  COND <=180.0 after 100 ms;
131  s_force==cond'ramp(0.5);
132
133  Auto:  entity vehicule(one) generic map (m=>m_veh,cx=>0.3,S=>1.8,v_init=>28.0)
134      port map(roue=>t1);
135  URoue: entity roue(A) generic map(route=>seche, m=>m_veh, rR=>0.275, IR=>0.4, mu0_D=>1.0,
136   As=>0.01, mu0_W=>0.5, Vc=>27.8)
137      port map(veh=>t1,frein=>t2);
138  Ufrein: entity frein(one) generic map (coef_fric=>0.36, S=>1.0e-3, R=>0.12)
139      port map(roue=>t2, MC=>t3);
140  UMc: entity maitre_cylindre(one) generic map (S=>1.0e-4, coef_assistance=>10.0)
141      port map(hyd=>t3, force=>s_force);
142  end E;
```

FIGURE 12 – Code VHDL Architecture E - Maître cylindre

Dans cette nouvelle entité, maître cylindre, elle nous permet l'injection de la pression dans le frein en fonction de la force exercée par le conducteur sur la pédale de frein. Dnas ce nouveau cas Force étant un réel il est nécessaire de déclarer une quantité de type réel contrairement aux précédentes déclaration des Terminaux ultérieurs.

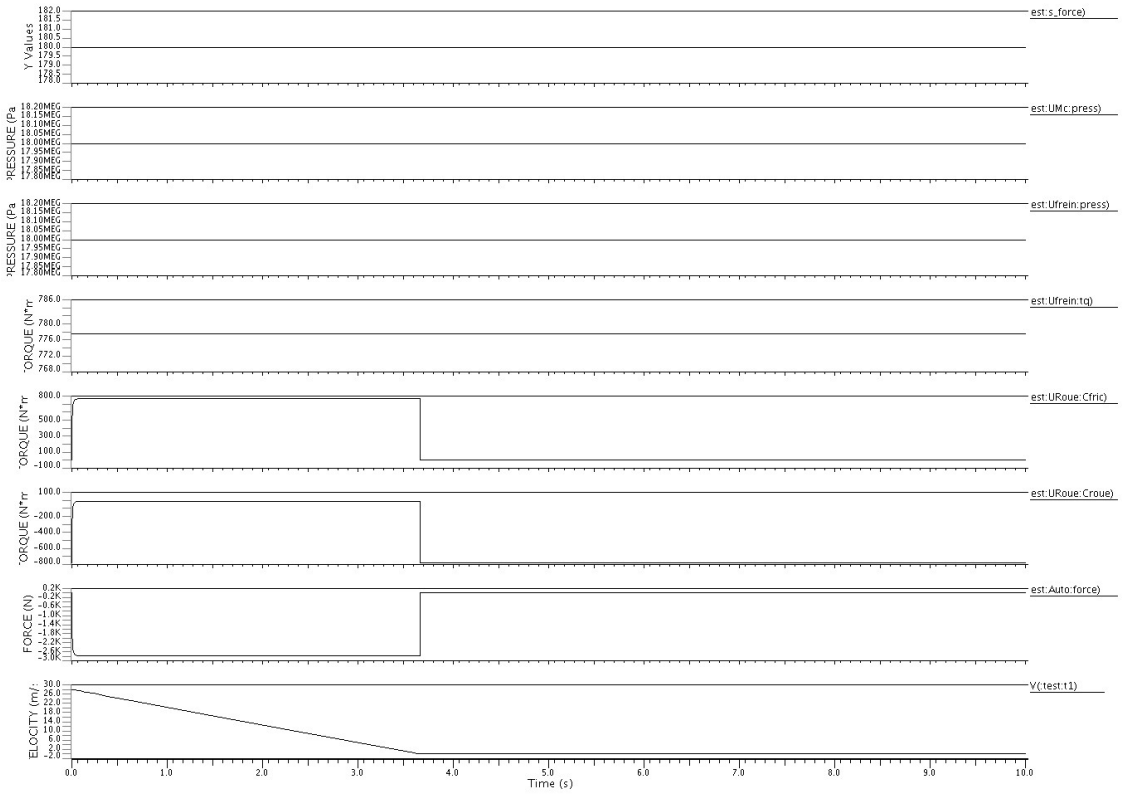


FIGURE 13 – Simulation du Terminal 4 - Maître cylindre

On constate avec l'outil de modélisation que lorsque nous appliquons une préssion de 180 N sur la pédale de frein nous obtenons un même temps de freinage et une même pression sur le frein que sur l'Implémentation précédente. Cette simulation nous permet de montrer la bonne implémentation de la partie correspondate au maître cylindre.

