

FILTRES MEDIANS

1 Introduction

Les filtres numériques médians glissants utilisent la propriété d'élimination des valeurs extrêmes inhérente à la médiane statistique d'une population de taille n . Ainsi, ces filtres éliminent le bruit essentiellement haute fréquence présent dans un signal ou une image. Ce document décrit une implantation de ces filtres en langage VHDL pour un signal ou des niveaux de luminance pour une des couleurs exprimés sur 8 bits non signés.

2 Principe de fonctionnement

Par définition, la médiane d'un ensemble de valeurs ou population est une valeur m telle que le nombre de valeurs inférieures à m est égale au nombre de valeurs supérieures à m .

Exemples :

<table border="1"><thead><tr><th>Valeurs</th><th>Médiane</th></tr></thead><tbody><tr><td>3</td><td>4</td></tr><tr><td>7</td><td></td></tr><tr><td>1</td><td></td></tr><tr><td>5</td><td></td></tr><tr><td>4</td><td></td></tr><tr><td>5</td><td></td></tr><tr><td>1</td><td></td></tr></tbody></table>	Valeurs	Médiane	3	4	7		1		5		4		5		1		Si le nombre de valeurs est pair la médiane devient :
Valeurs	Médiane																
3	4																
7																	
1																	
5																	
4																	
5																	
1																	
	<table border="1"><thead><tr><th>Valeurs</th><th>Médiane</th></tr></thead><tbody><tr><td>3</td><td>4,5</td></tr><tr><td>7</td><td></td></tr><tr><td>1</td><td></td></tr><tr><td>5</td><td></td></tr><tr><td>4</td><td></td></tr><tr><td>5</td><td></td></tr></tbody></table>	Valeurs	Médiane	3	4,5	7		1		5		4		5			
Valeurs	Médiane																
3	4,5																
7																	
1																	
5																	
4																	
5																	

En fait, la médiane s'obtient en effectuant un tri par valeur croissante puis en prenant en compte la valeur du milieu du tableau (dans le cas suivant la 4^e valeur du tableau trié).

Exemple :

Valeurs	Valeurs triées	Médiane
3	1	4
7	1	
1	3	
5	4	
4	5	
5	5	
1	7	

Le calcul de la médiane glissante revient à effectuer un tri des N échantillons¹ précédents contenus dans un tableau.

¹ Par exemple $N = 3, 5, 7$ etc. pour un signal ou 9 pour une image. Si $N = 3$, le tableau de tri ou *kernel* contient les échantillons $n, n-1$ et $n-2$ du signal.

Pour cette implantation, nous choisirons le tri par approche ou échange successif appelé aussi tri par bulle. Les valeurs binaires sont signées pour un signal ou non signées pour des niveaux de luminance d'une couleur d'une image (voir [figure 2-1](#)) :

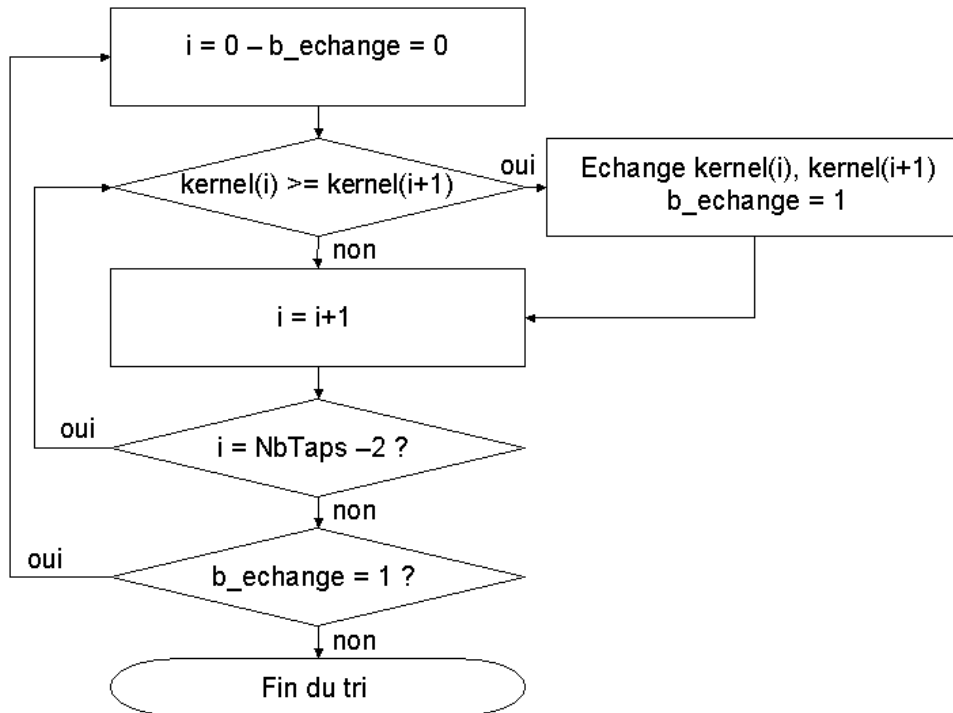
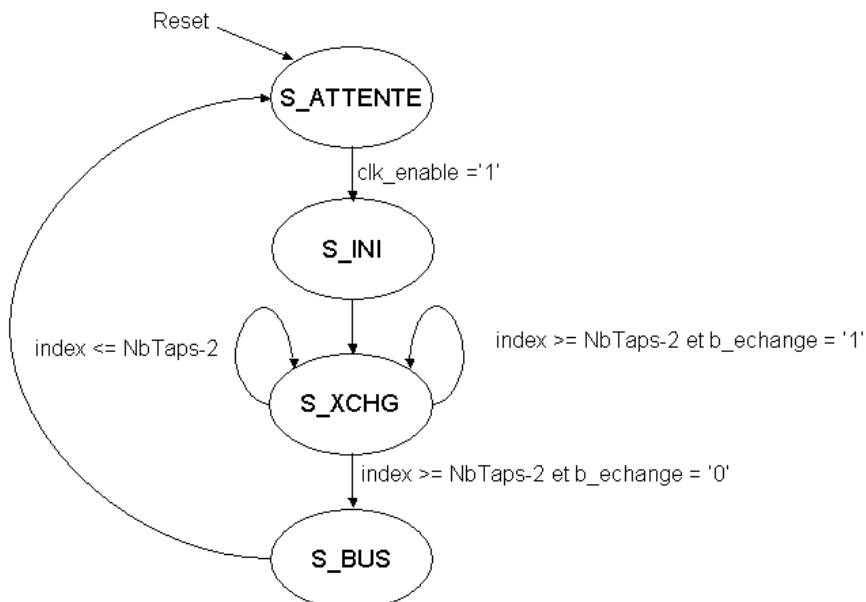


Figure 2-1 : Algorithme de tri par bulle

3 Implantation du tri et de la détermination de la médiane

Le graphe de machine à états finis ([figure 3-1](#)) et le chronogramme ([figure 3-2](#)) représentent l'implantation du tri par bulle pour une description VHDL de simulation avant synthèse (voir [§6-7](#)) :



clk_enable = autorisation cycle de filtrage du signal
 b_echange = '0' : aucun échange de valeur pendant le tri / '1' échange de valeur pendant le tri
 NbTaps = Nombre d'échantillons utilisés pour calculer la médiane des valeurs

Figure 3-1 : Machine à états finis du tri par bulle

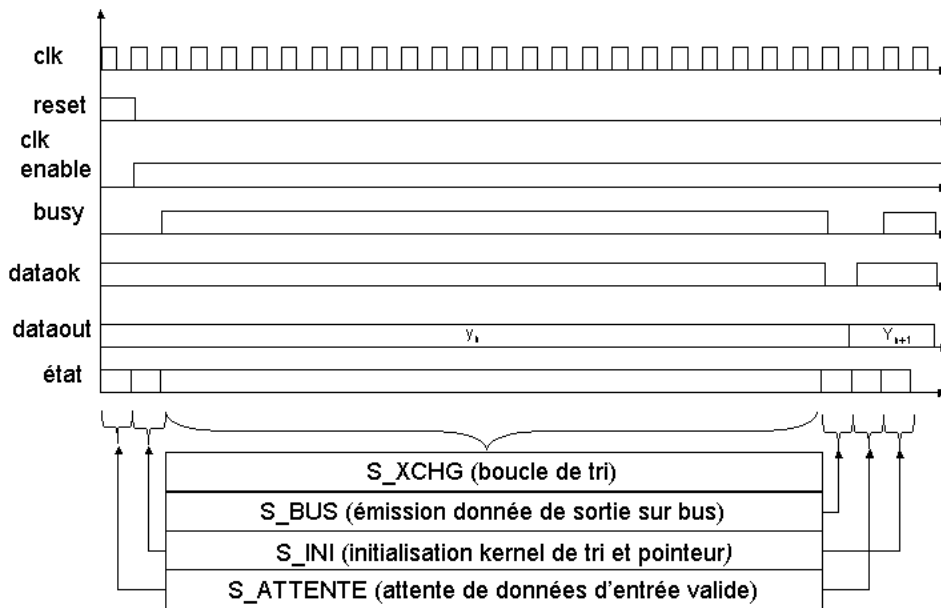


Figure 3-2 : Chronogramme de fonctionnement du tri par bulle

Le système d'adressage (Figure 3-3) défini dans le § 7.4 de la note d'application *Implementing IIR/FIR Filters with Motorola's DSP56000/ DSP56001 - FIR Implementation on the DSP56001* John Lane Garth Hillman Motorola 1991 est utilisé pour l'insertion du dernier échantillon du signal dans les mémoires d'échantillons et de tri (voir les résultats de simulation §6-4 et §6-6).

Ce système d'adressage évite les mouvements d'échantillons à travers l'espace mémoire de tri. C'est un avantage significatif.

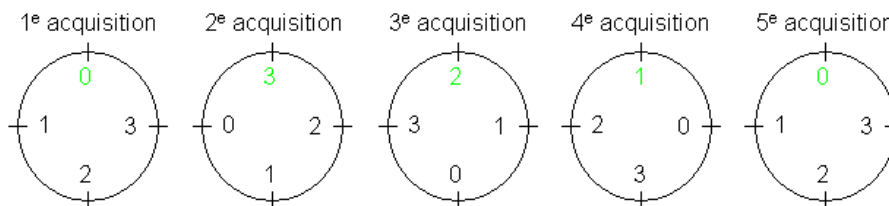


Figure 3-3 : Adressage pour l'insertion des échantillons récents dans les mémoires

- La machine reste dans l'état *S_ATTENTE* tant que la synchronisation *clk_enable* est à 0. elle quitte cet état sur le front montant du signal *clk_enable*. Les principaux indicateurs sont réinitialisés au cours de cet état ;
- Au cours de l'état *S_INI*, la mémoire de tri *Kernel* reçoit les échantillons *y* compris le plus récent. La mémoire des échantillons reçoit l'échantillon le plus récent. Cet état ne dure que pendant un cycle d'horloge ;
- Le cycle *S_XCHG* correspond au cycle de tri et d'incrémentatation des indices sur la mémoire de tri selon l'organigramme de tri par bulle. Lorsque les indices atteignent la valeur du nombre d'échantillons -2 et si aucun échange a eu lieu, l'état suivant sera celui d'émission sur le bus *S_BUS*. Au cours de cet état, les données sur le bus sont déclarées **invalides** (*data_ok* = '1') et le filtre est **occupé** (*busy* = '1') ;
- Au cours du cycle *S_BUS*, la valeur médiane de la table de tri est émise sur le bus. Au cours de cet état, les données sur le bus sont déclarées **valides** (*data_ok* = '0') et le filtre est signalé comme étant **disponible** (*busy* = '0'). Cet état ne dure qu'un cycle d'horloge

Remarque importante : Le nombre de cycle de l'état S_XCHG est fluctuant². Il dépend étroitement du nombre de tri à effectuer pour obtenir une mémoire de test complètement triée³ (§4-3). C'est pourquoi, l'indicateur de débordement de capacité (figure 3-4) doit être pris en compte pour un fonctionnement correct pour obtenir un fonctionnement correct :

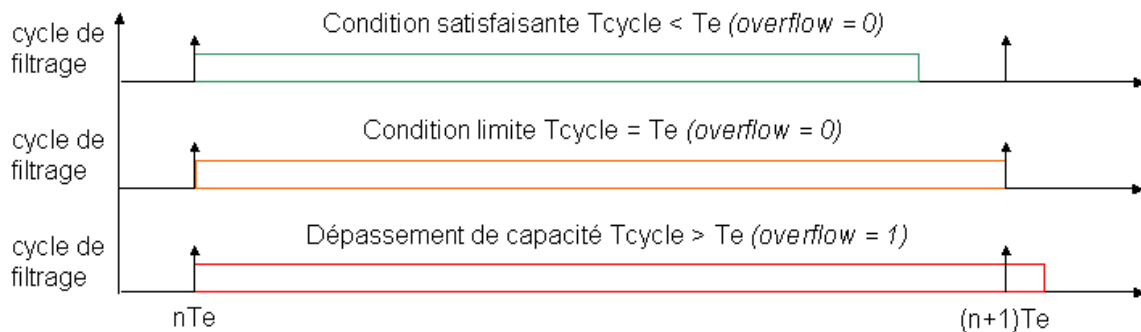


Figure 3-4 : Indicateur de débordement

Remarque sur la description VHD : Pour obtenir une description simple avec un nombre minimal d'états (4), les indices utilisés dans le tri par bulle, la mémoire de tri et l'indicateur d'échange sont des variables internes au process séquentiel de la machine à états finis. Dans ce cas, les données sont disponibles immédiatement alors que les données stockés dans des signaux sont disponibles au prochain front du signal d'horloge.

4 Résultats de simulation

Les simulations correspondent aux deux modes de fonctionnement du filtre médian :

- Le mode signal (§4-1) : filtre médian glissant avec 3 échantillons ;
- Le mode image⁴ (§4-2) : filtre médian glissant avec 9 pixels répartis autour du pixel central noté $p(0,0)$ dans le tableau qui suit :

$p(-1,-1)$	$p(0,-1)$	$p(1,-1)$
$p(-1,0)$	$p(0,0)$	$p(1,0)$
$p(-1,1)$	$p(0,1)$	$p(1,1)$

Dans les deux cas, le programme *tb_sinus.exe* crée les signaux d'entrée de simulation. Ce programme génère des signaux aléatoires en virgule fixe de 8 bits à 32 bits. Si le nombre de bits de données est 8 les valeurs générées peuvent être considérés soit comme des signaux binaires signés, soit comme des niveaux de luminosité d'une couleur d'une image exprimés sur 8 bits non signés.

