Rapport d'avancement Post-Doctoral n°3

CNES
IRCOM
NMDG Engineering

CALIBRAGE DU SYSTEME DE MESURE LSNA COUPLE A DES SONDES HAUTES-IMPEDANCES

MESURES DE FORMES D'ONDES TEMPORELLES SUR DES NŒUDS INTERNES DE CIRCUITS MMIC ET HYBRIDES POUR L'ANALYSE DE FIABILITE

Tibault Reveyrand

Juillet 2003

« I've seen things you people wouldn't believe.
Attack ships on fire off the shoulder of Orion.
I watched c-beams glitter in the dark near Tanhauser Gate.
All those moments will be lost in time, like tears in rain. »

Roy Batty in « Blade Runner »
Ridley Scott - 1981

SOMMAIRE

SOMMAIRE 3

INTRODUCTION 5

TRACABILITE DE L'ETALONNAGE DES SONDES HAUTES IMPEDANCES 6

Rappel sur l'hypothèse émise 6

Poursuite des investigations 7
Inconsistance sporadique sur les mesures d'amplitudes absolues 8
Procédure d'étalonnage d'une sonde HIP 14
Etalonnage des sondes HIP sur le pied de test Wiltron 15
Etalonnage des sondes HIP sous pointes 19

Vérifier la cohérence de l'étalonnage 20
Mesure du maillage de standards RF 20
Analyse du maillage sous 'Scilab' 22
Comparaison des mesures entre les sondes GSG et les sondes HIP 24
Déduction de la constante de propagation à partir des mesures expérimentales 25
Mesurer la tension, c'est bien. Avoir le courant, c'est mieux ! 26

ETALONNAGE DU SYSTEME DE MESURE LSNA COUPLE AUX SONDES HAUTES IMPEDANCES 30

Introduction 30

Aspects théoriques 31
La matrice d'erreur conventionnelle du LSNA 31
Méthodes 1 et 2 : modèle d'erreur simple (fonction de transfert en tension) 33
Méthode 3 : modèle d'erreur étendu (fonction de transfert en onde de propagation) 35
Synthèse des 3 configurations « Calibrage / Mesure » proposées 39

Aspects pragmatiques sur la correction des ondes mesurées par le LSNA 40
Fichier de calibrage standard du LSNA 41
Correction « temps réel » des mesures par le LSNA 43
Génération d'un fichier de calibrage hybride pour le LSNA 44
Avantages des méthodes utilisées 45
Comparaison des matrices d'erreurs issues des méthodes 2 et 3 46

Utilisation sous 'Mathematica' 49
Méthode 1 49
Méthode 2 50
Méthode 3 50

MESURES APPLICATIVES **52**

Mesurer des ondes de tensions internes à un circuit MMIC ou hybride **52**
1. Initialisation du LSNA 52
2. Calibrage du LSNA en LRRM 53
3. Mesure avec le LSNA calibré en LRRM 53
4. Etalonnage des sondes HIPs 54
5. Mesure via sonde HIP sur le substrat d'etalonnage 54
6. Mesures via sonde HIP sur un circuit MMIC 55

Vérification d'une topologie optimale pour un circuit MMIC **57**
Présentation de l'amplificateur 57
Banc de mesure 57
Résultats de mesures 59

Vérification de phénomènes dynamiques complexes : la stabilité non-linéaire de circuits MMIC **62**
Généralités sur les études de stabilité non-linéaire 62
Présentation de l'amplificateur bande X 68
Banc de mesure 70
Analyse comportementale de l'amplificateur 70
Mesures effectuées avec le système LSNA+HIP 74

CONCLUSION **79**

BIBLIOGRAPHIE **80**

ANNEXES **82**

Modifications de la procédure d'étalonnage LRRM sous l'environnement 'Mathematica' **82**

Programme 'Scilab' : Visualisation et traitement de maillages de lignes mesurés par sonde HIP **97**

Module « HIP` » sous 'Mathematica' **102**

Album de photos-souvenirs **110**

INTRODUCTION

Ce rapport est le troisième et dernier consacré au projet mis en commun entre le CNES, l'IRCOM et Agilent NMDG (devenu NMDG Engineering) sur la mise en place d'un système de mesure opérationnel permettant l'extraction de formes d'ondes temporelles multi-harmoniques par sondes hautes impédances placées sur des nœuds internes de circuits MMIC et hybrides.
Il complète donc les investigations répertoriées lors des deux premiers rapports et propose au final une solution simple et opérationnelle.

Ce rapport est divisé en trois chapitres.

Dans un premier temps, la cohérence des mesures et de l'étalonnage, décrit lors des précédents rapports, sera analysée. Pour cela, un nouvel outil a été développé sous le logiciel gratuit 'Scilab'.

Un second chapitre traite de l'étalonnage du système complet (LSNA associé aux sondes HIP). Il présentera aussi bien les aspects théoriques que pratique. Au final, les procédures d'étalonnage sont insérées dans l'environnement 'Mathematica' du LSNA. Le système est donc complètement opérationnel. Nous verrons que 3 procédures d'étalonnage sont envisageables suivant le nombre de paramètres physiques connus par le manipulateur.

Enfin, un troisième et dernier chapitre va traiter de mesures applicatives. Des thèmes fondamentaux dans la conception de circuits RF optimum peuvent s'enrichir de l'instrument de mesure qui a été développé, dont entre autre l'étude de la stabilité. Ces thèmes seront donc abordés à travers des exemples de mesures expérimentales ayant été faites sur des conceptions HBT CNES/IRCOM.

TRACABILITE DE L'ETALONNAGE DES SONDES HAUTES IMPEDANCES

Ce chapitre reprend le principe d'étalonnage des sondes hautes impédances (HIP) qui a été décris dans le précédent rapport.

Dans un premier paragraphe, on rappellera le principe d'étalonnage des HIP et notamment l'hypothèse considérée lors de l'étalonnage d'une sonde « haute impédance ».

Un second paragraphe va traiter de la poursuite des investigations qui ont été faites sur l'étalonnage des HIP.
En premier lieu, une incohérence existant sur le système LSNA utilisé (NMRC) a été détectée. Les fonctions 'Mathematica' utilisées pour mesurer à l'aide des sondes HIP considérerons les erreurs relatives à cette incohérence afin de minimiser ses effets.
Suite à quoi, le processus d'étalonnage LRRM du LSNA sera présenté. Quelques modifications du code initial 'Mathematica' ont permis d'utiliser cette procédure de calibrage aussi bien sur une table sous pointe classique que sur un pied de test.

Enfin, un troisième paragraphe proposera de nombreuses mesures réalisées sur des dispositifs passifs. Le maillage spatial de ces dispositifs (principalement des lignes) permet leur analyse (constante de propagation, ondes de propagation) et de vérifier la cohérence de l'étalonnage mis en place. Des limites vont être mises en évidence sur les possibilités de mesure du courant avec les sondes hautes impédances.

Rappel sur l'hypothèse émise

Des travaux antérieurs réalisés par J.C.M. Hwang et le NIST autour de sondes hautes impédances ont déjà été présentés lors du premier rapport d'avancement. L'hypothèse considérée pour permettre un étalonnage simple, rapide et efficace avec le LSNA provient d'un calcul de de-embedding publié par P. Kabos du NIST [1]. Cette hypothèse s'exprime selon une fonction de transfert telle que :

$$V_{\text{HIP plan DUT}}(f) = \widetilde{K}(f).V_{\text{mesuré}}(f)$$

Le $V_{\text{mesuré}}$ est dans [1] une mesure calibrée faite par un oscilloscope dans le plan coaxial. Pour un appareil non calibré, la relation reste valable compte tenu du modèle d'erreur de l'oscilloscope : le coefficient de réflexion de l'oscilloscope intervient alors dans l'expression de $\widetilde{K}(f)$. Les canaux du 'Downconverter Box' peuvent être considérés comme des oscilloscopes RF. Cela justifie l'emploi de cette hypothèse.

D'un point de vue pratique, $\tilde{K}(f)$ correspond à une fonction de transfert en tension de la sonde HIP, d'un câble RF, d'un DC Block (car la sonde HIP peut mesurer le DC : si le continu est présent en entrée de 'Downconverter Box', il peut l'endommager) et du système de réception du LSNA (conversion de fréquence par mélange harmonique, échantillonnage numérique de la tension FI).

La procédure d'étalonnage classique d'une sonde HIP requière de nombreuses mesures au VNA afin de déduire mathématiquement la matrice [T] de la sonde HIP [2].
L'énorme avantage du LSNA est qu'il propose par défaut une procédure de calibrage rigoureuse nous donnant accès aux valeurs de v et i dans les plans de mesure. Calibrer un système pourvu de sonde HIP devient extrêmement simple.

Le système de mesure LSNA couplé aux sondes « hautes impédances » est structuré tel que le port 2 reste inchangé par rapport à l'utilisation classique du LSNA (utilisation du « test-set » conventionnel) mais le port 1 est déconnecté : les canaux 1 et 2 du « Downconverter Box » ne sont plus reliés au « test-set » mais aux sondes HIP.
L'étalonnage d'un tel système comporte alors les étapes suivantes :

1. **Etalonnage du LSNA classique sous pointe** (procédure LRRM)
2. **Connexion des sondes HIP aux canaux 1 et 2 du « downconverter box ».**
3. **Etalonnage des sondes HIP.** Pour cela la mesure via le port 2 du LSNA permet la mesure corrigée de v et de i dans le plan de référence issue de l'étalonnage LRRM. Ce plan de référence devient alors notre étalon en tension (amplitude et phase) qui nous permet d'identifier la fonction de transfert $\tilde{K}(f)$ en faisant un rapport entre les tensions mesurées (référence) et les tension issue des HIP pour chacune des fréquences de la grille d'étalonnage LRRM du LSNA.

