

MA411-TP Optimisation et Analyse Statistique
sur HPADS

DESGRANGES Jean-Philippe DAVID Rajendra
SCHAPIRA Boris

le 11 Octobre 2005

Première partie

Optimisation sur HPADS

1 Configuration des éléments passifs

On paramètre chaque composant du filtre pour qu'il réponde aux caractéristiques suivantes :

- Sa valeur doit pouvoir être optimisée par le moteur de simulation (`Optimisation Status Enable`)
- Si c'est une impédance, sa valeur doit être comprise entre 0.2nH et 20nH
- Si c'est une capacité, sa valeur doit être comprise entre 0.2pF et 10pF

Une telle configuration permet au moteur de simulation d'optimiser les valeurs des composants dans les domaines qui lui sont donnés.

2 Configuration des "Buts" ou "Goal" d'Optimisation

C'est par l'intermédiaire des blocs "Goal" qu'on explique au moteur ADS quels sont les critères à optimiser. Dans le cas du filtre passe-bas qui nous intéresse, nous allons demander à ADS d'optimiser selon deux critères : l'atténuation en bande passante (0Hz->0.8GHz) doit être inférieure à 0.2dB et supérieure à 50dB à partir de 1.3GHz

Le premier bloc "Goal" correspond à l'optimisation en bande passante, et se définit comme tel :

- `Expr="dB(S(2,1))"` : Optimise le gain en puissance du filtre
- `SimInstanceName="SP1"` : Nom du moteur d'optimisation
- `Min=-0.2` : minimum de gain pour la bande passante
- `Max=0` : maximum de gain pour la bande passante
- `Weight=1` : On peut appliquer une pondération
- `RangeVar[1]="freq"` : La variable de progression est la fréquence
- `RangeMin[1]=0 MHz` : minimum de fréquence pour la bande passante
- `RangeMax[1]=0.8 GHz` : maximum de fréquence pour la bande passante

On paramètre de même le second "Goal" :

- `Expr="dB(S(2,1))"` : Optimise le gain en puissance du filtre
- `SimInstanceName="SP1"` : Nom du moteur d'optimisation
- `Min=` : pas de minimum de gain pour la bande passante
- `Max=-50` : maximum de gain pour la bande passante (en réalité pas de max)
- `Weight=1` : On peut appliquer une pondération
- `RangeVar[1]="freq"` : La variable de progression est la fréquence
- `RangeMin[1]=1.3 MHz` : minimum de fréquence pour la bande passante
- `RangeMax[1]=5 GHz` : maximum de fréquence pour la bande passante

3 Configuration du moteur d'Optimisation

Le moteur d'optimisation de la fenêtre schématique doit se configurer selon de nombreux paramètres. On règle ces paramètres comme demandé dans l'énoncé et testons différents types d'optimisation comme la méthode du Gradient par les Moindres Carrés, la méthode du Gradient par la méthode du Minimax ou encore un algorithme d'optimisation génétique (cf Annexes en fin de rapport).

L'optimiseur propose pour chaque optimisation un jeu de valeurs pour les composants, que l'on réactualise au niveau schématique. A ce stade, on peut supposer

qu'on dispose de valeurs optimales pour les éléments du circuit. Reste à évaluer si de telles valeurs sont statistiquement fiables en terme de performances et de sensibilité aux variations.

Deuxième partie

Performance et Sensibilité du filtre

On va chercher à étudier les performances du filtre et sa sensibilité aux variations de valeur des éléments (car il est très rare qu'un composant corresponde exactement à la valeur nominale).

4 Analyse statistique

Dans le fichier dédié à l'analyse du Yield, nous remplaçons les valeurs des composants par nos valeurs optimales. Les performances du filtre correspondent bien au gabarit suggéré.

4.1 Configuration des éléments

On effectue désormais une analyse de rendement du filtre afin d'évaluer sa résistance à la dispersion des composants. Pour cela, on commence par donner à chaque composant une tolérance gaussienne de 5% pour les capacités et de 10% pour les inductances.

4.2 Configuration des "Buts" ou "YieldSpec" d'Analyse

C'est par l'intermédiaire des blocs "YieldSpec" qu'on explique au moteur ADS quels sont les critères à analyser. Dans le cas du filtre passe-bas qui nous intéresse, nous allons demander à ADS d'analyser selon deux critères : l'atténuation en bande passante (0Hz->0.8GHz) tolérée doit être inférieure à 0.4dB et supérieure à 40dB à partir de 1.3GHz. Ces critères sont moins exigeants que les critères d'optimisation. Ils définissent jusqu'à quel point un jeu de valeurs de composant peut être jugé viable.

Le premier bloc "YieldSpec" correspond à l'optimisation en bande passante, et se définit comme tel :

- Expr="dB(S(2,1))" : Analyse le gain en puissance du filtre
- SimInstanceName="SP1" : Nom du moteur d'analyse
- Min=-0.4 : minimum de gain pour la bande passante
- Max=0 : maximum de gain pour la bande passante
- Weight=1 : On peut appliquer une pondération
- RangeVar[1]="freq" : La variable de progression est la fréquence
- RangeMin[1]=0 MHz : minimum de fréquence pour la bande passante
- RangeMax[1]=0.8 GHz : maximum de fréquence pour la bande passante

On paramètre de même le second "YieldSpec" :

- Expr="dB(S(2,1))" : Analyse le gain en puissance du filtre
- SimInstanceName="SP1" : Nom du moteur d'Analyse
- Min=- : pas de minimum de gain pour la bande passante
- Max=-50 : maximum de gain pour la bande passante
- Weight=1 : On peut appliquer une pondération
- RangeVar[1]="freq" : La variable de progression est la fréquence
- RangeMin[1]=1.3 GHz : minimum de fréquence pour la bande passante

– `RangeMax[1]=5 GHz` : maximum de fréquence pour la bande passante

On paramètre également le moteur d'analyse statistique `Yield`, notamment en lui demandant un certain nombre d'itérations : `NumIters=500`

On lance l'analyse statistique. Dans la fenêtre `Status`, on peut lire le nombre de circuits testés (500, comme demandé au `Yield`) et sur ces 500 circuits, le nombre de circuits qui satisfont le gabarit moins exigeant défini par les `YieldSpec`. Le pourcentage de circuits corrects s'appelle le *yield*.

On observe un *yield* relativement faible, de l'ordre de 15%.

On décide d'observer les histogrammes de performances. Ils affichent le nombre de circuits satisfaisant le gabarit en fonction de l'éloignement à la valeur primaire.

En relâchant suffisamment le gabarit défini par les `YieldSpec`, on obtient un *yield* de 50%. Cependant les histogrammes obtenus sont complètement décentrés et le gabarit n'a plus rien à voir avec le gabarit d'origine.

5 Design Centering

Afin de réoptimiser ce circuit, dans le but de garantir un bon *yield*, on utilise un autre moteur de simulation : le *design centering* par la méthode Monté Carlo. L'optimisation est effectuée à partir des `YieldSpec`, c'est pourquoi on leur redonne les valeurs définies en 4.2 (moins éloignées du gabarit réel que celles définies en fin de 4. pour obtenir un *yield* correct).

On remet à jour les valeurs des composants ainsi calculée pour répondre à un objectif de rendement optimal. Une analyse de rendement nous renvoie alors un *Yield* de l'ordre de 40% avec des histogrammes plus centrés qu'auparavant (bonne répartition des circuits face à la dispersion des éléments qui les composent). L'optimisation dans un objectif de rendement remplit bien son rôle.