² Pour les filtres à réponse impulsionnelle finie, le nombre de cycles de calcul – prédictible - dépend du nombre d'échantillons sur lesquels porte le calcul.

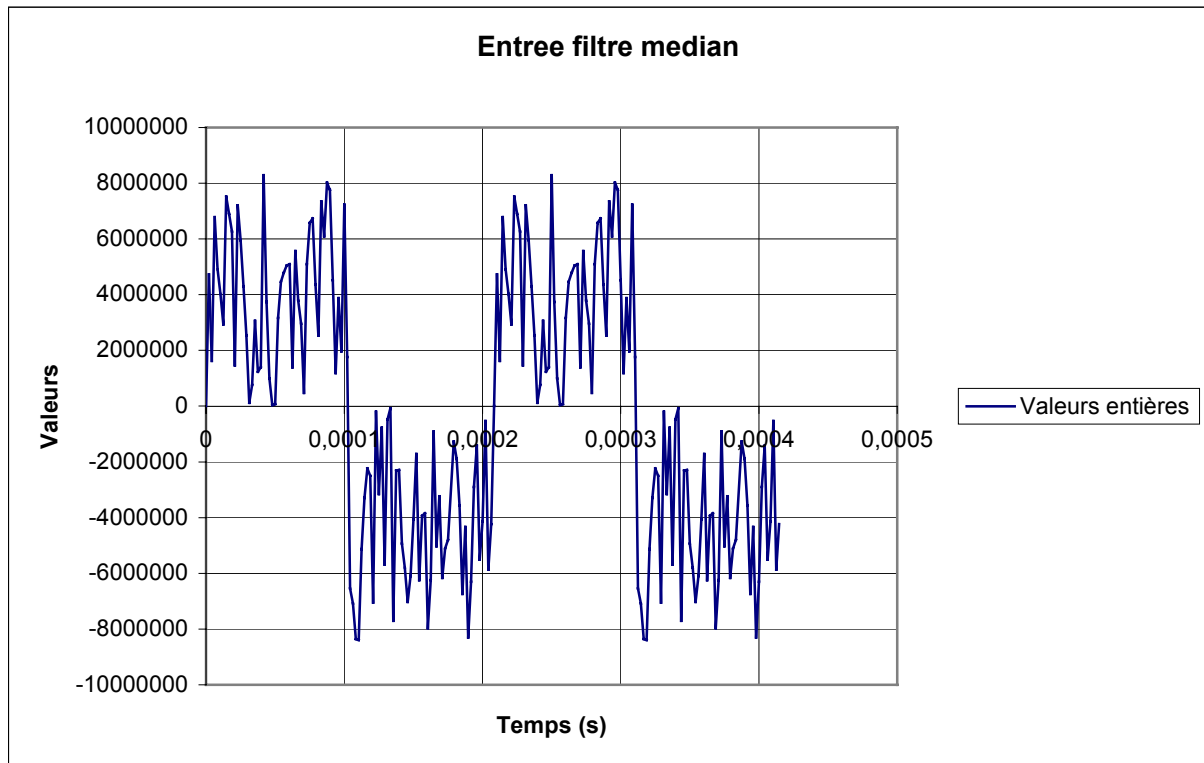
³ Le minimum correspond au nombre d'échantillons à trier si aucun échange est nécessaire.

⁴ Dans ce cas, les données doivent être stockées au préalable dans des tableaux 9 lignes, une colonne. Leur adressage doit être modifié pour conserver la cohérence dans l'image.

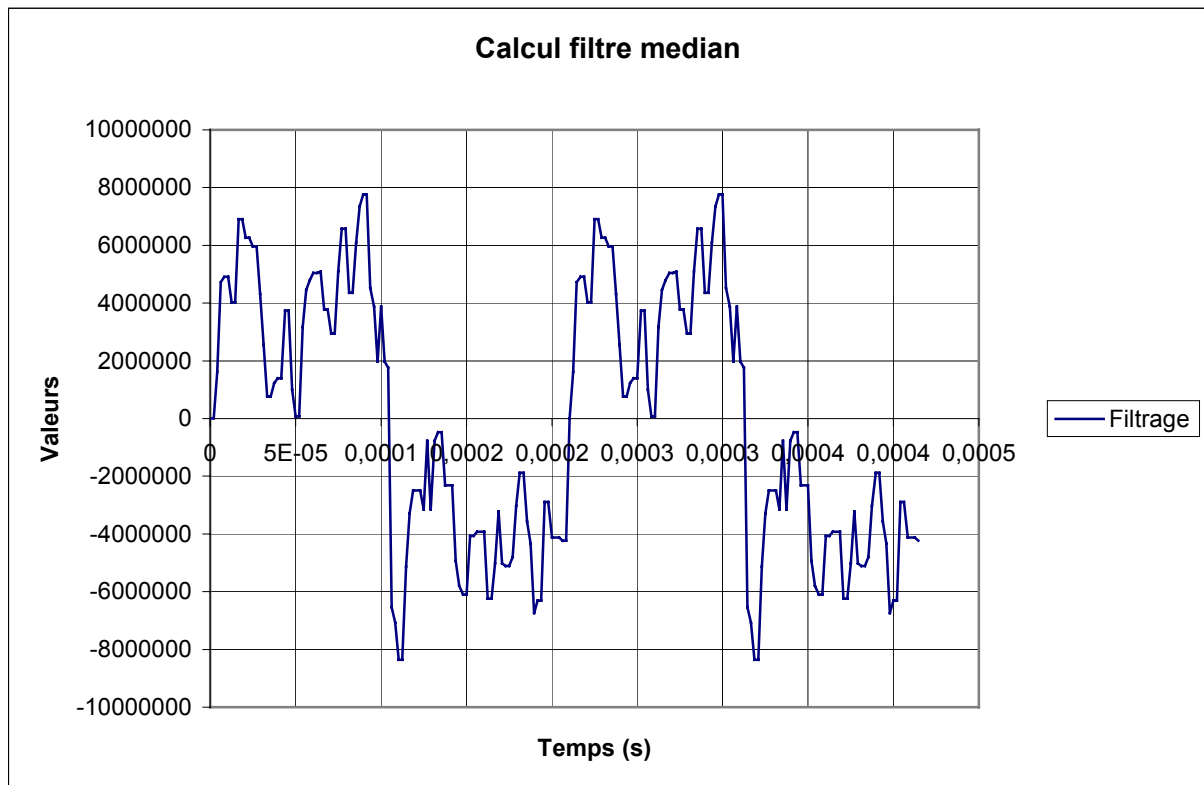
4-1 Simulation mode signal – données sur 24 bits signés