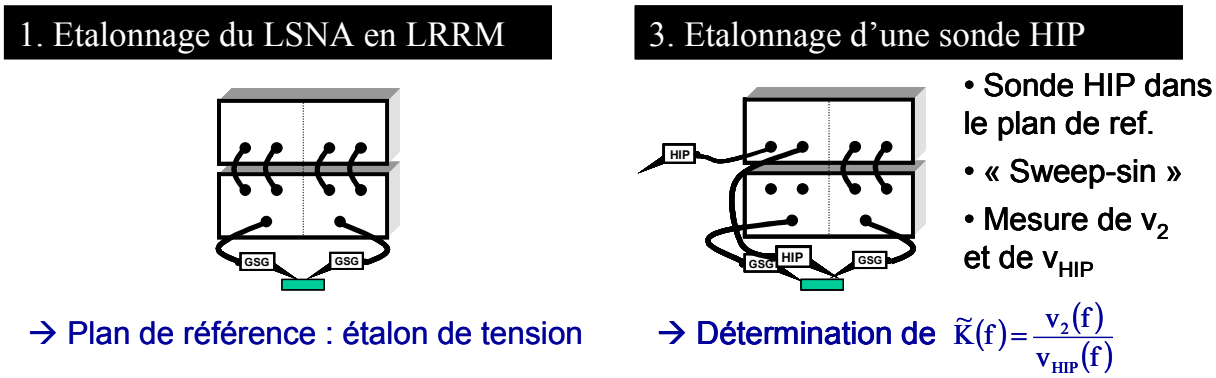


Fig. 1 - Synoptiques des étapes 1 et 3 pour calibrer le système de mesure LSNA+HIP

Poursuite des investigations

Ce paragraphe met en place les fondations de l'étalonnage des sondes HIP. Dans un premier temps nous décrirons une erreur hardware présente dans le système LSNA utilisé (système NMRC) : une perte de référence de phase 10 MHz qui s'opère de façon sporadique. Il conviendra de comprendre ce phénomène pour passer outre et s'assurer que le calibrage du LSNA en configuration normale est rigoureux. Cet étalonnage va ensuite être notre point de départ concernant la détermination de $\tilde{K}(f)$.

Lors du précédent rapport nous avons pu calibrer les sondes HIP sur une ligne dotée de connecteurs SMA à ses extrémités. Cependant une inconsistance de taille était présente : l'absence (lors de l'étalonnage standard du LSNA : en SOLT) d'un plan de référence accessible par la sonde HIP. Afin d'éliminer cette inconsistance, l'étalonnage classique du LSNA se fait désormais en LRRM sous pointes (banc Cascade) ou avec un pied de Test (UTF Wiltron).
Ainsi la procédure d'étalonnage du LSNA en LRRM sera décrite dans ce paragraphe ainsi que les modifications qui y ont étés apportées afin de permettre un étalonnage sous différents environnements : c'est à dire aussi bien sous pointes (déjà implémenté) qu'avec le pied de test Wiltron (sujet des modifications du code).

La détermination du rapport $\widetilde{K}(f)$, entre autres, se fait par l'appel d'une fonction appartenant au module « HIP` ». Ce module, dont le listing est présent en annexes de ce rapport, propose différentes fonctions qui seront appelées dans le cadre de l'utilisation des sondes HIP. Au final ce module s'intègre parfaitement dans l'environnement 'Mathematica' du LSNA et permet à l'utilisateur de manipuler des formes d'ondes corrigées exactement comme si elles provenaient d'un « test-set » classique.

Inconsistance sporadique sur les mesures d'amplitudes absolues

Mise à défaut de la reproductibilité des mesures

Suite à plusieurs investigations sur les tensions mesurées le long de lignes sous test, il est apparu une incohérence concernant l'exactitude de l'étalonnage des sondes haute-impédance. La reproductibilité des mesures corrigées via sondes hautes impédances était mise à défaut.

Nous avons vu dans le précédent rapport que cette fonction de transfert est déterminée telle que : $\widetilde{K}(f) = \frac{V_2(f)}{R_1(f)}$ où V_2 est la tension dans le plan de référence et R_1 la tension brute (non corrigée) issue de la sonde HIP. La mesure des composantes de ce rapport se fait de façon simultanée pour une valeur de f fixée. Lors de l'étalonnage, un balayage de f est opéré. La reproductibilité de $\widetilde{K}(f)$ est bonne.
En revanche, l'obtention de la tension corrigée issue de la sonde HIP est peu reproductible.
Cela s'explique par le fait que cette tension corrigée s'exprime par un produit de quantités qui ne sont pas mesurées simultanément ($\widetilde{K}(f)$ et $V_{HIP\ brute}(f)$) tel que : $V_{HIP\ corrigée}(f) = \widetilde{K}(f).V_{HIP\ brute}(f)$.

Revenons dans une configuration classique du LSNA : c'est à dire à une utilisation conventionnelle du test-set.

On a pu constater qu'en réalisant un grand nombre de mesures consécutives, « a priori » identiques, les réponses du « downconverter box » diffèrent. Pour un même signal RF, les signaux BF en sortie du « Downconverter Box » (c'est à dire R1, R2, R3 et R4 obtenus

simultanément) restent cohérents entre eux. Les phases relatives de ces signaux restent cohérentes entre 2 mesures successives mais pas l'amplitude.

Pour illustrer ce phénomène, on se propose de faire un balayage en fréquence de type « Sweep Sin » à puissance constante lorsque le LSNA est calibré en connecteurs SOLT et qu'il est dans une configuration de connexion directe. La phase de référence est la phase de V_2.

```
In[7]:= V2freqsweep := Table[
          SetSourceFreq[(i * Fundamental)];
          NNMSProcessing`PhaseReferenceFrequency = i;
          Pause;
          Measure;
          (v2[[i + 1]]), {i, 1, 20, 1}
         ];

In[8]:= V1freqsweep := Table[
          SetSourceFreq[(i * Fundamental)];
          NNMSProcessing`PhaseReferenceFrequency = i;
          Pause;
          Measure;
          (v1[[i + 1]]), {i, 1, 20, 1}
         ];
```

On réalise ce « Sweep Sin » 10 fois afin de vérifier la reproductibilité des quantités mesurées. On représente alors les valeurs obtenues pour V1 et V2 en fonction de la fréquence et pour chacune des 10 mesures. Théoriquement, on doit avoir une courbe unique et décroissante traduisant les pertes produite par le test-set entre le niveau constant de la source RF et le plan de mesure.
Ci dessous sont représentées les courbes pour les mesures de modules Abs[V1] et Abs[V2].

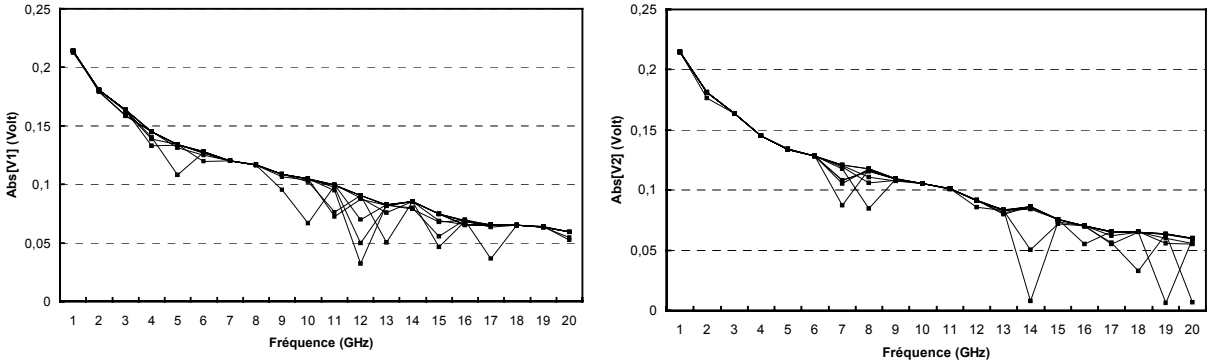


Fig. 2 - Reproductibilité des mesures : Module de V1 (à gauche) et de V2 (à droite) lors de 10 « Sweep-sin » consécutifs

On constate que de façon sporadique, le module de la tension mesuré est inférieur à la vraie valeur du module de la tension présente dans le plan de référence. Dans les courbes représentées, les mesures de V1 et de V2 ne sont pas simultanées.

Si ces mesures étaient simultanées, on observerait une même erreur sur V1 et sur V2 au même endroit. La figure suivante en est la preuve : elle représente 100 mesures consécutives à une même fréquences (pas de « sweep-sin »). Les V1 et V2 ont alors été obtenus de façon simultanée (la phase de V1 est la phase de référence).