Pour la suite de cette partie on nous demande d'intégrer un temps de réaction du conducteur à notre entrée, permettant d'obtenir une simulation plus réaliste du système. Pour cela nous ajoutons un signal de type réel qui prendra la valeur de 180 N après 100 ms et par la suite affecter à ce signal une quatité de "Force" qui évoluera linéairement de 0 à 180 durant 500 ms.

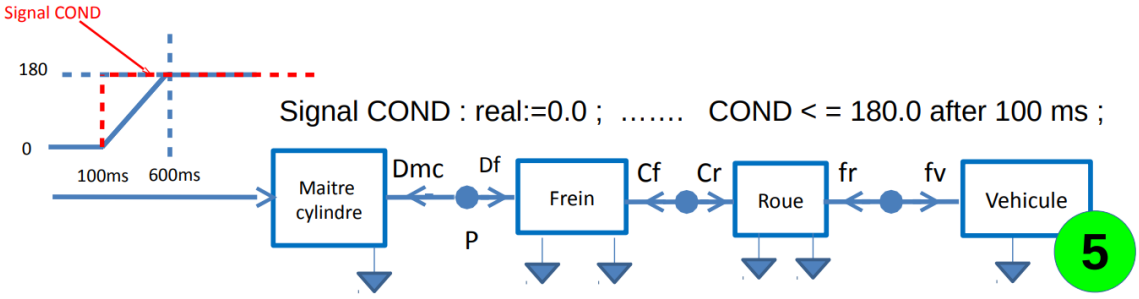


FIGURE 14 – Intégration rampe au schéma bloc

```
156   quantity Tension across Courant through t5;
157   quantity s_force : real;
158
159   signal COND : real:=0.0;
160   signal COND2 : real:=0.0;
161
162   begin
163
164   COND <=180.0 after 100 ms;
165   COND2 <=5.0 after 100 ms;
166
167   s_force==cond'ramp(0.5);
168   Tension==cond2'ramp(2.0);
```

FIGURE 15 – Condition rampe entrée Maître cylibdre

En ajoutant cette condition nous pouvant oberser avec l'outil de simulation la nouvelle évolution de la vitesse du véhicule, qui maitenant, ne décroit plus linéairement. On constate que le véhicule met plus de temps pour freiner entièrement son déplacement, ce qui peut être expliqué du fait que tant que la force maximum du frein n'est pas appliquée le temps de freinage sera plus long.

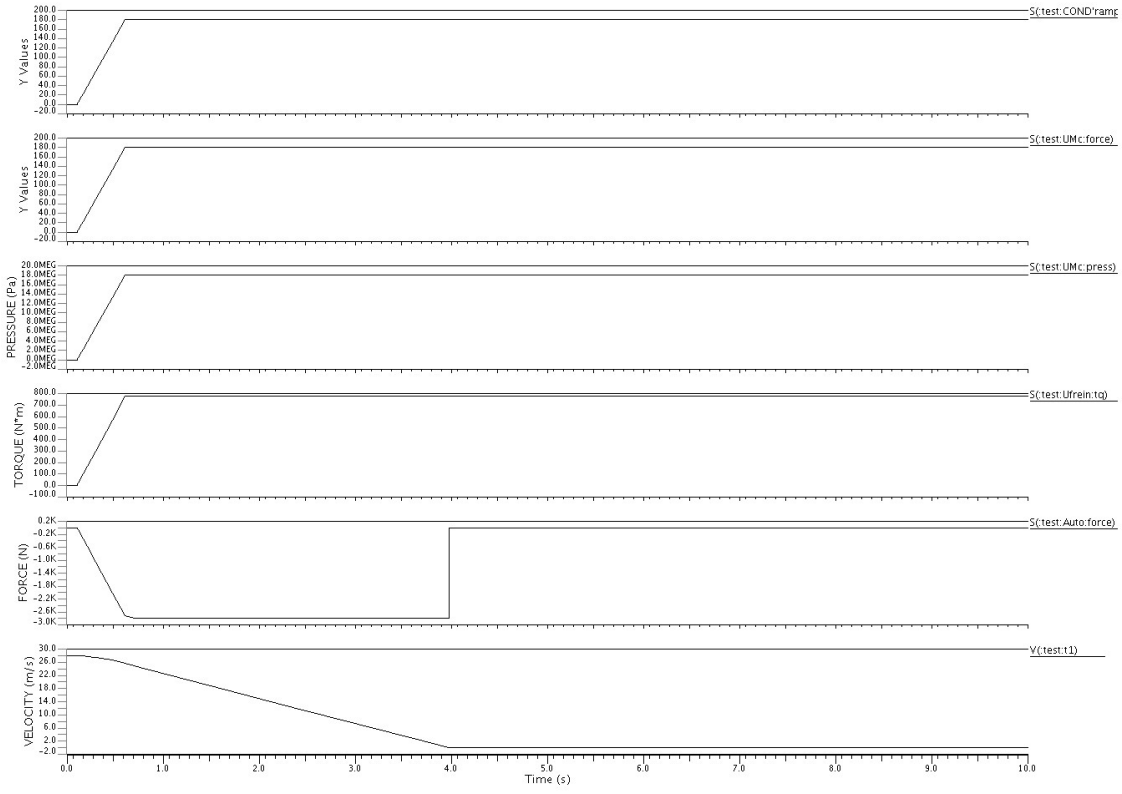


FIGURE 16 – Simulation avec temps de réaction sur le freinage

## 1.5 Modélisation du régulateur de pression

Pour conclure ces premières étapes avant d'instancier un système d'ABS on nous demande d'ajouter un régulateur de pression grâce à un signal de commande. Pour cela on utilisera l'équation suivante dans le domaine de Laplace, pour réguler la pression :