4-1-1 Signal aléatoire

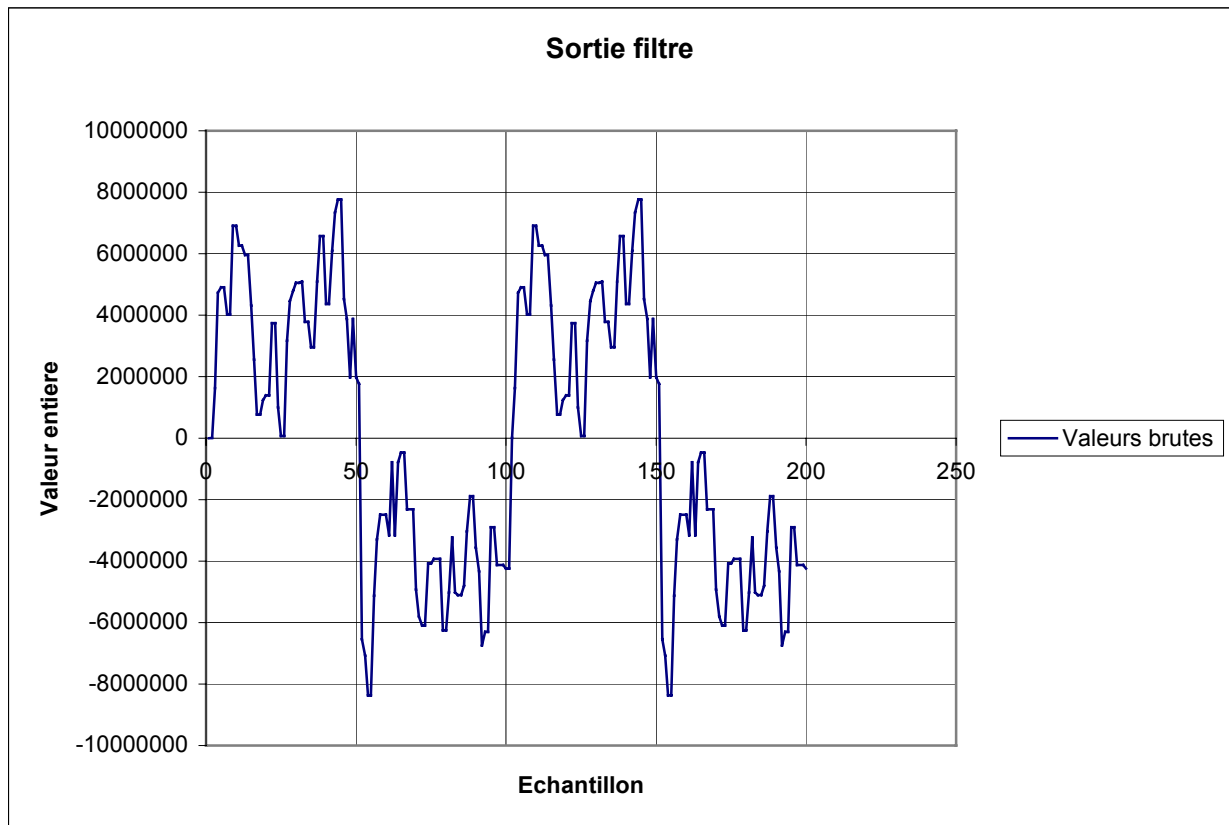
4-1-1-1 Signal d'entrée



4-1-1-2 Détermination du signal de sortie

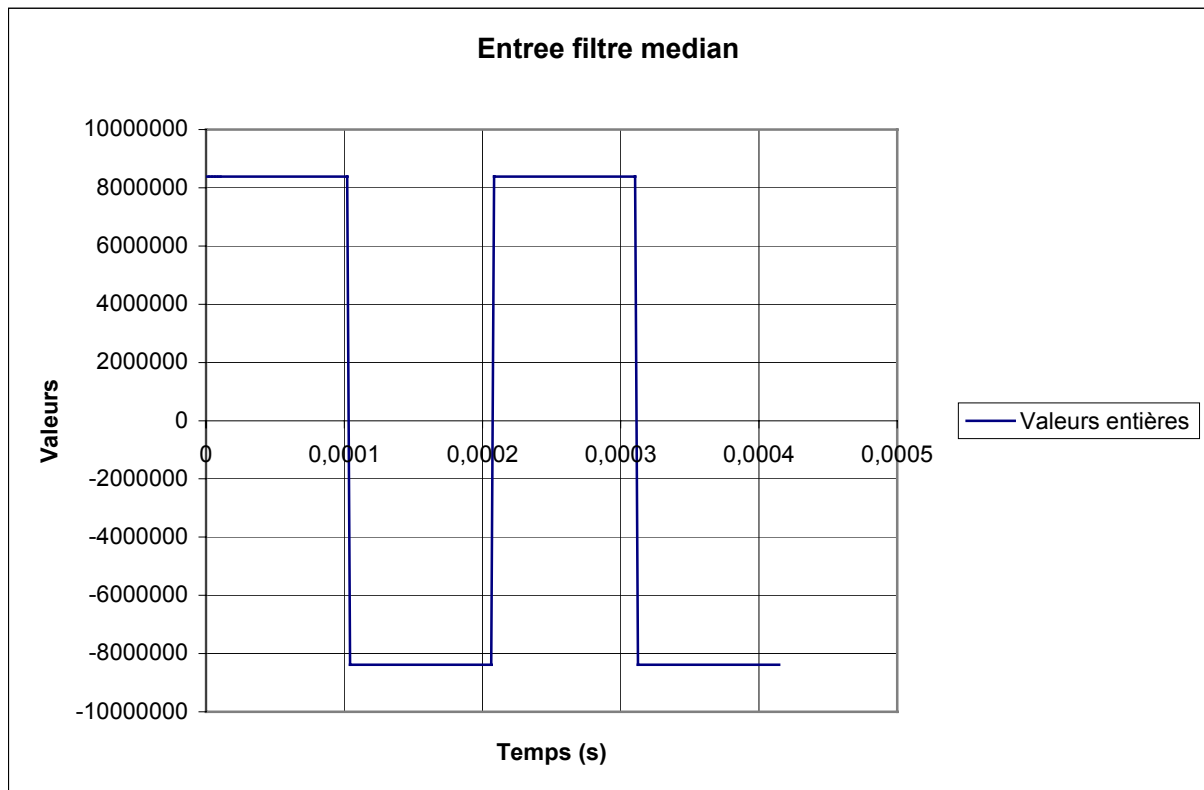


4-1-1-3 Résultats de simulation

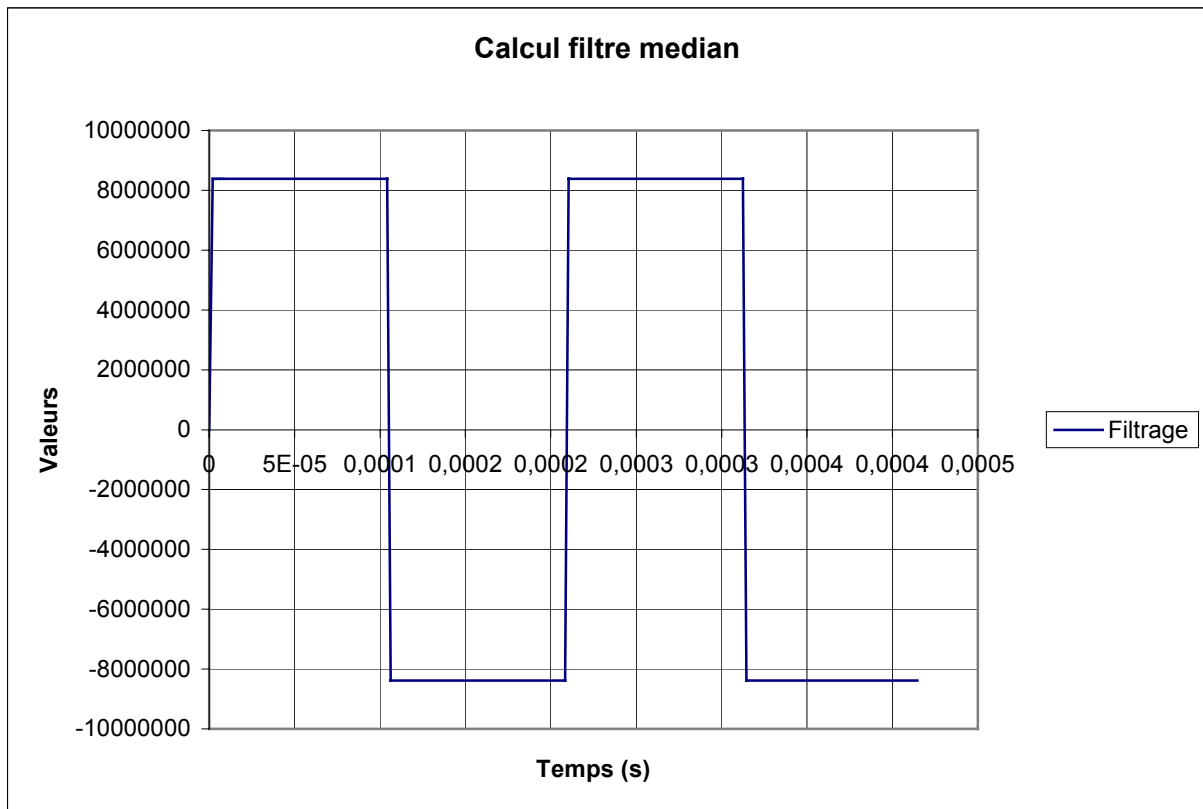


4-1-2 Signal carré

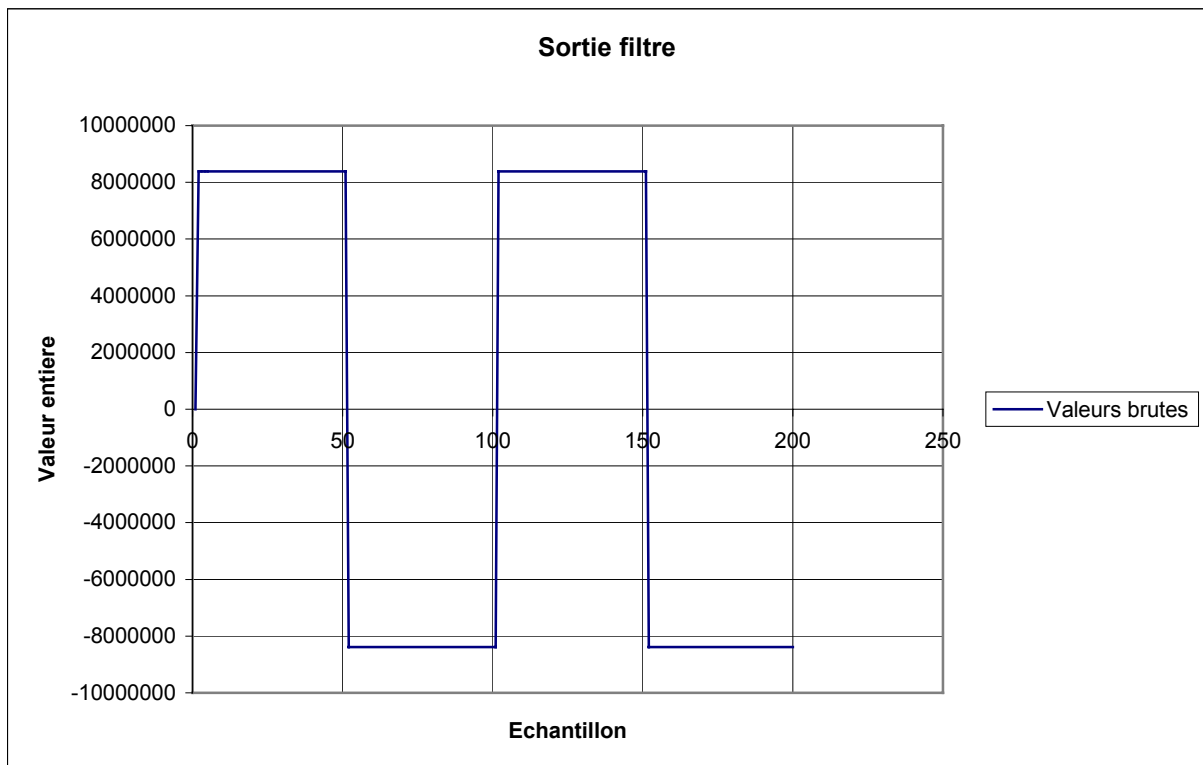
4-1-2-1 Signal d'entrée



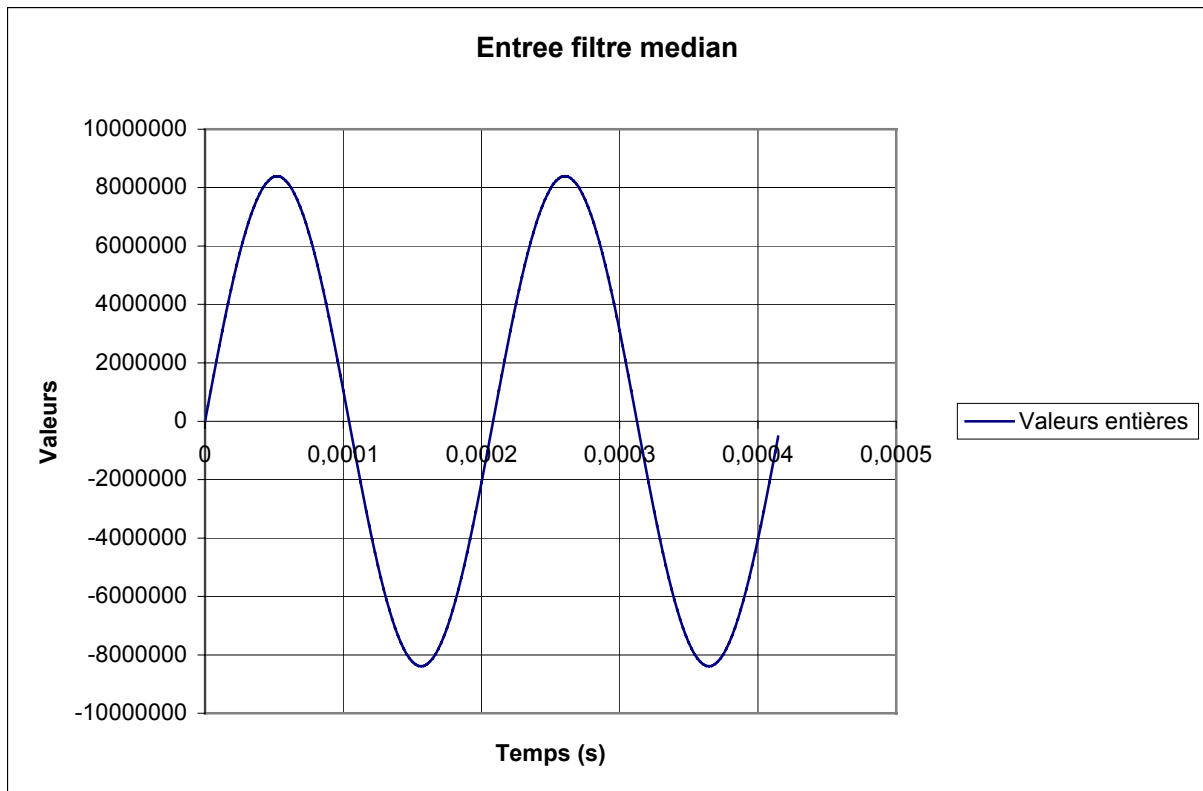
4-1-2-2 Détermination du signal de sortie



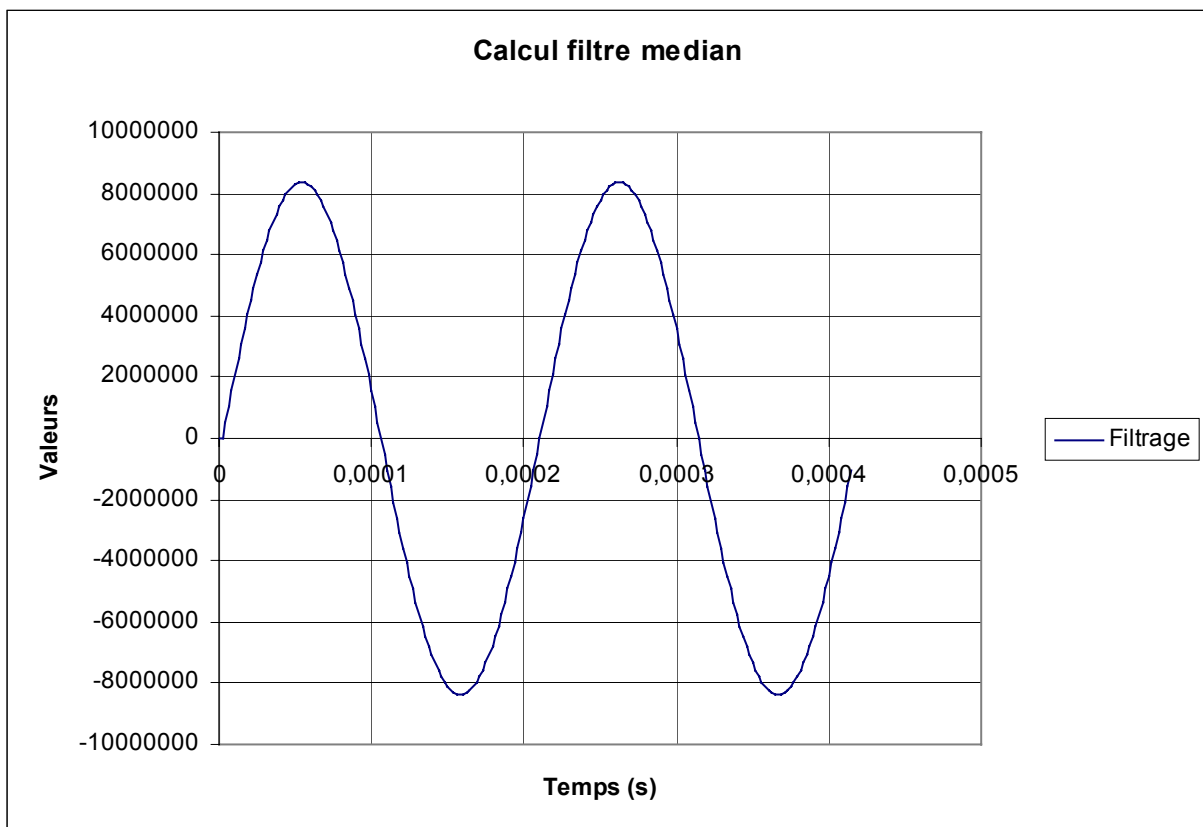
4-1-2-3 Résultats de simulation



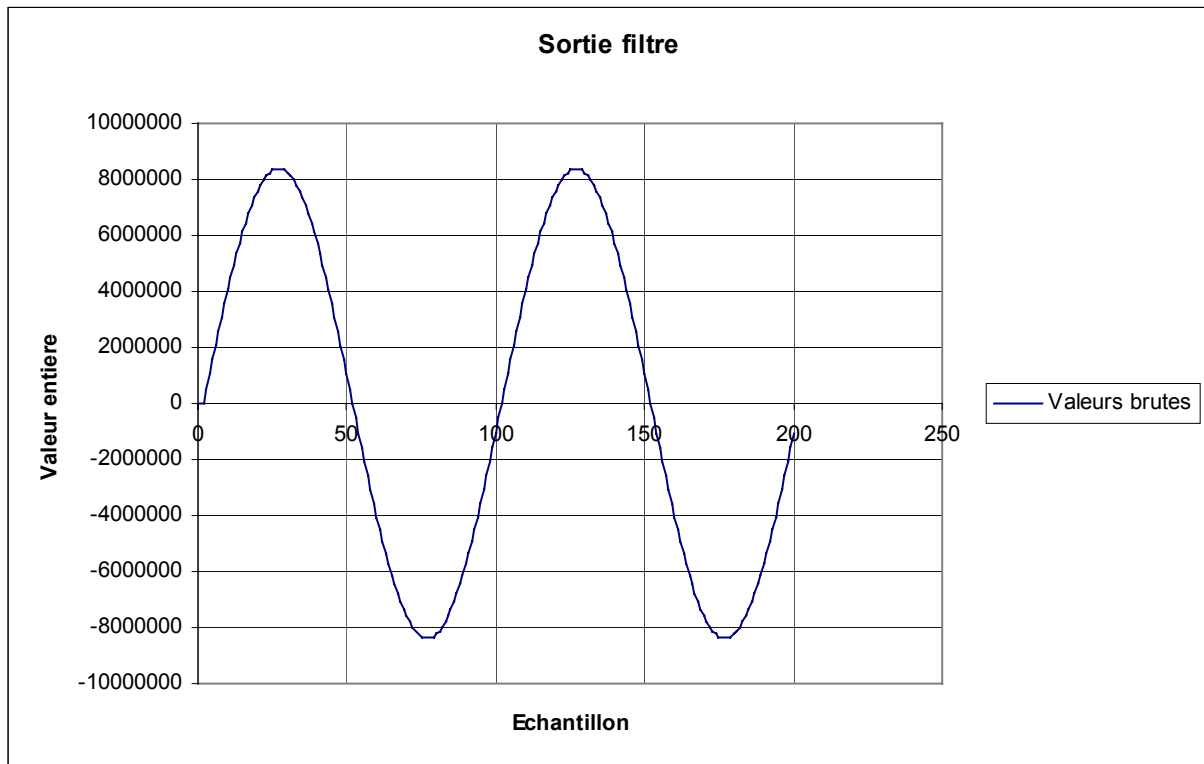
4-1-3 Signal sinusoïdal
4-1-3-1 Signal d'entrée