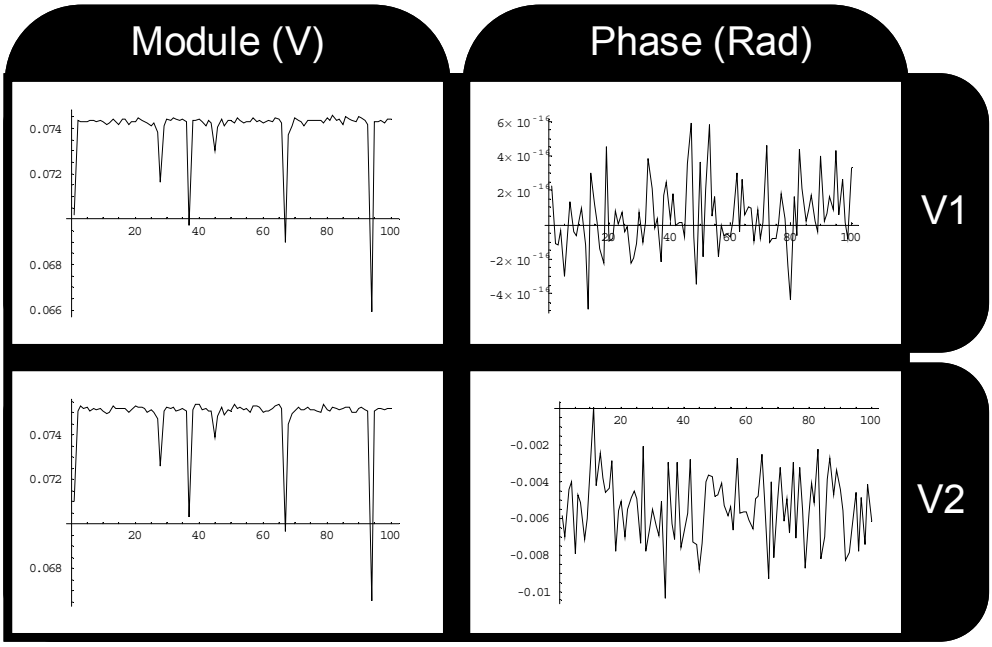


Fig. 3 - 100 mesures successives pour f = 15 * fondamental et un niveau constant. V1 et V2 sont simultanés

On s'intéresse maintenant à la phase de V_1 lors du « Sweep-sin ».Rappelons que la phase de V_2 est nulle car nous l'avons choisie comme phase de référence. Les résultats des 10 « Sweep-sin » sont représentés ci-dessous. On peut constater l'existence d'une erreur systématique sur la phase de V_1, mais également d'une erreur stochastique similaire à celle observée sur l'amplitude.

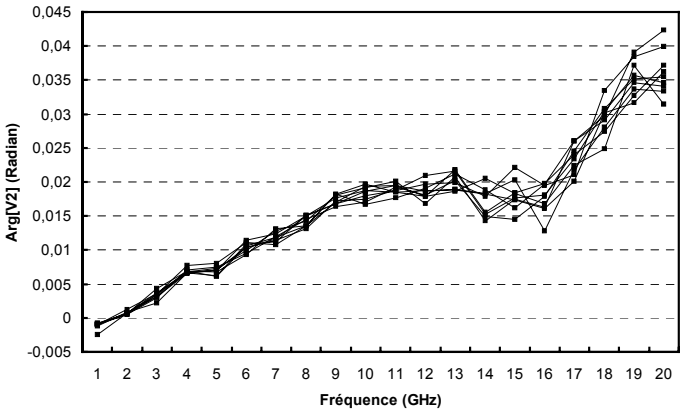


Fig. 4 - Reproductibilité des mesures : Phase de V1 lors de 10 « Sweep-sin » consécutifs. La phase de V2 est alors nulle (phase de référence).

En ce qui concerne l'erreur systématique, la procédure d'étalonnage du LSNA (ici SOLT) est à mettre en cause. Le SOLT n'est pas en soit une bonne procédure de calibrage compte tenu du nombre élevé d'hypothèses à prendre en compte. Par ailleurs, il a été déjà observé par NMDG, une inconsistance dans l'étape du « Thru » lors de l'étalonnage du LSNA en SOLT : la réponse n'étant pas la même suivant le sens de propagation de l'onde RF.
L'erreur stochastique, alors qu'elle est primordiale lors de l'analyse des modules est négligeable en ce qui concerne la phase : on observe un écart inférieur à 0.6° entre les valeurs les plus divergentes.

On s'intéresse maintenant au rapport V2/V1. L'erreur occasionnée lors d'un tel rapport peut être considérable. Cette erreur peut avoir lieu lors de la détermination de $\widetilde{K}(f)$ lorsque les quantité R1 et V2 ne sont pas mesurées simultanément, lors de la déduction de la tension mesurée par la HIP et corrigée par le rapport $\widetilde{K}(f)$ lorsque la quantité R1 mesurée est erronée et enfin lors de la procédure d'étalonnage standard du LSNA : ce qui est extrêmement gênant compte tenu que l'étalonnage de la HIP est référencé par rapport au LSNA calibré.
La figure suivante propose les rapports V2/V1 des données précédemment présentées. Idéalement, ce rapport est égal à 1. C'est ce que l'on obtient (courbe identifiée par les cercles blancs) en réalisant un certain nombre de mesures pour V1 et V2 et en ne considérant que les mesures dont le module soit maximal.

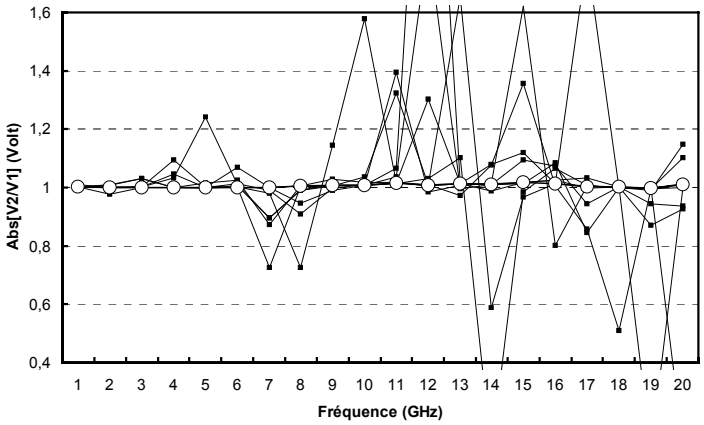


Fig. 5 - Reproductibilité des mesures : Module du rapport V2/V1.

Des mesures simultanées relatives restent correctes mais pas les mesures absolues. Un tel constat laisse supposer une perte de synchronisation du downconverter box avec le 10 MHz de référence de façon sporadique. Un tel phénomène a déjà été observé sur des travaux autour d'un VNA en mode récepteur [3].

Nouvelle fonction « Sweep-sin » pour le LSNA

De nombreuses fonctions utilisent un balayage fréquentiel de type « Sweep-sin » : l'étalonnage des sondes HIP, la mesure par maillage de dispositifs passifs et même l'étalonnage standard relatif du LSNA (SOLT et LRRM). Les procédures « sweep-sin » utilisées dans le cadre des travaux sur les sondes hautes impédances sont faites pour éviter les phénomènes stochastiques. Elles consistent à réaliser plusieurs mesures et à conserver celles dont le module est le plus grand.

Un module « HIP` » a été développé pour l'environnement 'Mathematica' du LSNA. Ce module regroupe toute les nouvelles fonctions qui sont utilisées couramment dans le cadre des travaux, ici menés, sur les sondes hautes impédances. Les procédures de type « Sweep-sin » inclues dans le module « HIP` » utilisent la méthode décrite précédemment : c'est à dire un tri parmi plusieurs mesures sur le critère du module.

Cependant une procédure de « Sweep-sin » est utilisée lors de l'étalonnage standard du LSNA et cette procédure n'a pas été modifiée afin de ne pas perturber le code initial 'Mathematica' du LSNA. Les problèmes d'erreur d'amplitude n'ont aucune incidence sur l'étalonnage relatif de type VNA, ni sur la mesure de phases (HPR), mais influent lors de la

phase d'étalonnage avec le bolomètre. Il convient de vérifier l'étalonnage standard du LSNA avant de poursuivre vers l'étalonnage des sondes HIP.

Vérifier l'étalonnage intrasèque du LSNA avant d'étalonner les sondes HIP

La phase d'étalonnage conventionnelle du LSNA en présence du bolomètre est soumise aux incertitudes aléatoires. Une fois étalonné, le LSNA présente alors ces incertitudes comme des erreurs systématiques car elles sont intégrées dans les matrices d'étalonnage. Il convient donc de vérifier tout étalonnage après avoir sauvegardé les matrices d'erreurs correspondantes, et de les corriger si le besoin s'en fait sentir.

Une vérification simpliste mais néanmoins efficace consiste à réaliser le « Sweep-sin » précédemment décrit sur le standard « Thru ». On peut alors comparer plusieurs étalonnages. La fonction de « sweep-sin » intégrée dans le module « HIP` » est hautement répétable en sélectionnant plusieurs mesures à opérer avant le tri (4 ou 5 sont largement suffisantes).
Si le « sweep-sin » offre des résultats divergents en fonction de l'étalonnage sélectionné, il y a un problème dans les matrices d'erreurs. On peut alors obtenir une nouvelle matrice d'étalonnage du LSNA à partir de 2 ou 3 matrices erronées. Pour cela, la fonction ***LRRMDebugCalFile*** a été développée puis intégrée dans le module « HIP` ». Elle va faire un tri en fonction de la fréquence des éléments des matrices d'erreurs à conserver pour en faire une nouvelle matrice d'erreur exempte de problème stochastique.

HIP`

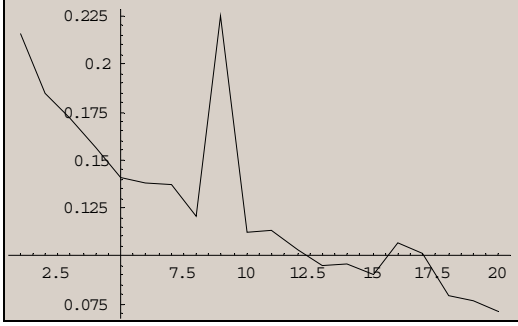
Cal2HIP	GenerateHybridCalFile
DefineGammaL1	GenerateHybridVNACalFile
DefineK	LRRMDebugCalFile
DefineK1	MakeSubMatrix1
DefineK2	MakeSubMatrix2
ExtractVNACalFile	MakeSubMatrix3
Freqsweep	MeasureHIP
Freqsweep0	Multimeasure
Freqsweep2	MultimeasureAll

Fig. 6 - Fonctions proposées par le module « HIP` » dans l'environnement 'Mathematica' du LSNA.