$$\frac{Pout}{Pin} = \frac{V}{5 \times (1 + 0.002p)}$$

Ce qui nous permet d'obtenir dans le domaine temporel une nouvelle équation utilisable cette fois-ci pour le code VHDL :

$$P_{out} + 0.002 \times \frac{dPout}{dt} = \frac{V \times Pin}{5}$$

Ce qui nous permet d'ajouter un nouveau bloc à notre schéma bloc système qui aura pour entrée les trois paramètres suivants :

— $T_{Pin} \rightarrow$ Pression d'entrée qui vient du maître cylindre.
— $T_{Pout} \rightarrow$ Pression de sortie en entrée du frein.
— $TV \rightarrow$ Tension correspondate à la commande du régulateur.

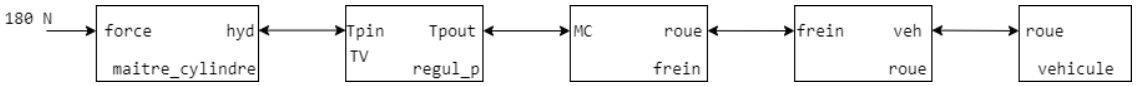


FIGURE 17 – Ajout bloc Régulateur de pression

# 2 Modélisation du système de freinage équipé d'un ABS

# 3 Intégration simplifiée du moteur thermique

# 4 Conclusion

*En conclusion, ces différentes séances de travaux pratiques nous ont permis de réaliser de la simulation de différentes architectures électroniques en relation étroite avec leur environnement. Pour cela nous avons mis en œuvre les connaissances théoriques acquises lors des cours magistraux et des travaux dirigés sur le VHDL-AMS. Nous avons également pu découvrir l'outil "ADVanceMS", qui est le premier simulateur de signaux mixtes de l'industrie. Grâce à son architecture unique, ADVance MS offre une simulation haute performance, permettant au solveur ModelSim d'évaluer des conceptions purement numériques et une variété d'algorithmes analogiques.*

*Par conséquent nous pouvons dire que le VHDL-AMS ne constitue pas un langage de conception ou synthèse mais plutôt un langage de description du matériel.*