4-1-3-2 Détermination du signal de sortie

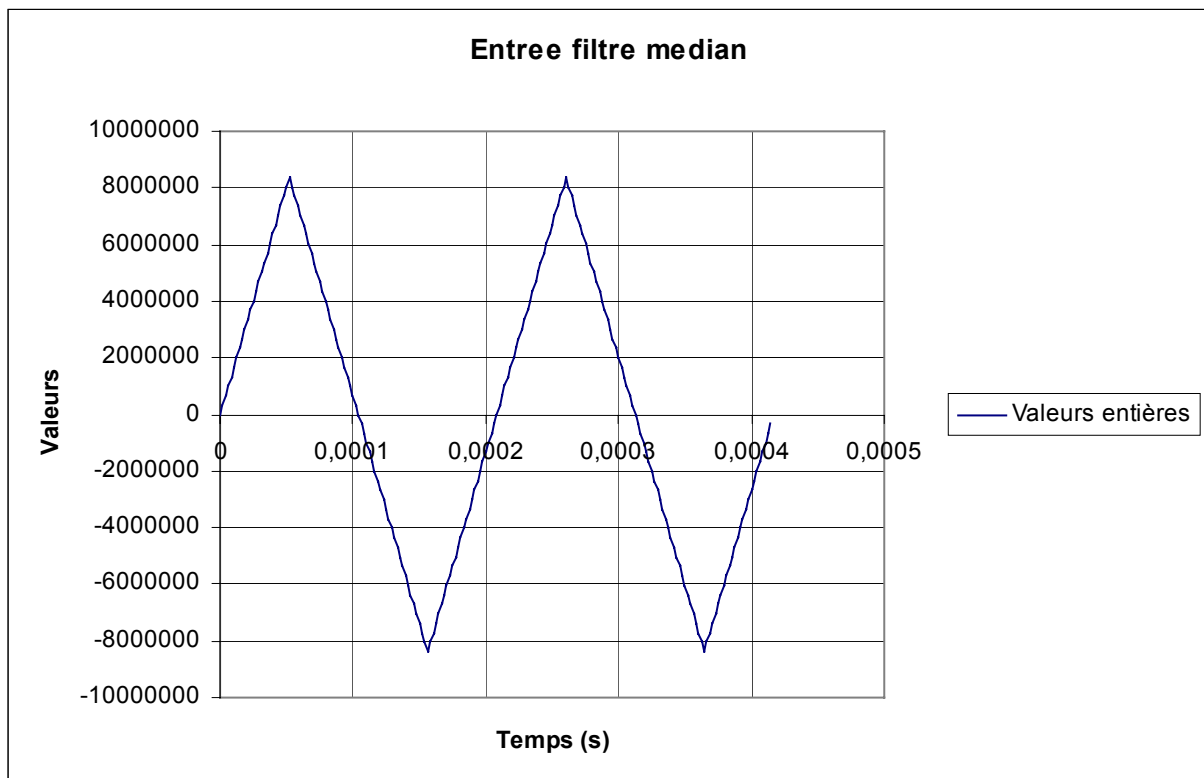


4-1-3-3 Résultats de simulation

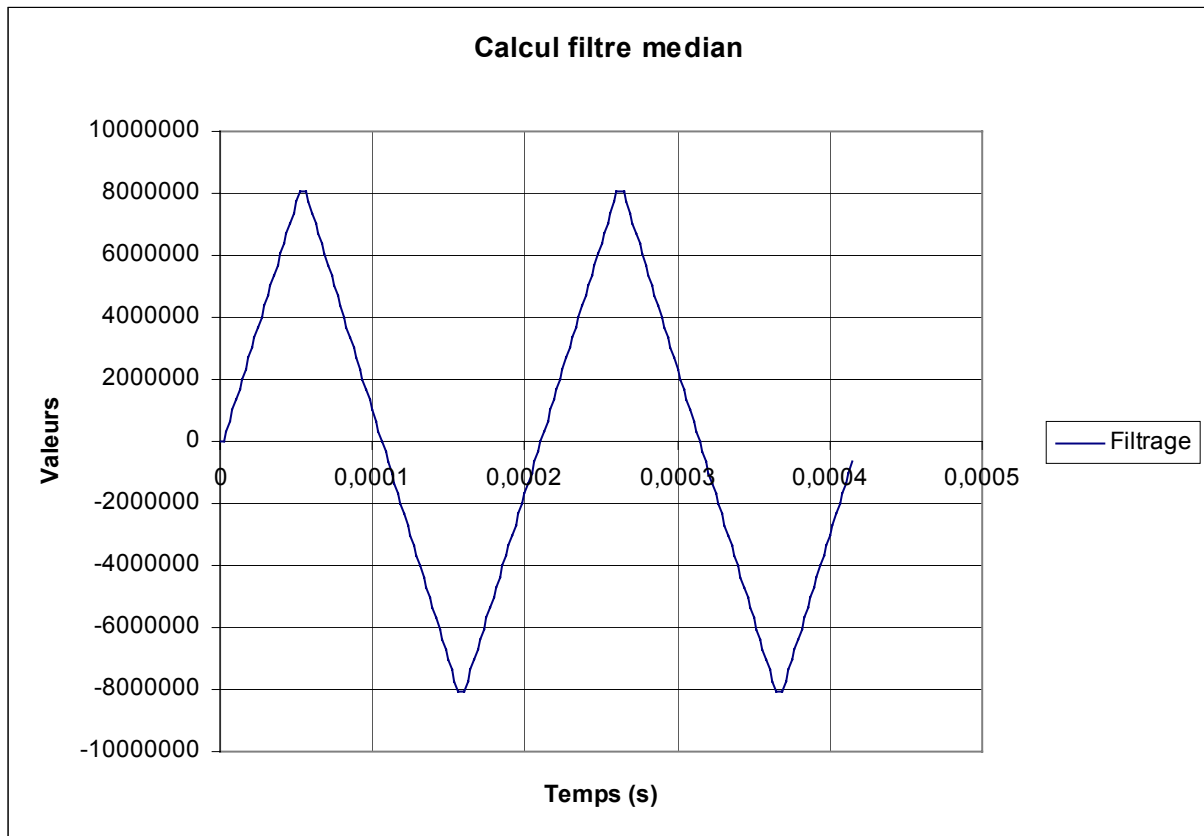


4-1-4 Signal triangulaire

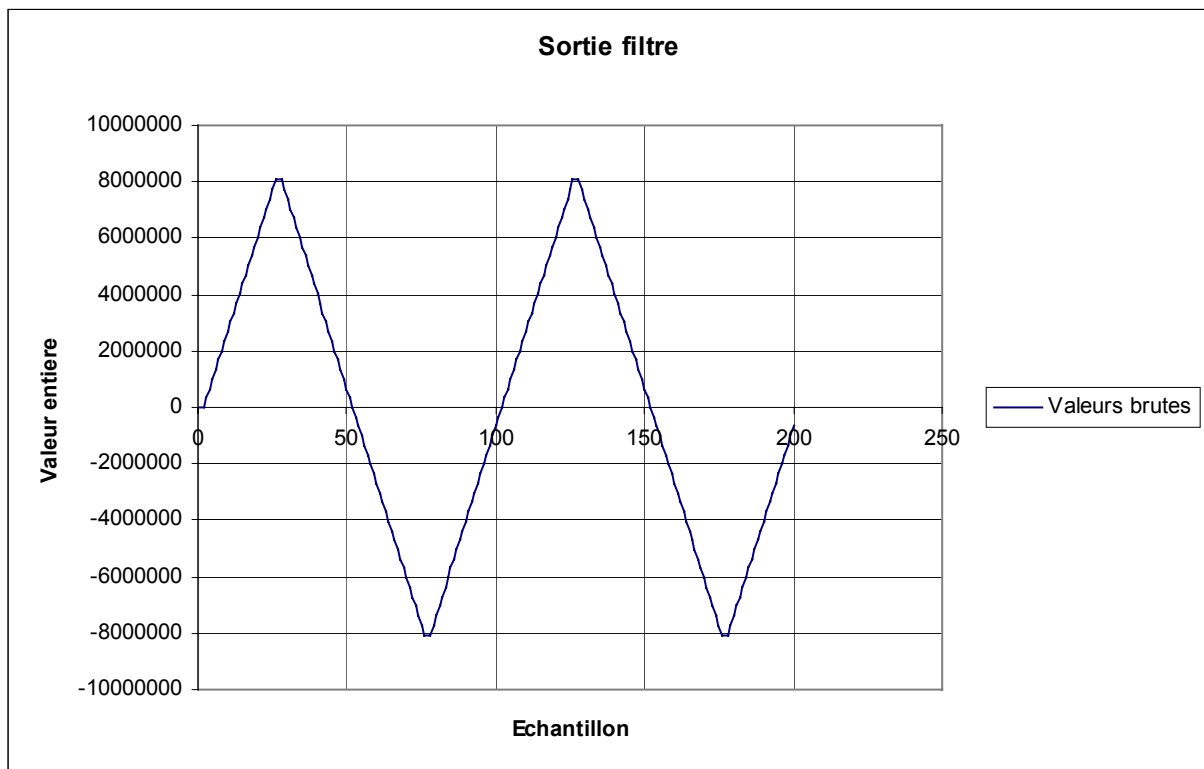
4-1-4-1 Signal d'entrée



4-1-4-2 Détermination du signal de sortie

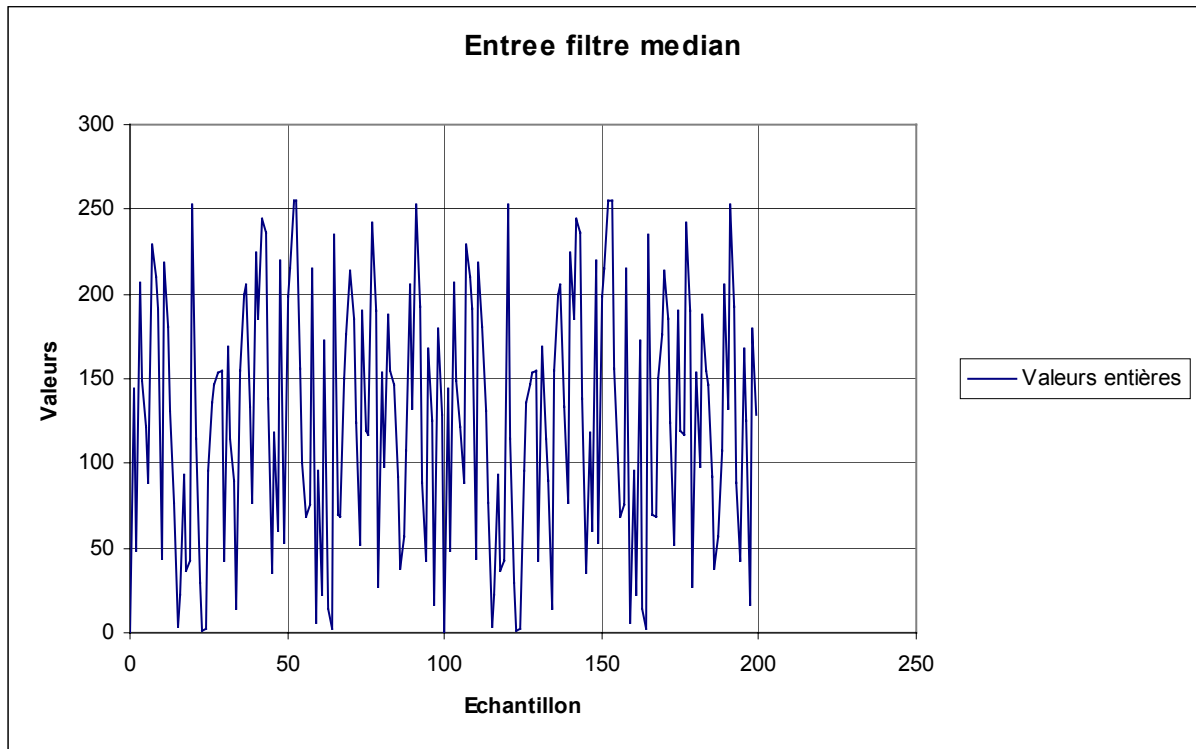


4-1-4-3 Résultats de simulation

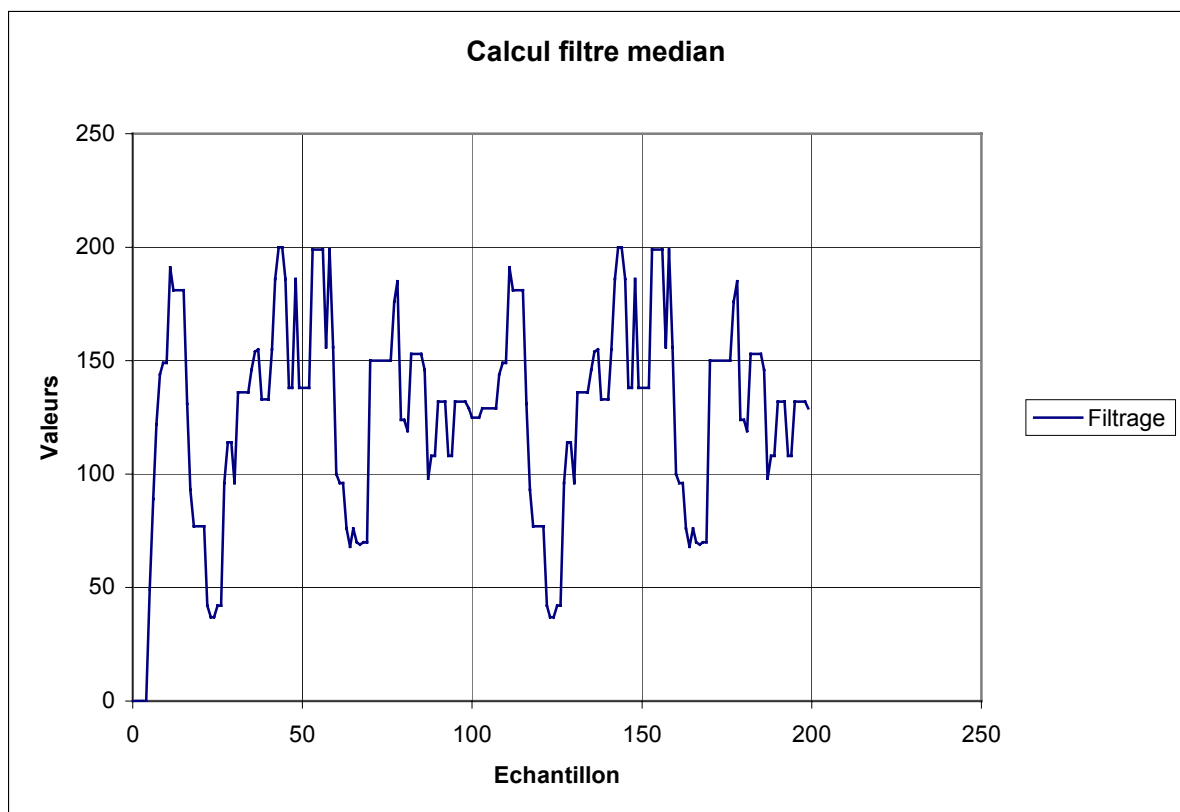


4-2 Simulation mode image – données sur 8 bits non signés

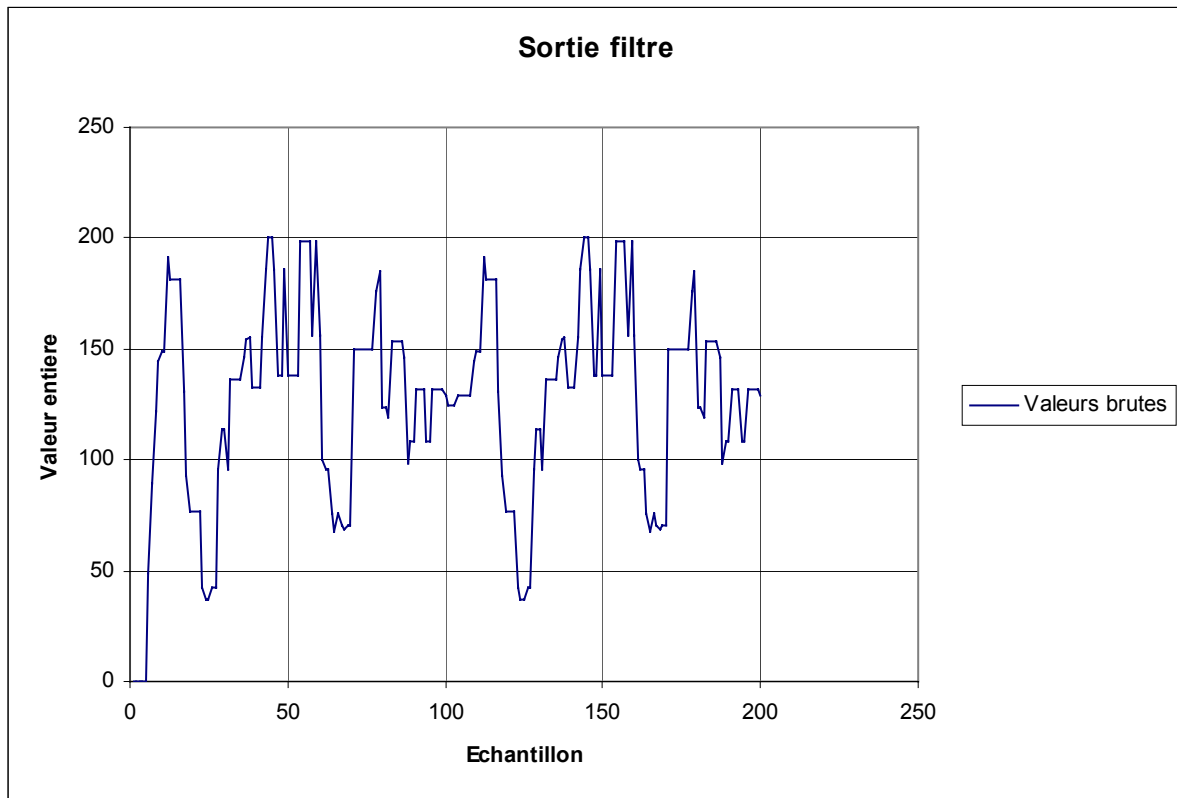