L'exemple suivant illustre le problème de ces calibrages inconsistants. Les problèmes sont mis en évidence grâce à un « sweep-sin » à puissance constante (fonction ***Freqsweep0***) avant d'être corrigés.

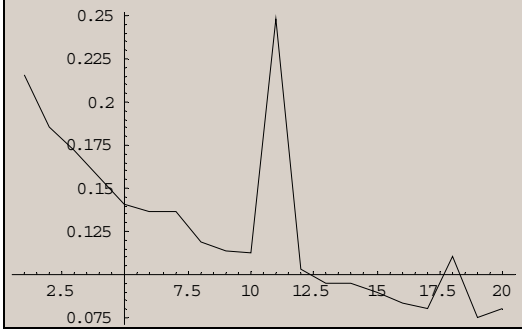
Test du premier calibrage par « Sweep-sin » sous 'Mathematica' :

```
LoadCal["D:\\Public\\Tibo\\CAL_numero_1.m"]

SetPhaseReferenceQuantity["v2"];

PhaseReferenceFrequency = 1;

SetSourcePower[0]; SetSourceFreq[Fundamental]; SetSourceOutput[On]; AutoRange;
ListPlot[Abs[Freqsweep0[20, 4, "v2", 0]], PlotJoined → True, PlotRange → All];
```



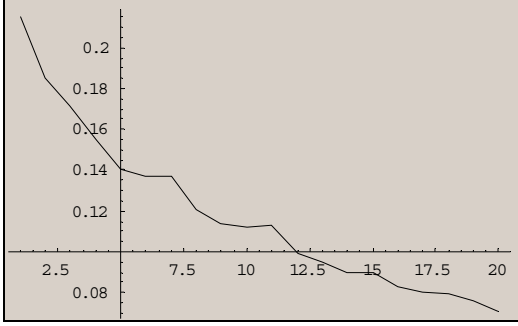
Test du second calibrage par « Sweep-sin » sous 'Mathematica' :

```
(* CAL n°2 *)

LoadCal["D:\\Public\\Tibo\\CAL_numero_2.m"]

SetPhaseReferenceQuantity["v2"];

PhaseReferenceFrequency = 1;

SetSourcePower[0]; SetSourceFreq[Fundamental]; SetSourceOutput[On]; AutoRange;
ListPlot[Abs[Freqsweep0[20, 4, "v2", 0]], PlotJoined → True, PlotRange → All];
```



En procédant à plusieurs appel de la fonction **LRRMDebugCalFile[]**, on peut trier les points fréquentiels corrects dans les matrices de calibrage afin de construire un fichier représentant une matrice d'erreur conventionnelle non erronée.

```
LoadCal["D:\\Public\\Tibo\\CAL_finale.m"]

SetSourcePower[0]; SetSourceFreq[Fundamental]; SetSourceOutput[On]; AutoRange;
ListPlot[Abs[Freqsweep0[20, 4, "v2", 0]], PlotJoined → True, PlotRange → All];
```



Le descriptif de cette fonction ainsi que de toutes celles inclues dans le module « HIP` » est donné à travers le listing de ce module en annexe de ce rapport.

Nota bene : Toutes les transformations de matrices de calibrage (*LRRMDebugCalFile[]* en est une ; nous en verrons d'autres plus tard dans ce rapport) **ne sont compatibles qu'avec des fichier de calibrage ne comportant qu'un seul user-setup**. L'utilisateur, s'il désire profiter de user-setup multiples (plusieurs configurations de LSNA dans le même calibrage), devra adapter le code 'Mathematica' en conséquence.

Une fois le LSNA correctement calibré, il constitue à travers le port 2 du test-set notre étalon en tension (module et phase). C'est à partir de cet étalon que nous pouvons calibrer une sonde HIP.

Procédure d'étalonnage d'une sonde HIP

La procédure d'étalonnage d'une sonde haute impédance est relativement simple, et le principe a été rappelé plus haut.
Dans un premier temps, on réalise un calibrage LRRM du LSNA standard. On vérifiera la cohérence de l'étalonnage via un sweep-sin comme cela a été expliqué précédemment. Une fois le calibrage correct, le port 2 du LSNA devient notre standard en tension (module et phase).

On déconnecte alors les canaux 1 et 2 du « downconverter box » afin de les relier aux sondes HIP. On peut évidemment utiliser des « switch » pour éviter toute déconnexion.
Si des mesures de continu sont attendues, on prendra soins de placer un « DC block » entre la sonde HIP et l'entrée du « downconverter box ». Notons que l'on peut utiliser un té de polarisation pour éviter d'injecter le continu sur l'entrée du « downconverter box ». Cette solution est extrêmement judicieuse car de la mesure issue de la sonde HIP, l'onde RF multi-harmonique est analysée par le « downconverter box » tandis que le continu peut être mesuré par un simple voltmètre connecté au té de polarisation.

La sonde HIP à calibrer doit ensuite être placée dans le plan de référence issue de l'étalonnage standard LRRM du LSNA. Cette manipulation se fait donc sur la ligne « Thru » 50Ω du kit d'étalonnage utilisé.

Une fois la sonde HIP en contact avec la ligne « Thru », un « Sweep-sin » est réalisé afin de déterminer la valeur de $\widetilde{K}(f)$. La détermination de cette fonction de transfert telle qu'elle est implémentée dans le module « Mathematica » fait varier la puissance de la Source RF en fonction de la fréquence. La plage de variation est celle utilisée lors des procédures de calibrage avec le « test-set ». Cela permet initialement de compenser les pertes induites entre autres par les coupleurs. Dans notre cas, avec les sondes hautes-impédances, cela permet d'améliorer la dynamique de mesure. Les fonctions 'Mathematica' qui déterminent ces variations de puissance en fonction de la fréquence sont issues d'une pré-caractérisation du système (du test-set). Ce sont :

- **MaxPowerPowerCal[f] :** Fournit les niveaux maximums acceptables par la source RF
- **RelCalPowerFunction[f][[1]]** : Fournit les pertes provoquées par le test-set.

Il est possible également de faire l'étalonnage d'une sonde HIP pour un niveau plus élevé de source RF. Cela peut provoquer un changement de position des atténuateurs programmables internes du « downconverter box » : ce changement fait suite à un « overload » detecté sur les ADC du VXI. L'étalonnage d'une sonde HIP peut se faire pour une position quelconque des atténuateurs relatifs au port 2. En revanche les atténuateurs reliés aux sondes HIP doivent impérativement être dans leurs position minimale (0 dB d'atténuation) nous verrons la raison de cette conditions dans le second chapitre de ce rapport.

Etalonnage des sondes HIP sur le pied de test Wiltron

Suite au précédent rapport, des investigations se sont poursuivies sur des lignes en connecteurs SMA. L'absence de plan de référence accessible par une sonde HIP a été un point déterminant pour utiliser une autre procédure d'étalonnage que la procédure en SOLT (présence de connecteurs) du LSNA. Le pied de test Wiltron est une première approche expérimentale pour étalonner le LSNA en LRRM avant de disposer de la station sous pointes 'Cascade'.

Présentation de l'UTF (Universal Test Fixture)

Le pied de test utilisé est le 'Wiltron 3680 Universal Test Fixture'. Il est associé à un kit de calibrage constitué de lignes de différentes longueurs et de standards classiques (Short, Open, Load). Ces standards sont intégrés dans des lignes semblables à la ligne Thru 50 Ohms. Ce qui revient à considérer lors de l'étalonnage un retard nul pour la ligne « Thru ».

L'étalonnage LRRM devient alors intéressant avec ce pied de test. En effet il est plus robuste qu'un étalonnage de type SOLT et conduit à un plan de référence qui est le milieu de la ligne « Thru »[1]. Ce plan de référence sera utilisé pour l'étalonnage des sondes HIP.

Etalonnage LRRM du LSNA

L'étalonnage absolu réalisé permettant la mesure de dispositifs dans le plan de connecteurs a déjà été présenté dans le rapport précédent. Cet étalonnage fait intervenir un étalonnage relatif (type VNA) dans le plan des connecteurs des plans de mesures, puis une mesure en puissance grâce à l'étalon de puissance (le bolomètre) et enfin un étalonnage en phase (dont l'étalon est la HPR). Les connecteurs du bolomètre et de la HPR sont en 2.4 mm. Pour des mesures en connecteurs en SMA ou K, l'emploi d'une transition s'impose mais ne perturbe pas la matrice d'erreur issue de l'étalonnage. En effet, une simple transition présente des pertes négligeables et un déphasage linéaire en fonction de la fréquence dans la bande d'intérêt.

Pour des dispositifs connectorisés, la procédure d'étalonnage est le SOLT. La figure 7 représente les différentes étapes lors de l'étalonnage du LSNA et la construction de la

[1] Avec un retard nul pour la ligne « Thru ».

matrices d'erreur théorique[2] entre les ondes mesurées et celles présentes dans le plan de référence.