4-2-1 Données d'entrée



4-2-2 Détermination des données de sortie



4-2-3 Résultats de simulation

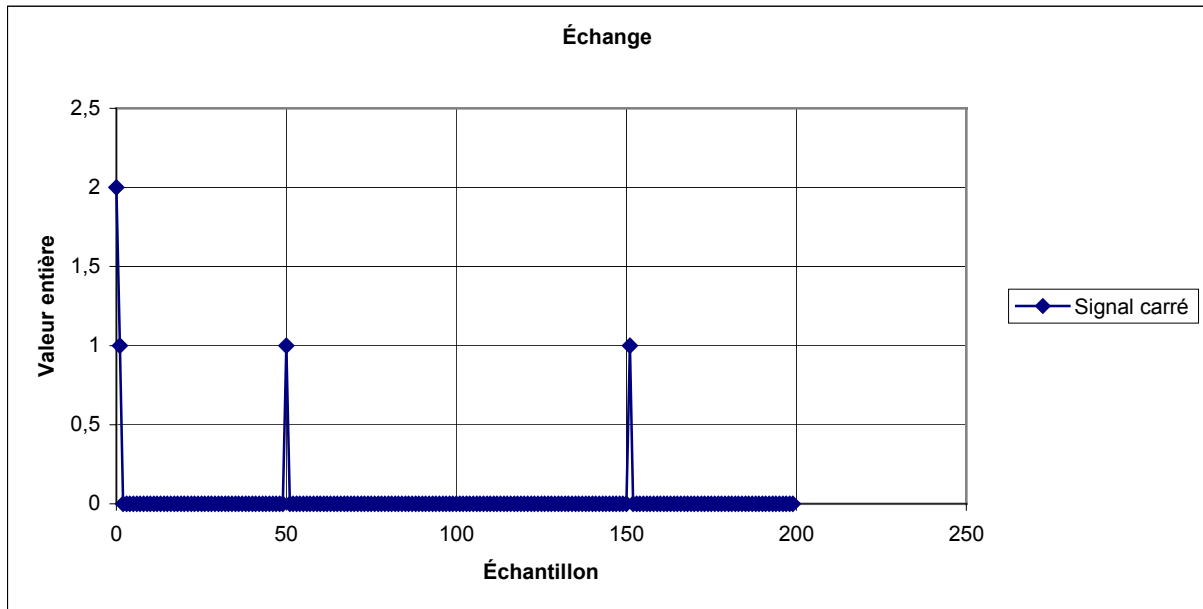


4-3 Echanges des données

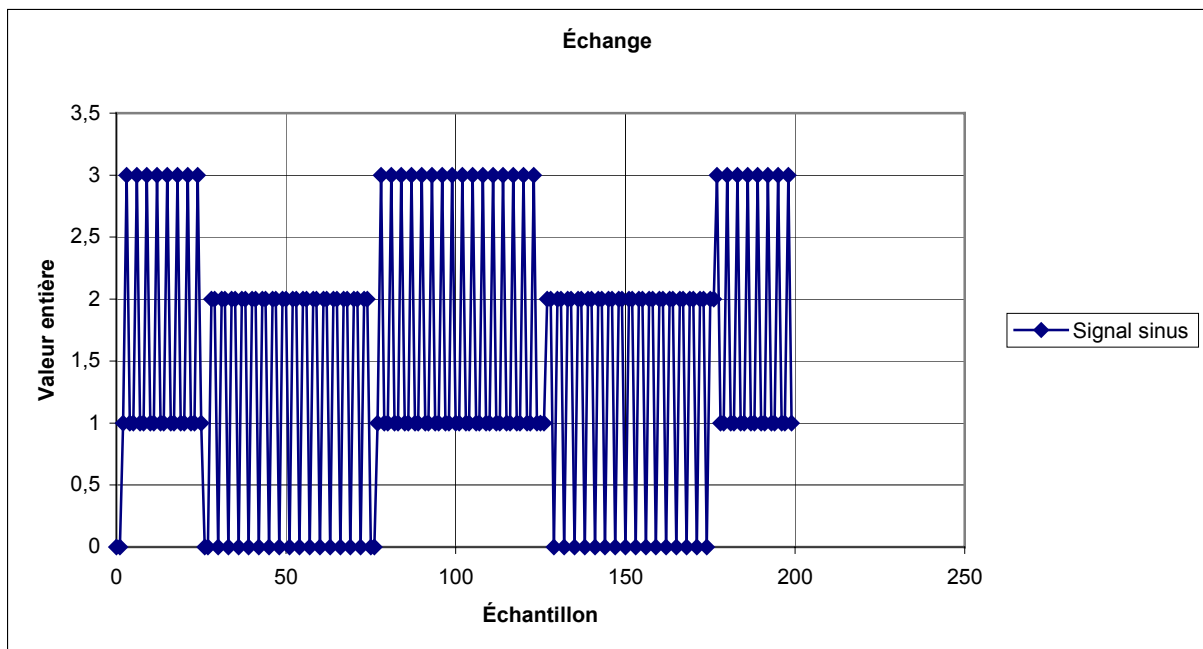
Cette partie montre les résultats obtenus en incluant un compteur d'échange dans les descriptions VHDL. Ce compteur ne doit être implanté que pour les descriptions avant synthèse. Ces comparaisons concernent les signaux et les niveaux de gris aléatoires. Les signaux carrés nécessitent des échanges uniquement lors des transitions.