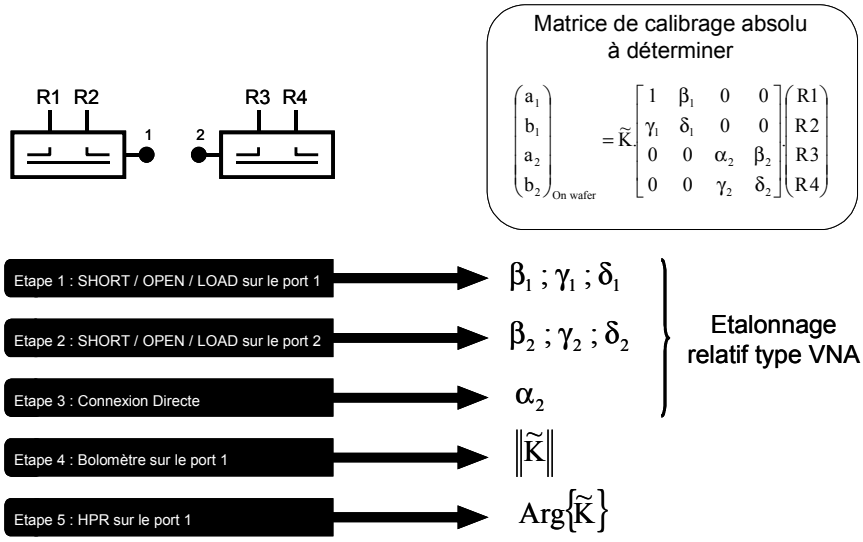


Fig. 7 - Illustration des étapes lors d'un étalonnage SOLT (en présence de connecteurs)

L'étalonnage de type LRRM est une procédure qualitativement meilleure que le SOLT ou que le TRL classique : en effet, elle fait intervenir moins d'hypothèses de départ sur les caractéristiques des étalons. Elle est utilisée pour étalonner le LSNA lors d'une configuration « On wafer ». Le plan de référence est alors le centre de la ligne « Thru » utilisée lors de l'étalonnage.
Le problème majeur dans ce type d'étalonnage est que les étalons d'amplitude et de phase sont valable dans un plan de référence en connecteur et non « on wafer ». L'étalonnage absolu du LSNA dans un milieu « on wafer » a néanmoins été rendue possible par Jan Verspecht [4].
Le principe pour étalonner de façon absolue le LSNA consiste à faire 2 étalonnages. Un étalonnage complet en connecteur (SOLT + Wattmètre + HPR) est réalisée afin d'avoir les ondes absolue dans le plan de référence coaxial. Un étalonnage LRRM permet d'avoir la matrice d'erreur relative dans le plan de référence sous pointes. La figure 8 représente ces différentes étapes lors d'un calibrage LRRM du LSNA.

Les valeurs absolues des ondes (amplitude et relation de phases) dans le plan de référence sous pointe , et donc la matrice d'erreur absolue, sont déduites en émettant l'hypothèse de réciprocité sur le quadripôle établissant la relation entre les 2 plans de références. En considérant la matrice [T] de ce quadripôle, cela revient à écrire la relation : $t_{11}.t_{22} - t_{21}.t_{12} = 1$. Par substitution des équations, il vient l'expression : $\widetilde{K}(f) = \pm\sqrt{\widetilde{L}^2(f).\left(\frac{\nu(f) - \lambda(f).\mu(f)}{\delta(f) - \beta(f).\gamma(f)}\right)}$. Cette expression conduit à 2*n solutions pour n fréquences considérées dans l'étalonnage. La solution adoptée pour chacune des fréquences est celle qui conduit à une variation minimale de la phase en fonction de la fréquence. Il reste alors à choisir parmi 2 solutions ($\pm$). La solution conservée est celle qui permet de retrouver la polarité du pulse de la HPR (qui est connue car préalablement mesurée au scope RF).

[2] Nous verrons plus tard que d'une façon pratique, le LSNA utilise une expression de sa matrice d'erreur en fonction des tension et courant et non pas des ondes incidentes et réfléchies.

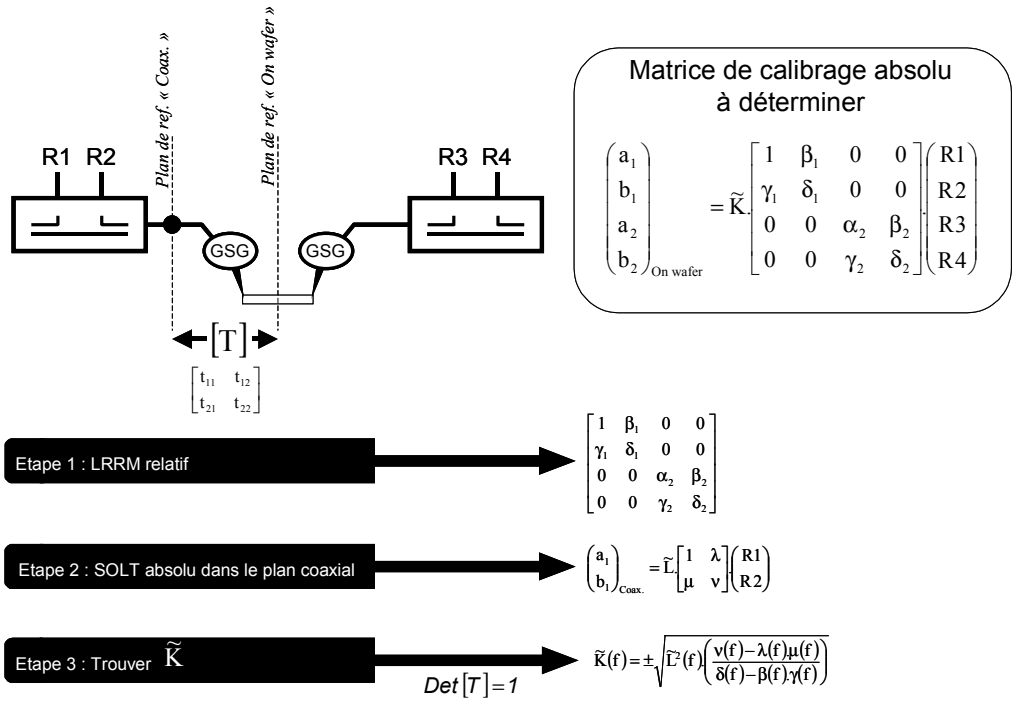


Fig. 8 - Illustration des étapes lors d'un étalonnage LRRM du LSNA

Ainsi, un calibrage absolu sous pointe nécessite une déconnexion d'un port d'un test-set classique. Afin d'éviter cela, le test-set du LSNA est pourvu d'un port « Auxiliaire » afin de pouvoir réaliser un étalonnage LRRM sans la moindre déconnexion. La configuration interne du test-set (dont la gestion du port auxiliaire) est gérée par l'emploi de relais internes pilotables sous 'Mathematica'.

Nous proposons un exemple d'étalonnage LRRM du LSNA dans l'environnement 'Mathematica'.
Pour réaliser un étalonnage LRRM avec le LSNA, il faut tout d'abord vérifier (éventuellement modifier) les paramètres d'étalonnage inclus dans le fichier ressource du LRRM.
Une fois dans l'environnement 'Mathematica', la procédure d'étalonnage se lance comme la procédure SOLT : en appelant le module 'Mathematica' adéquat.

Etalonnage	**LRRM**		**SOLT**	
Mode	**VNA**	**LSNA**	**VNA**	**LSNA**
Fichier Ressource	VNALRRM.mrs	LRRM.mrs	-	-
Fichier Module	VNALRRM.m	LRRM.m	VNALOS.m	LOS.m
Module	VNALRRM`	LRRM`	VNALOS`	LOS`

Fig. 9 - Récapitulatif des différents étalonnages avec le LSNA. Il n'existe pas de fichier « ressource » pour les modules SOLT : les paramètres étalons sont spécifiés dans des « Citifiles ».

<u>Exemples :</u>

Pour le mode VNA, avec une analyse de 1 à 20 GHz en 201 points, on peut taper les lignes :

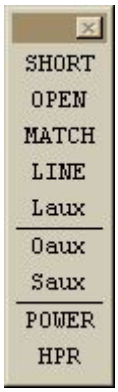
```
StartCal["VNALRRM`", 1*10^9, 20*10^9, 201]

SaveCal["D:\Public\Tibo\ErrorMatrix_vna.m"]
```

Une boite de dialogue sur les étalons utilisée s'affiche au moment du « SartCal ». Elle est similaire à celle du calibrage SOLT. On clique sur chaque standard après les avoir mis en place sur la table sous pointes.
Le calibrage terminé, l'instruction SaveCal conserve dans un fichier texte la matrice d'erreurs relatives obtenue.

Pour le mode LSNA, on peut taper les lignes suivantes pour une analyse de 20 harmoniques dont le fondamental est à 1 GHz.

```
OpenModule["CalKit`"]

LoadCalElementFile["D:\LSNA\Lsna_2_1\NNMS\CalKits\LOS.85056D.citi"]

StartCal["LRRM`", {1*10^9, 20, 0}]

SaveCal["D:\\Public\\Tibo\\CAL21.m"]
```

Les 2 premières lignes sont accessoires car le fichier 'LOS.85056D.citi' est le 'Citifile' par défaut. Il contient les paramètres des étalons SOLT utilisés dans le calibrage absolu LRRM.
La boite de dialogue engendrée par le 'StartCal' est représentée ci-contre. Les 'Short', 'Open', 'Match' et 'Line' correspondent aux standards de la stations sous pointes. Cela correspond à l'étape 1 de la figure 8. Pour l'étape 2, il faut conserver la ligne 'Thru' sous pointes et procéder à l'étalonnage absolu du port 'Auxiliaire' en SOLT : c'est à dire utiliser les standards proposés restants.



Modification du code LRRM pour l'UTF Wiltron

Nous avons souhaité utiliser le pied de test Wiltron en LRRM pour analyser de façon simple la véracité de l'étalonnage des sondes HIP et pour disposer d'un authentique plan de référence. La procédure LRRM telle qu'elle est implémentée n'est malheureusement pas compatible avec les étalons du pied de test. En effet, pour un même standard, le code analyse la propagation de l'onde dans les 2 sens afin de caractériser les erreurs relatives aux 2 ports. Ce code est donc compatible avec des standards « On wafer » mais pas avec ceux du pied de test Wiltron. Ce problème est illustré ci-dessous.