4-3-1 Echanges sur les signaux

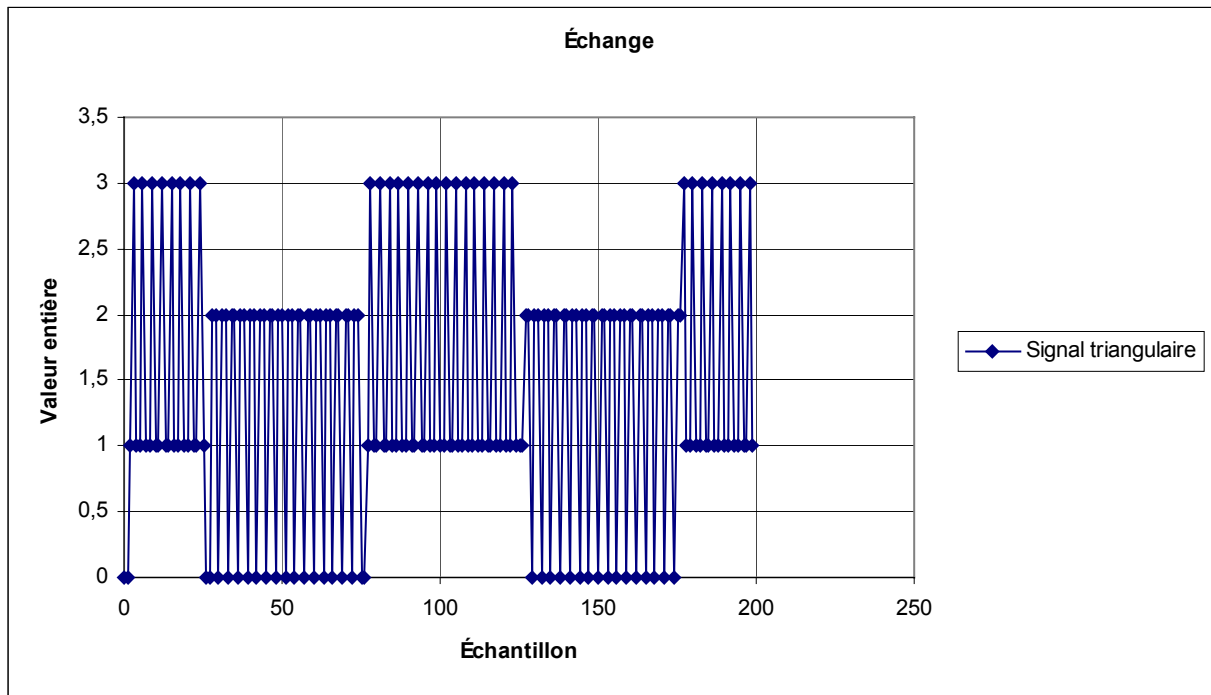
4-3-1-1 Signal carré



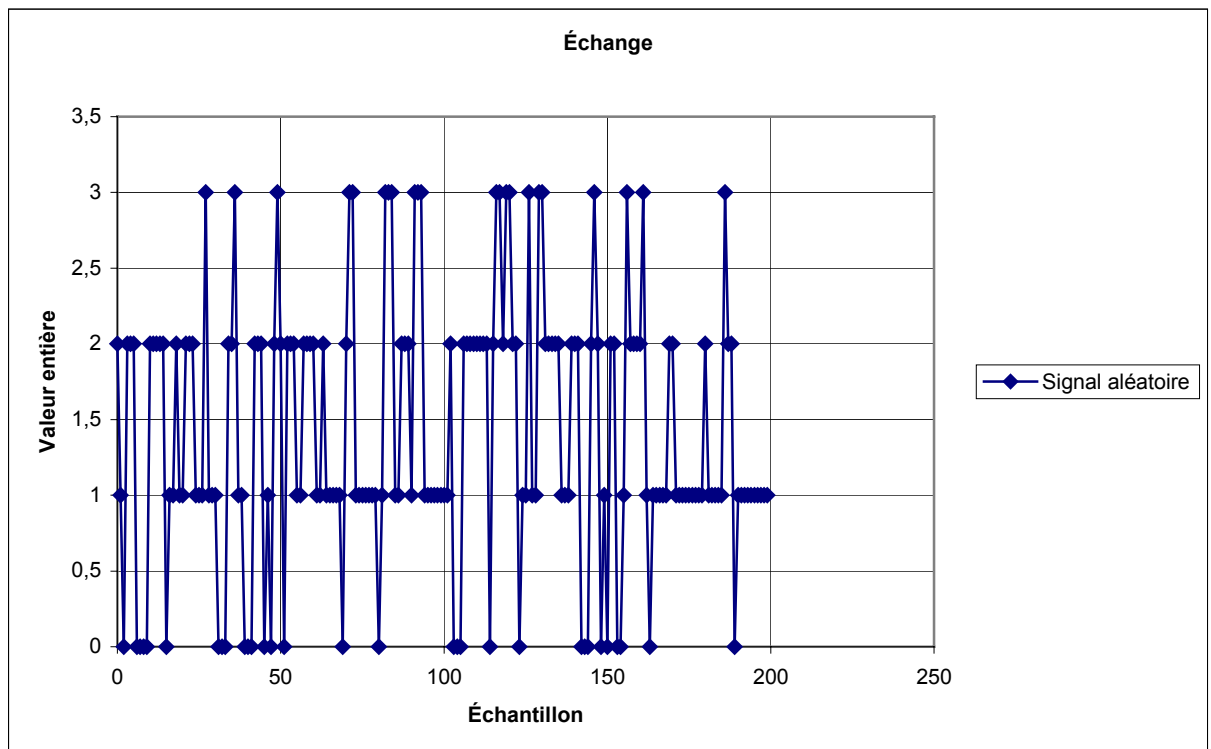
4-3-1-2 Signal sinusoïdal



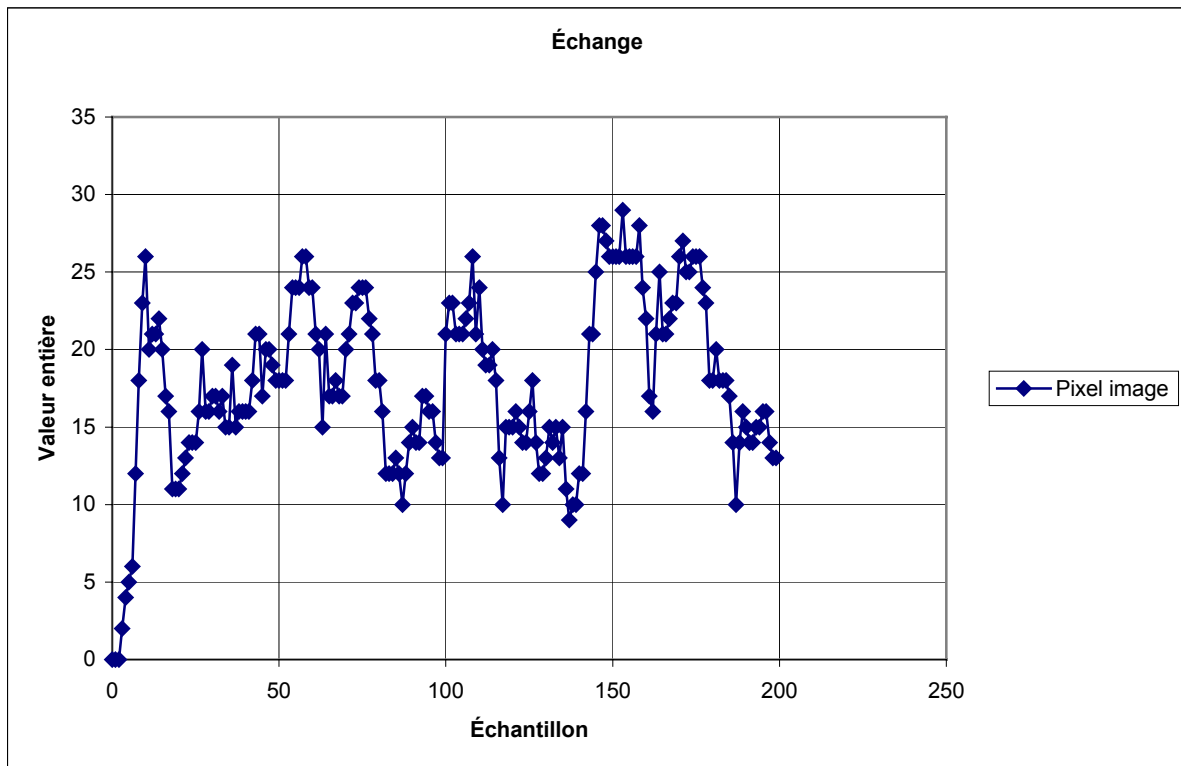
4-3-1-3 Signal triangulaire



4-3-1-4 Signal aléatoire



4-3-2 Échanges sur les images



4-3-3 Résumé statistique

Le nombre d'échanges croît avec les variations du signal et avec le nombre d'échantillons des mémoires de tri et d'échantillons précédents.

4-3-3-1 Signaux

Paramètres	Signal carre	Signal sinus	Signal triangulaire	Signal aléatoire
Médiane	0	1	1	1
Moyenne	0,025	1,465	1,465	1,395
Minimum	0	0	0	0
Maximum	2	3	3	3
Étendue	2	3	3	3

4-3-3-2 Images

Paramètres	Pixel image
Médiane	17,5
Moyenne	17,85
Minimum	0
Maximum	29
Étendue	29

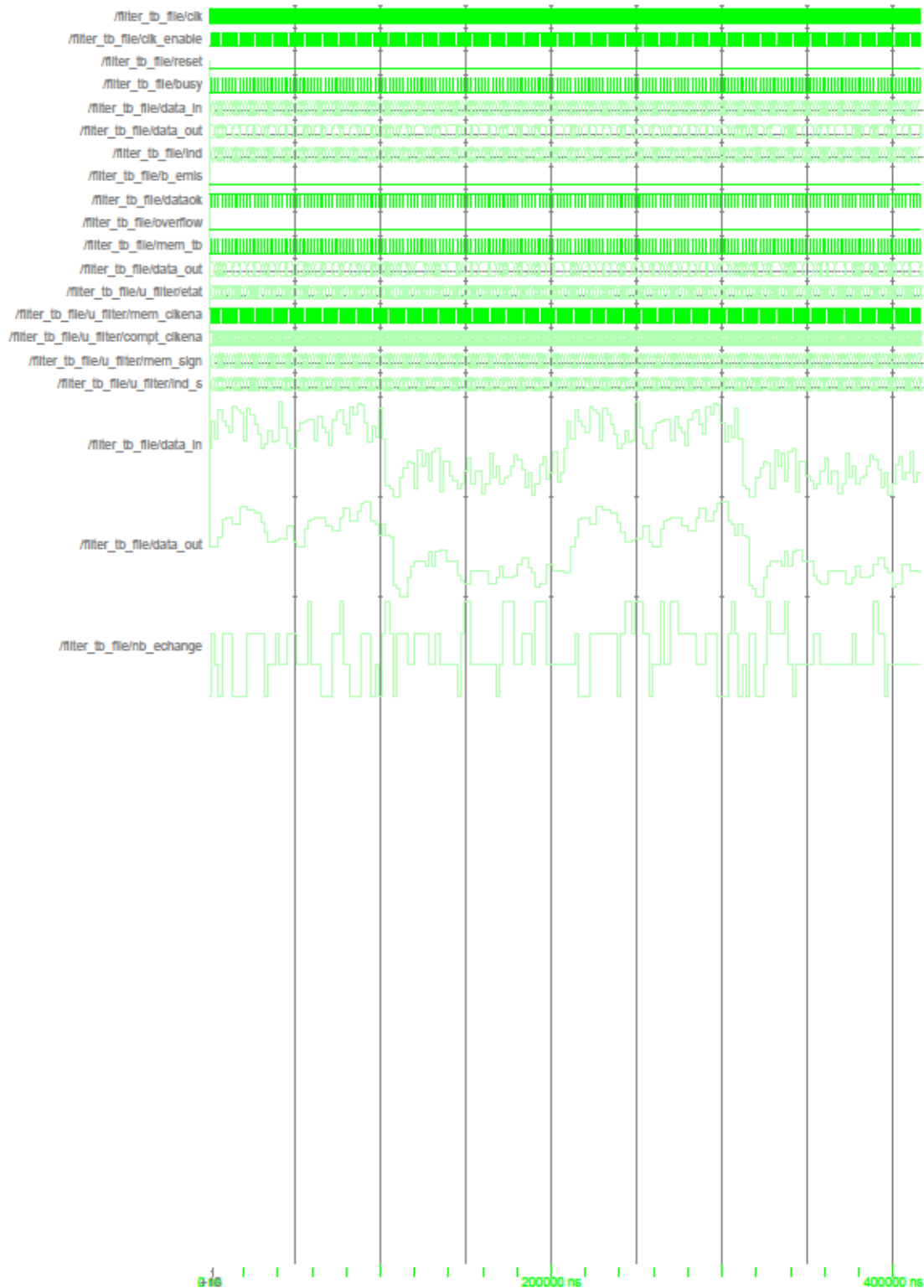
5 Conclusion

- Cette étude montre que les filtres médians éliminent le bruit haute fréquence ;
- A la différence des filtres à réponse impulsionnelle finie :
 - Les coefficients sont inconnus ;
 - Le retard de groupe et le déphasage ne sont pas définis et varient ;
 - Le nombre de cycles de calcul varie en fonction des données à filtrer et de la profondeur mémoire des échantillons précédents ;
 - L'écart temporel entre le début du filtrage et l'écriture sur le bus de l'échantillon filtré varie.
- Un algorithme de tri plus rapide que le tri par bulle permettrait une amélioration du fonctionnement des filtres.

6 Résultats de simulation dans Mentor® ModelSim®

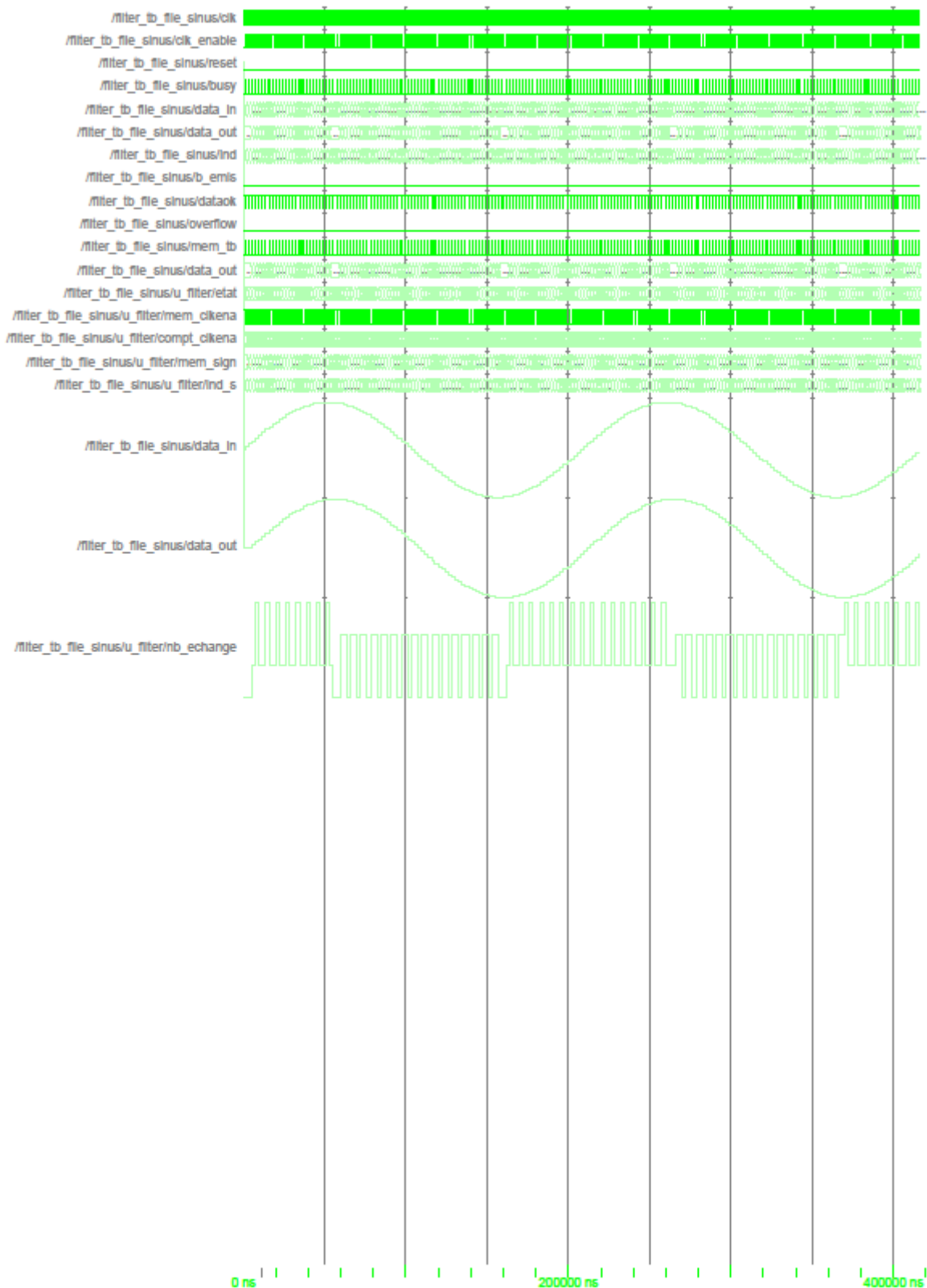
Cette partie regroupe les résultats des signaux aléatoire (§6-1), carré (§6-2), sinusoïdal (§6-3) triangulaire (§6-4) et d'images (§6-6). L'évolution des indices de stockage des échantillons précédents pour les signaux (§6-5) et les images (§6-7) est une information pertinente à prendre en compte. Le §6-8 présente la succession des états.