Standards Classiques		Standards Wiltron	
Port 1	**Port 2**	**Port 1**	**Port 2**
OPEN	OPEN	OPEN	SHORT
SHORT	SHORT	SHORT	OPEN
LOAD	LOAD	LOAD	LOAD
LINE		LINE	

Fig. 10 - Différences entre les standards classiques (gérés par les modules LRRM) et les standards du pied de test Wiltron

Les codes des modules LRRM` et VNALRRM`, ainsi que les ressources ont donc été modifiés afin de les rendre compatibles avec le pied de test Wiltron. Ces modifications sont présentées en annexe. Au final, il suffit de rajouter l'instruction « LRRM`UTF=True » (ou « VNALRRM`UTF=True ») dans le fichier ressource correspondant à l'étalonnage désiré pour que l'instruction 'StartCal' conduise aux boites de dialogues suivantes :

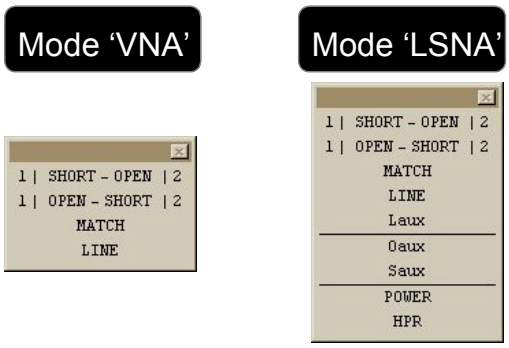


Fig. 11 - Menus interactifs de la phase de calibrage LRRM lorsque la variable UTF est spécifiée vraie

Etalonnage des sondes HIP

Compte tenu des standards utilisés, le plan de référence issu de l'étalonnage est le milieu de la ligne thru 50Ω de 1 cm.

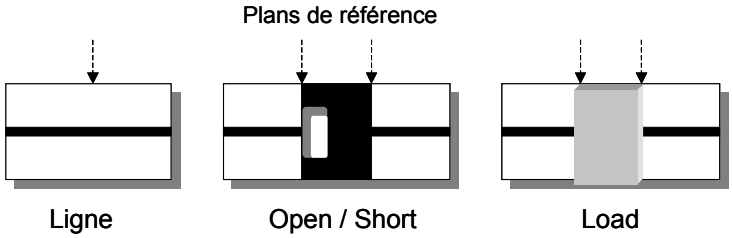


Fig. 12 - Kit de calibrage du pied de test Wiltron

En conséquence, il conviendra de préciser dans le fichier ressource « **LRRM.mrs** » une longueur nulle pour la ligne d'étalonnage car elle se retrouve dans les autres étalons et sera finalement incluse dans la matrice d'erreur du LSNA. La ressource LRRM doit donc inclure les lignes suivantes :

```
LRRM`LineDelay=0*10^-12;
LRRM`UTF=True ;
```

Etalonnage des sondes HIP sous pointes

L'étalonnage des sondes HIP sous pointes a été réalisé avec un kit d'étalonnage 'Cascade'. Cet étalonnage ne nécessite aucune modification du module LRRM fourni initialement avec le LSNA. Le standard 'Thru' est une ligne 50Ω présentant un retard d'environ 1 ps dans la bande DC-40 GHz. La procédure d'étalonnage du LSNA considère les plans de référence à l'extrémité des pointes GSG par de-embedding du centre de la ligne 'Thru' par un retard équivalent à la moitié cette même ligne.
Le fichier ressource LRRM doit présenter les instructions suivantes :

```
LRRM`LineDelay=1*10^-12;
LRRM`UTF=False ;
```

Vérifier la cohérence de l'étalonnage

Le rapport d'avancement précédent s'achevait sur le maillage de la tension le long d'une ligne 50Ω dotée de connecteurs SMA à ses extrémités. L'analyse de tels maillages peut permettre d'évaluer la qualité de l'étalonnage.
Ce paragraphe présente un exemple de mesure conduisant au maillage d'un dispositif avec le LSNA sous l'environnement 'Mathematica'. Les mesures ainsi réalisées sont sauvegardées dans un fichier ASCII afin d'être exportables et exploitables avec d'autres logiciels (Excel ou Scilab).
Un outil d'analyse des mesures de maillage a été réalisé sous l'environnement 'Scilab'. Ce programme sera présenté. Il a permis d'établir des limites aux mesures via sondes hautes impédances notamment sur la déduction de la constante de propagation et sur l'espacement optimal entre les sondes c'est à dire sur les mesures de courants avec les sondes HIP.

Mesure du maillage de standards RF

Un exemple de programme 'Mathematica' est fournit ici. Il conduit à des fichier ASCII correspondant au module et phase de la tension corrigée le long d'une ligne. Ces fichier serons ensuite analyser avec le programme développé sous Scilab.

Exemple sous 'Mathematica'

1. Le LSNA est dans sa configuration classique donc le calibrage aura préalablement été fait (fichier « my_cal.m »). On mesure V1 et V2 : c'est à dire les tensions mesurées dans le plan des GSG. L'origine des phases est imposée telle que la phase de V2 pour la fréquence en cours du « sweep-sin » est nulle. Les données sont sauvegardées dans la matrice **mesures**. Notons que le calibrage de la sonde HIP ainsi que les mesures faites par la suite doivent être réalisés dans la même configuration des atténuateurs programmables internes du « downconverter box » (ici –2 : l'atténuation est minimale).

```
LoadCal[« my_cal.m »] ;
SetSourcePower[0] ;SetSourceFreq[Fundamental] ;SetSourceOutput[On] ;
AutoRange ;
SetPhaseReferenceQuantity[« v2 »] ;
Range // dB
{-2.,-2.}

mesures = Freqsweep[20, 5, "v1"];
mesures = {mesures, Freqsweep[20, 5, "v2"]};
```

2. On déconnecte les canaux 1 et 2 du 'downconverter box' du 'test-set'. Une sonde HIP est reliée au canal 1. L'extrémité de la sonde est placée dans le plan de référence de l'étalonnage LRRM. On détermine le vecteur $\widetilde{K}(f)$ au cours d'un 'sweep-sin'.

```
myK = DefineK1[20];
mesures = FlattenAt[{mesures, myK}, 1];
```

3. On place la sonde HIP en z=0. Cette valeur est arbitraire mais elle constitue le point de départ de notre maillage.

```
mesures = FlattenAt[{mesures, MeasureHIP[myK, 5]}, 1];
```

4. On place la sonde au point suivant. Cette 4ème étape est répétée pour chaque point spatial du maillage. La distance entre les points de mesure est connue par les graduations du positionneur 'Cascade' utilisé avec les sondes HIP.

```
mesures=FlattenAt[{mesures,MeasureHIP[myK,5]},1];
mesures=FlattenAt[{mesures,MeasureHIP[myK,5]},1];
mesures=FlattenAt[{mesures,MeasureHIP[myK,5]},1];
…etc…
```

5. On achève les mesures en remettant la matrice de mesures en forme avant de la sauvegarder en fichier ASCII via la fonction « **Export** ».

```
mesures = Transpose[mesures];
Export["D:\Public\Tibo\Phase_Ligne.txt", Arg[mesures], "Table"] ;
Export["D:\Public\Tibo\Module_Ligne.txt", Abs[mesures], "Table"] ;
```

Les étapes 1 et 2 sont optionnelles. L'analyse sous le programme 'Scilab' peut se faire avec des matrices de mesures ne comportant ni V1, ni V2, ni $\widetilde{K}(f)$. Ceci est fait lorsque le substrat analysé n'est pas le kit de calibrage ayant permis de déterminer $\widetilde{K}(f)$.

Notons qu'il s'agit là de prémices à l'étalonnage du système LSNA associé aux sondes HIP ; En effet, la valeur de $\widetilde{K}(f)$ va changer en fonction de la position des atténuateurs du 'downconverter box'. Nous verrons dans le chapitre 2 le calibrage final du système qui permet des mesures calibrées qu'elle que soit la puissance sondée en ayant fait une seule détermination de $\widetilde{K}(f)$.

Structure du fichier ASCII exporté

La structure des fichiers ASCII exportés est présentée figure 13. Les unités peuvent être pour le module en volt ou en dBV, et pour la phase en degré ou en radian. Cela reste compatible avec l'analyse 'Scilab'.

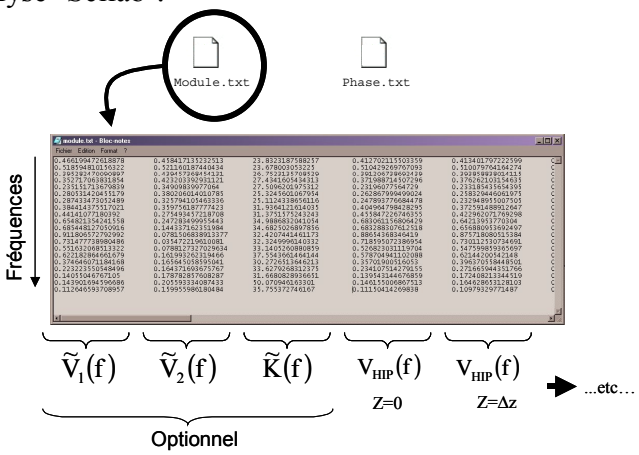


Fig. 13 - Structure des fichiers exportés

Analyse du maillage sous ‘Scilab’

L’analyse des mesures issues du maillage d’une ligne peut vite devenir complexe. Il a donc été utile de développer une interface d’affichage et de calcul. La solution la plus intuitive aurait consisté à développer un module ‘Mathematica’ supplémentaire pour le coupler au LSNA. Mais l’analyse d’un maillage ne nécessite pas de mesures en temps réel car elle travaille à partir d’une banque de données pré-établie. L’analyse se faisant de façon déconnectée des mesures, il est apparu logique de la développer sous ‘Scilab’ afin de pouvoir travailler sur une machine quelconque sans aucun problème de licence.