6-1 Signal aléatoire



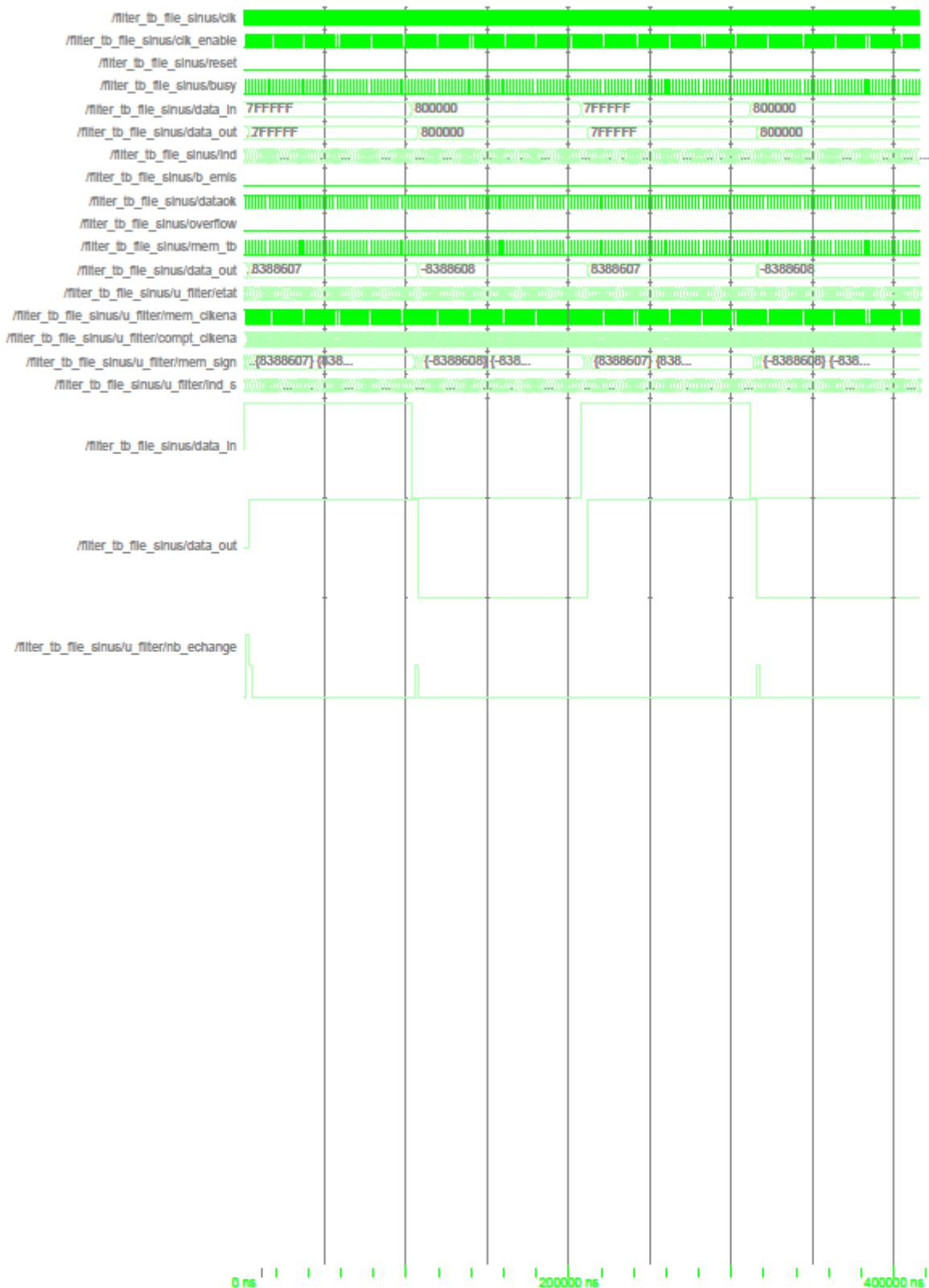
Entity: filer_tb_file Architecture: test Date: Thu Apr 01 17:35:51 Paris, Madrid (heure d'été) 2010 Row: 1 Page: 1

6-2 Signal sinusoïdal



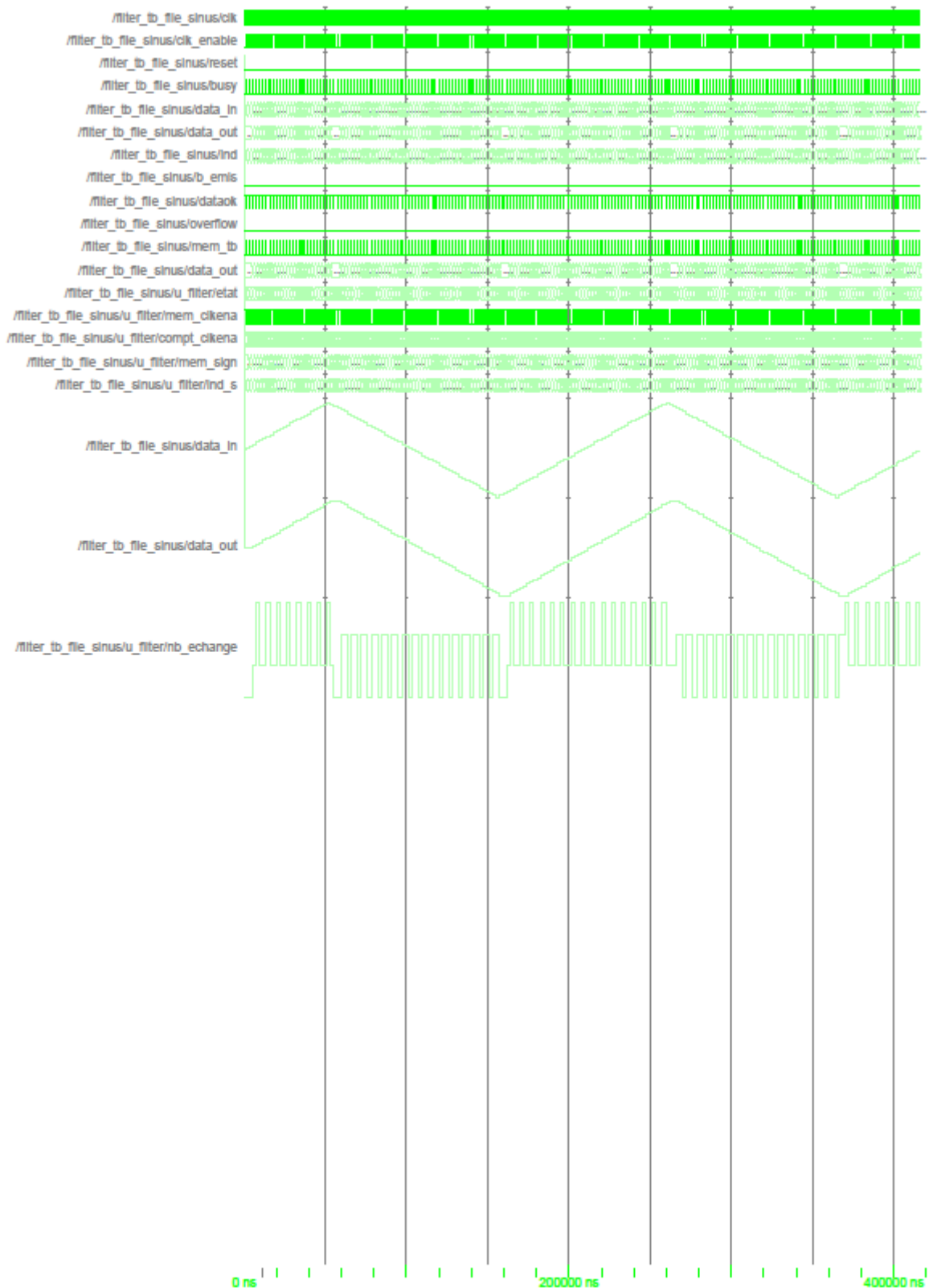
Entity: filter_tb_file_sinusk Architecture: test Date: Thu Apr 01 18:01:14 Paris, Madrid (heure d'été) 2010 Row: 1 Page: 1

6-3 Signal carré



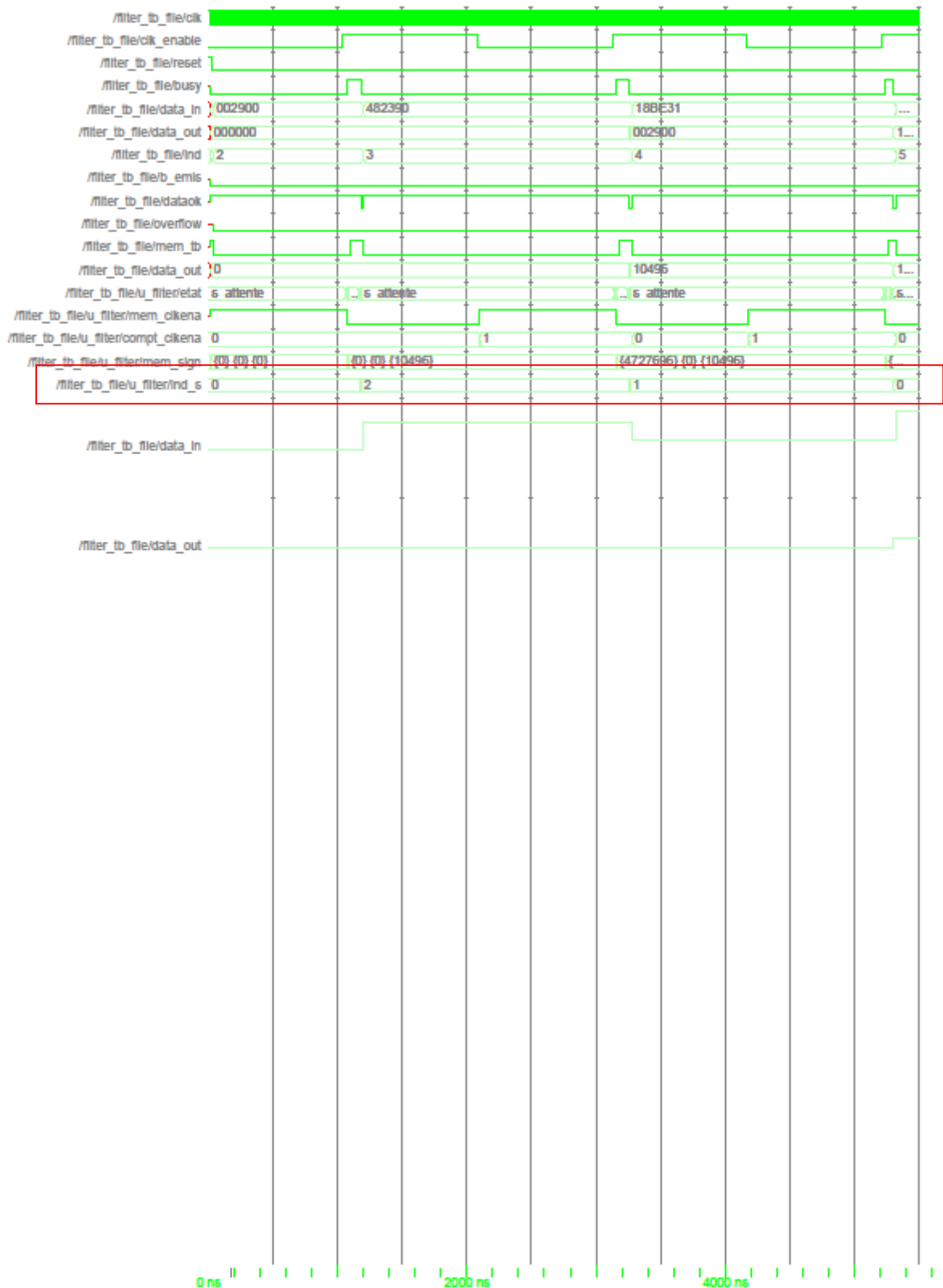
Entity: filter_tb_file_sinusk Architecture: test Date: Thu Apr 01 18:02:27 Paris, Madrid (heure d'été) 2010 Row: 1 Page: 1

6-4 Signal triangulaire



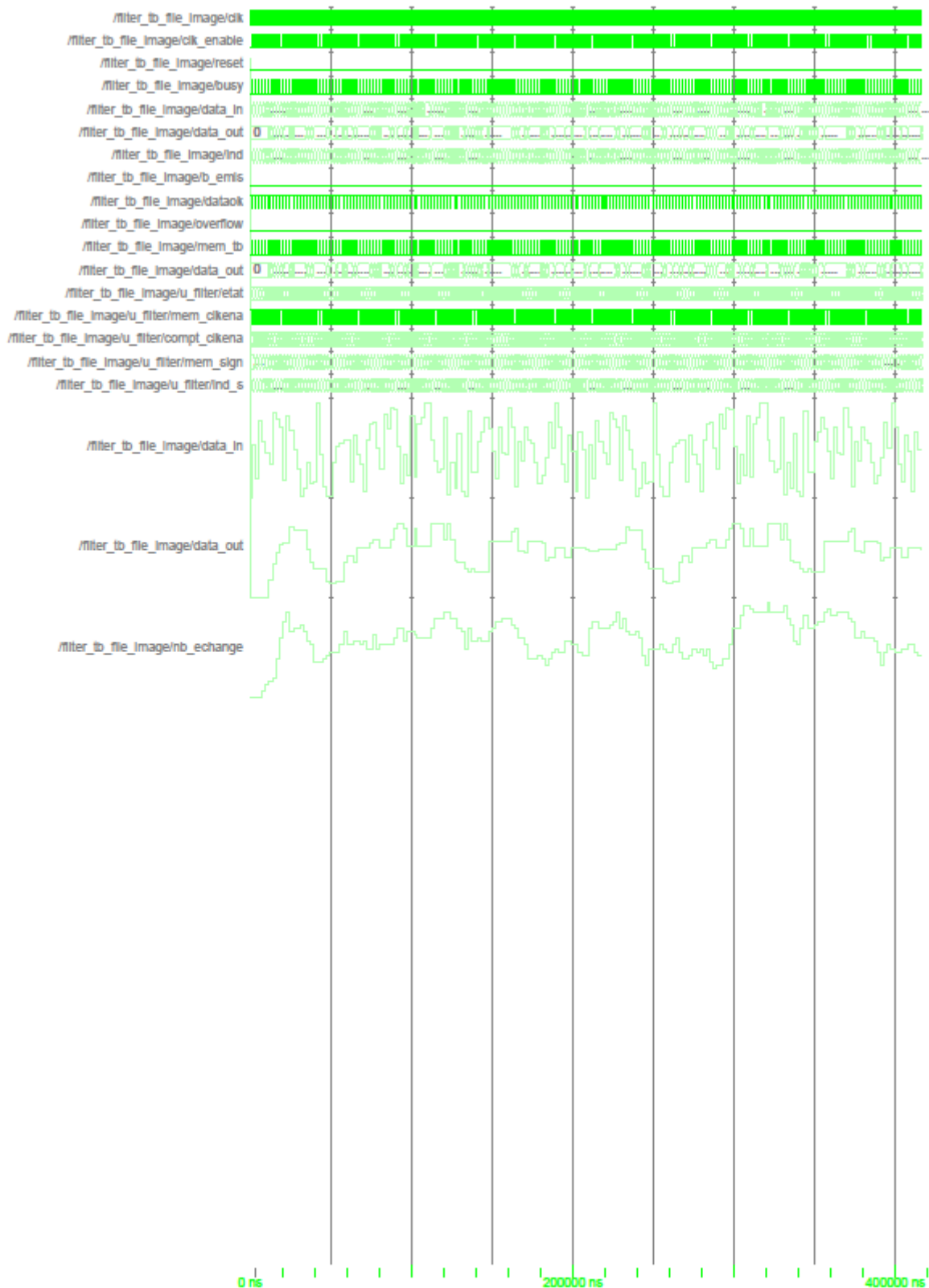
Entity:mlier_tb_file_sinus Architecture:test Date: Tue Apr 06 11:03:02 Paris, Madrid (heure d'été) 2010 Row: 1 Page: 1