Présentation du programme développé sous ‘Scilab’

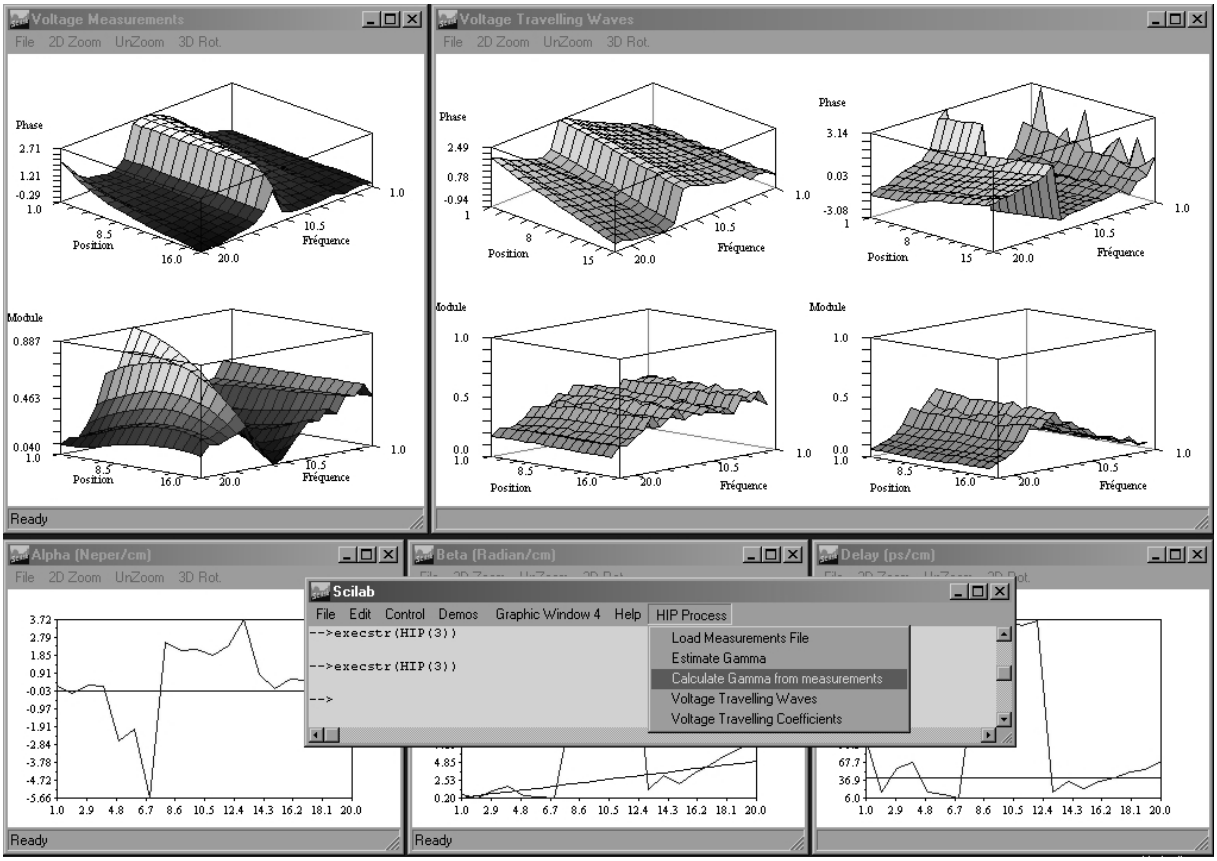


Fig. 14 - Saisie écran de Scilab lors de l’analyse du maillage d’une ligne 50Ω sous le banc Cascade. Les fenêtres en arrière plan représentent ici les tensions mesurées, les ondes de propagation en tension a=V+ et b=V-, ainsi que la constante de propagation déduite d’après certains points. Au premier plan, la fenêtre ‘Scilab’ permet d’exécuter des instructions ‘personalisées’ de l’utilisateur.

Le programme ‘Scilab’ permettant l’analyse des maillages en tension est présent en annexe de ce rapport. Les fonctions élémentaires sont paramétrables via des boites de dialogues TK et son accessibles dans un menu déroulant de l’éditeur ‘Scilab’.
Ces fonctions élémentaires sont :

- le chargement d’une matrice de mesures (les 2 fichiers ASCII correspondants aux module et phase) ;
- la définition d’une estimation de la constante de propagation. Cette estimation est nécessaire pour la détermination expérimentale de la constante de propagation[5]. Les estimateurs implémentés sont le retard linéique (constant dans la bande) et la valeur complexe de la permittivité électrique ;

- le calcul expérimental de la constante de propagation de la ligne sondée à partir de 3 points de mesures. Pour cela l'utilisateur donne les emplacement de 2 mesures : l'emplacement de la 3ème dans le maillage est telles que ces mesures soient équidistantes.
- le calcul des ondes a et b le long du maillage a partir de mesures consécutives de v ;
- le calcul des ondes a et b en un point pour différents espacement entre les mesures de v ;
- l'affichage de ces données en 2D ou 3D.

Des études plus poussées peuvent être faites sous Scilab en réutilisant les procédures proposées.

Exemples de mesures de standards

Des exemples des « mapping » de la tension mesurée via sonde HIP sur les standards du kit de calibrage de l'UTF Wiltron sont présentés ci-dessous.

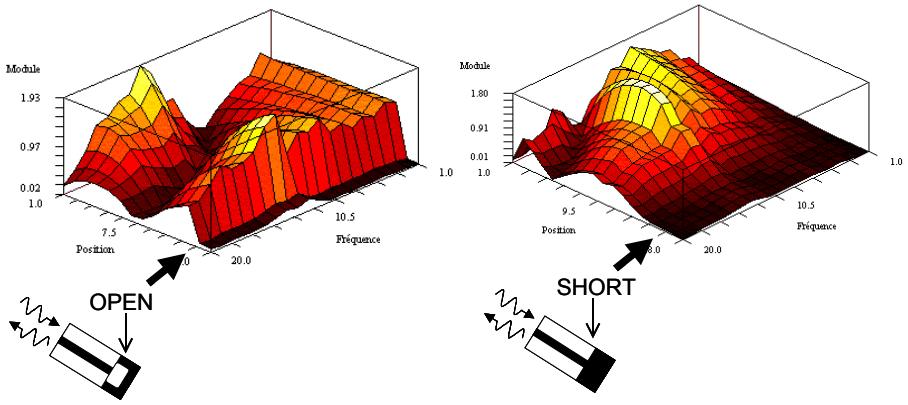


Fig. 15 - Module de la tension mesurée et corrigée issue d'une sonde HIP en fonction de l'emplacement de la sonde et de la fréquence pour les standards OPEN et SHORT.

Le maillage sur la « betty line » permet de calculer les ondes a et b. Ce calcul se fait à partir d'une estimation de la constante de propagation comme un retard linéique uniforme. Le tracé du module des ondes a et b met bien en évidence les variations des ondes aux positions correspondant au changements d'impédance caractéristique. La « betty line » présente une impédance caractéristique de 30Ω au centre et de 50Ω aux extrémités. La constante de propagation n'est pas la même dans toute la structure. L'estimation de γ utilisée pour calculer les ondes a et b est telle que :

$\varepsilon_{r\,eff} = 6.598$ pour Z_C=50Ω (donné par Wiltron dans les caractéristique du kit de calibrage) et

$\varepsilon_{r\,eff} = 7.242$ pour Z_C=30Ω (calculé avec LineCalc disponible dans ADS).

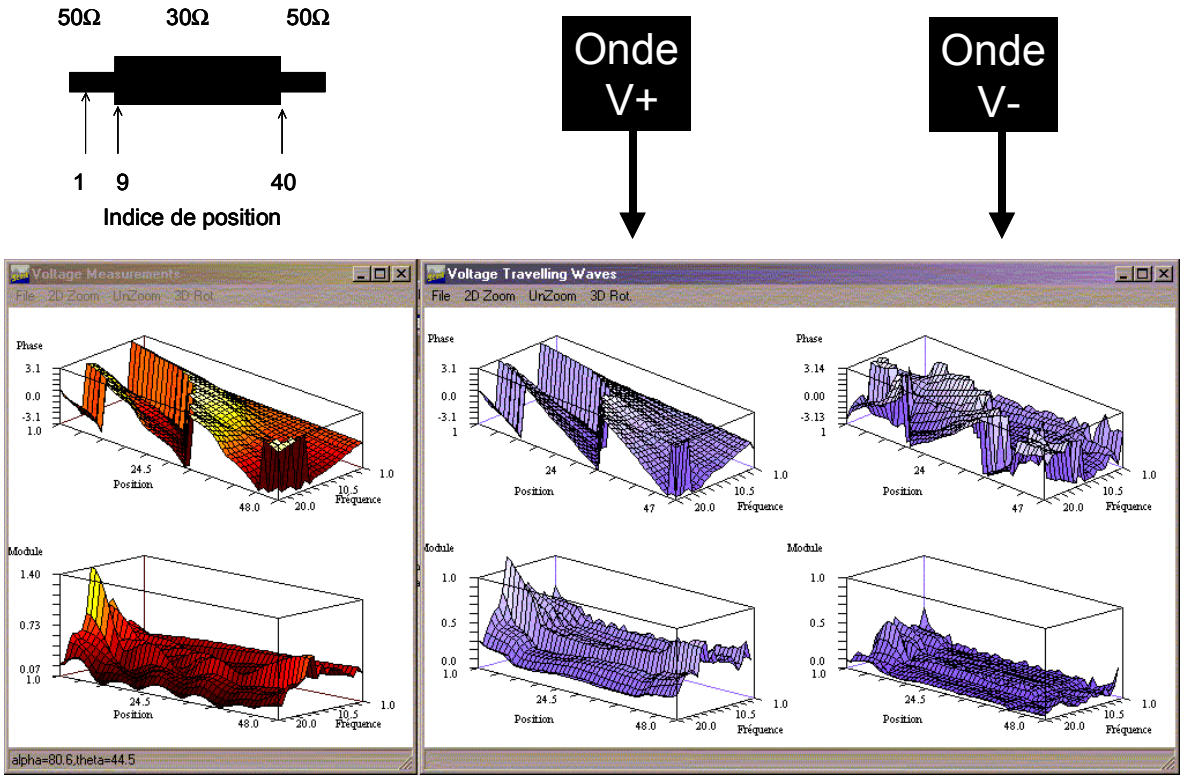


Fig. 16 - Analyse du maillage de la « betty line ». La forme spatiale des ondes a et b montrent clairement le changement d'impédance, même pour une estimation unique de la constante de propagation utilisée pour la « betty-line ».