6-5 Evolution des indices de stockage mémoire de signaux (profondeur 3)



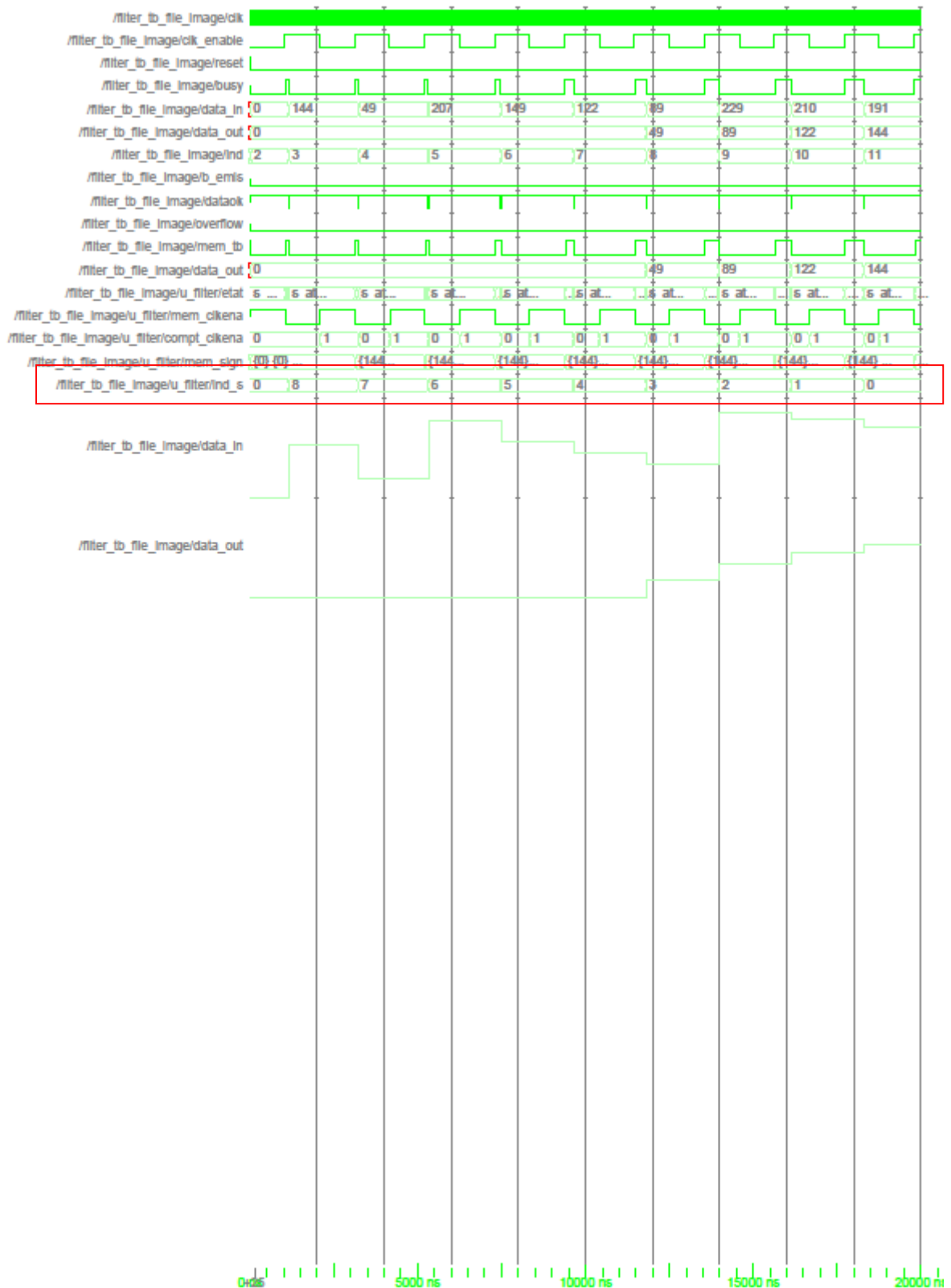
Entity: filter_tb_file Architecture: test Date: Thu Apr 01 08:23:32 Paris, Madrid (heure d'été) 2010 Row: 1 Page: 1

6-6 Niveau de luminance de pixels



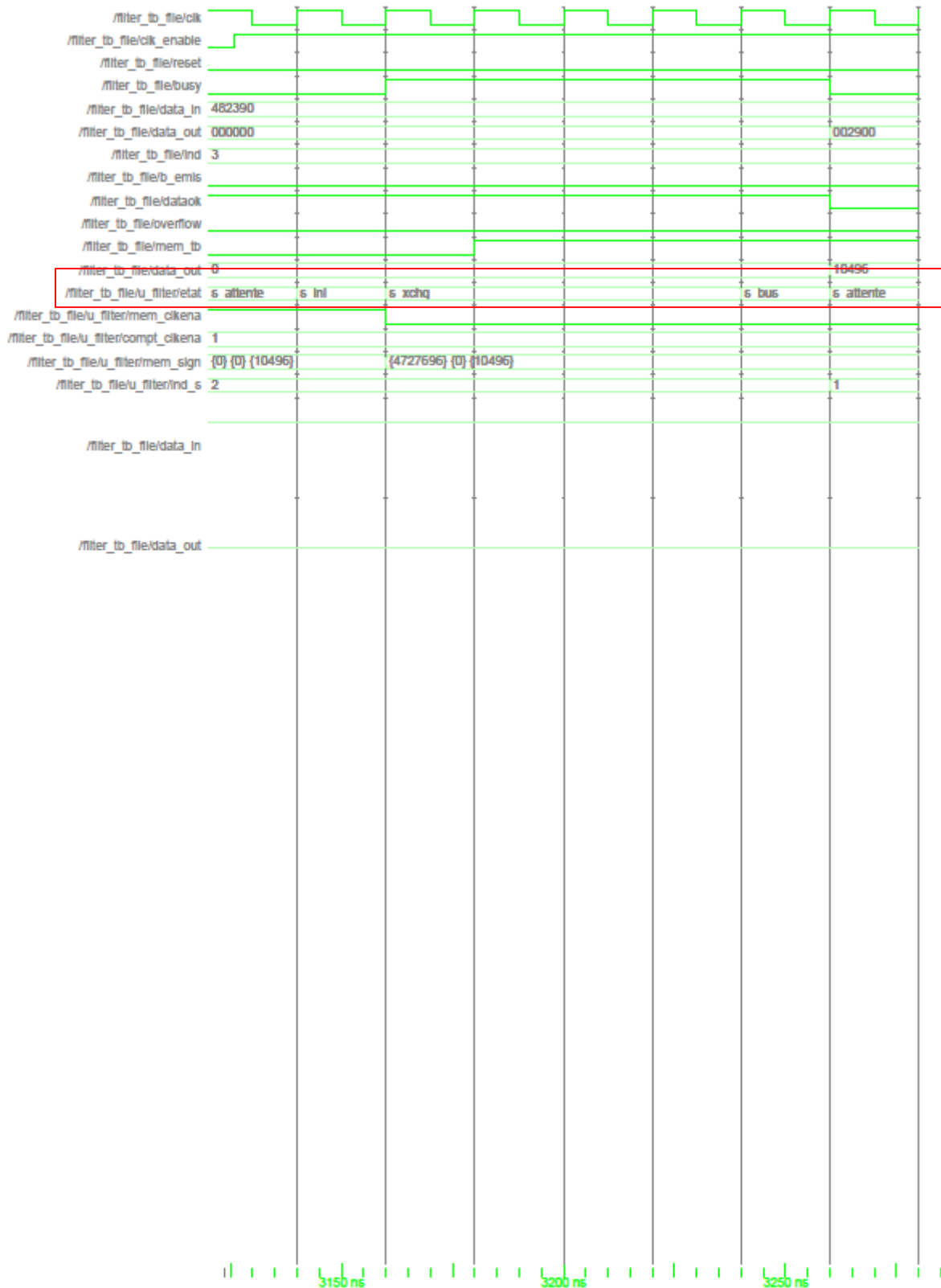
Entity:filter_tb_file_image Architecture:test Date: Thu Apr 01 17:47:44 Paris, Madrid (heure d'été) 2010 Row: 1 Page: 1

6-7 Evolution des indices de stockage mémoire de luminance de pixels (profondeur 9)

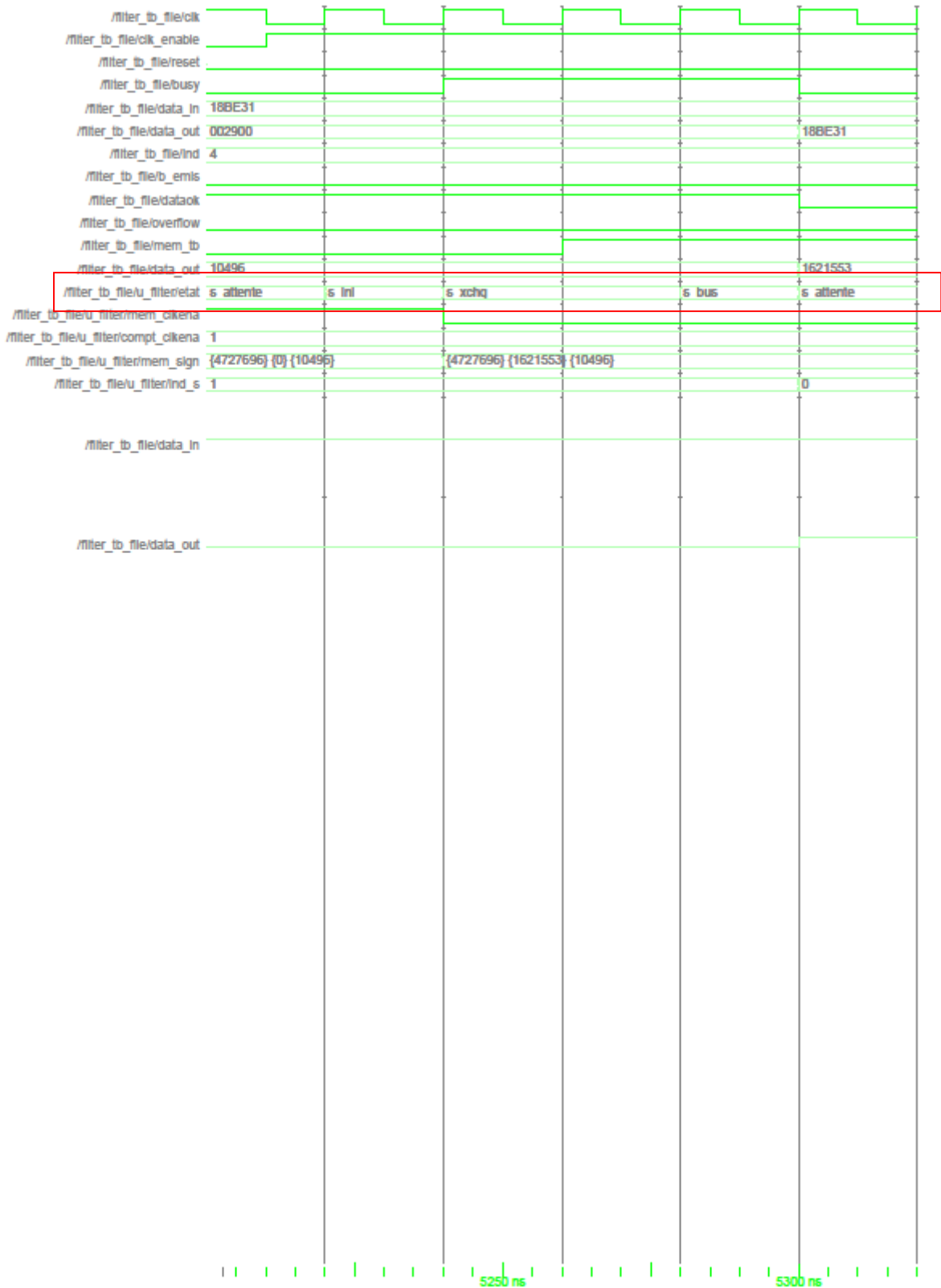


Entity:mfilter_tb_file_image Architecture:test Date: Thu Apr 01 06:34:29 Paris, Madrid (heure d'été) 2010 Row: 1 Page: 1

6-8 Evolution des états



Entity: filter_tb_file Architecture: test Date: Thu Apr 01 08:28:53 Paris, Madrid (heure d'été) 2010 Row: 1 Page: 1



Entity:mter_tb_file Architecture:test Date: Thu Apr 01 08:31:28 Paris, Madrid (heure d'0t0) 2010 Row: 1 Page: 1