Comparaison des mesures entre les sondes GSG et les sondes HIP

Une analyse essentielle consiste à comparer un signal mesuré par les sondes GSG (calibrage LRRM classique du LSNA) avec celui mesuré par les sondes HIP calibrées.
Pour cela, dans un premier temps, l'étalonnage des sondes HIP a été réalisé sur le kit de calibrage 'Cascade' afin de déterminer $\widetilde{K}(f)$. Ensuite une ligne coplanaire 50Ω, de grande taille et présente sur le même substrat que les étalons a fait l'objet d'une analyse par maillage.

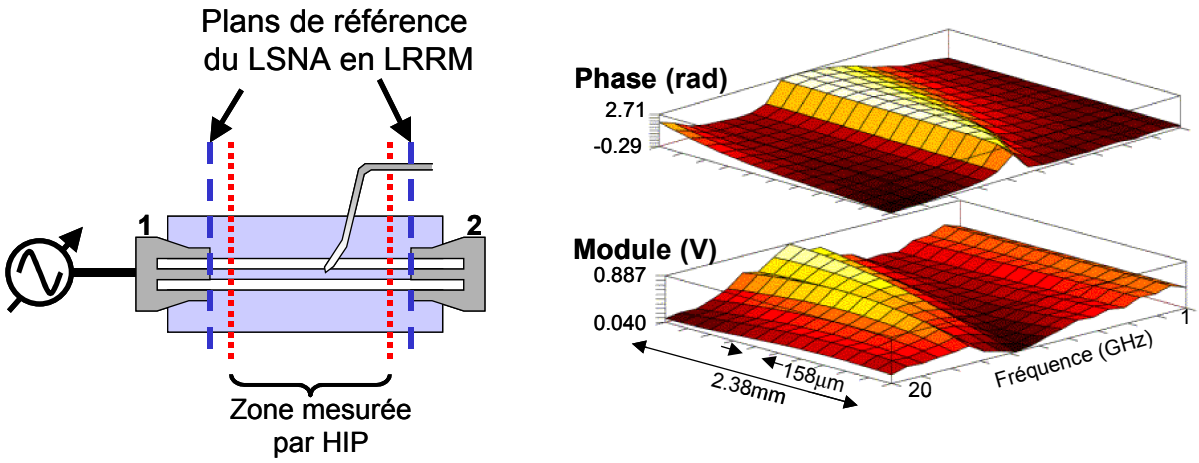


Fig. 17 - Maillage d'une ligne coplanaire sur le banc 'Cascade'

On peut alors comparer les mesures faites dans des conditions identiques par le LSNA classique étalonné en LRRM et le LSNA en configuration HIP calibré lors d'un « sweep-sin ». Ce comparatif est présenté figure 18.

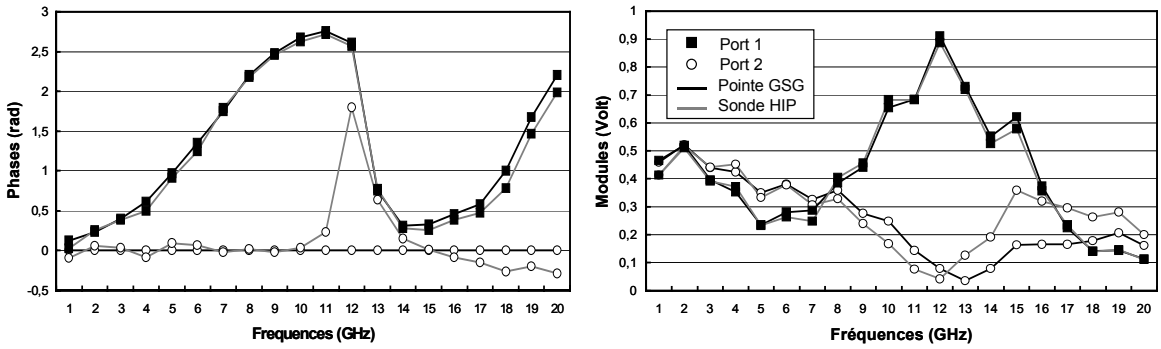


Fig. 18 - Comparatif entre les mesures par sondes GSG et HIP lorsque que la sonde HIP est au plus près du plan de référence GSG.

On constate que les mesures issue de la HIP et de la GSG correspondent. En effet, le plan de mesure HIP n'est pas le même que le plan de référence GSG ce qui explique les quelque différences. Mais en considérant l'évolution du module et de la phase de V_{HIP} en fonction de la position figure 17, par extrapolation spatiale, la concordance des résultats est évidente.

Déduction de la constante de propagation à partir des mesures expérimentales

Le maillage de standards passifs peut servir à valider la procédure d'étalonnage sur le critère de la constante de propagation γ. La détermination expérimentale de γ se fait de la façon suivante :

- Dans un premier temps nous devons disposer d'une estimation de γ. Cette estimation nous permettra de lever l'indétermination dans l'expression du γ mesuré (présence d'un $\pm$).
- Ensuite on doit disposer d'un maillage de ligne dont les points de mesures sont équidistants. De ce maillage, on extrait 3 points de mesures équidistants entre eux. A partir de ces 3 mesures, l'expression donnant γ est :

$$e^{\pm\gamma.L_1} = \frac{V_1 + V_3 \pm \sqrt{(V_1 + V_3)^2 - 4.V_2^2}}{2.V_2}$$

La détermination de γ nécessite quelques précautions compte tenu que la partie imaginaire est modulo 2π. Les solutions trouvées sont comparées à l'estimation afin de ne conserver que la bonne solution.

Pour un même maillage, le calcul de γ en fonction de la fréquence est réalisé pour différentes positions et écarts entre les 3 mesures de base. La cohérence du γ mesuré dépend de la méthode de calcul.

Une première méthode consiste à calculer les 2 solutions $e^{+\gamma.L_1}$ et $e^{-\gamma.L_1}$, les comparer à l'estimation et déterminer ainsi la solution la plus proche de l'estimation afin de résoudre le

problème du signe. Cette méthode est relativement simple, mais peu efficace : le tracé de $\gamma(f)$ présente des quantités fortement incohérentes.

Une seconde méthode consiste à comparer l'estimation aux quantités $\frac{1}{2}.\left[e^{+\gamma.L_1} + \frac{1}{e^{-\gamma.L_1}}\right]$ et $\frac{1}{2}.\left[e^{-\gamma.L_1} + \frac{1}{e^{+\gamma.L_1}}\right]$ afin de déduire la bonne solution pour $\gamma(f)$. Cette méthode s'inspire directement des premières étapes de la procédure d'étalonnage MTRL [5] [6] et a été implémentée dans le programme d'analyse sous Scilab. La résultante est bien meilleure que celle obtenue par la première méthode, mais n'est toujours pas satisfaisante pour une analyse 'broadband' (1 à 20 GHz).

Une troisième méthode consisterait à déterminer $\gamma(f)$ exactement comme dans le code du MTRL. Cela signifie qu'il faut réaliser la seconde méthode pour plusieurs longueurs de lignes (c'est à dire plusieurs espacement entres les 3 points de mesures équidistants) afin d'obtenir une solution qui soit valable pour une large bande spectrale. L'algorithme de cette méthode a été détaillé par D.C. DeGroot [5] mais n'est pour l'instant pas implémenté dans le programme 'Scilab'. La détermination précise de $\gamma(f)$ doit se faire par longueurs de lignes multiples. Cela est dû à la distance optimale entre 2 points de mesures qui dépend étroitement de la fréquence considérée. La détermination de la constante de propagation **ne peut donc se faire avec 3 points de mesures** mais en nécessite un grand nombre.

Mesurer la tension, c'est bien. Avoir le courant, c'est mieux !

Considérons 2 mesures faites via sondes HIP et les 3 paramètres suivants :

$\gamma(f)$: la constante de propagation de la ligne sondée ;
$Z_C(f)$: l'impédance caractéristique de la ligne sondée ;
L : la distance entre les 2 points de mesure.

Il devient alors possible de déduire les ondes de propagations en courant. Mais quelle valeur choisir pour L ?

Choix de la distance entre 2 points de mesure

On se propose d'analyser le maillage d'une ligne en déduisant les ondes V+ et V- (a et b) à partir d'une valeur estimée de la constante de propagation (retard linéique uniforme).
Les ondes sont déduites à partir de 2 mesures de tension : une en position 1 (fixe) et la seconde dont la position varie. Les ondes obtenues sont celles présentes au point de mesure positionné en 1.

La figure 19 illustre ce calcul. L'axe des positions correspond à la position de V2 (la position de V1 est 1). Théoriquement les valeurs de V+ et V- devraient être constantes, or ce n'est pas le cas.
En effet, l'écriture de l'équation des télégraphistes implique une discontinuité lorsque $\gamma.L = k.\pi$ avec k entier.

$$\begin{cases} V_+ = \dfrac{V_1.e^{+\gamma.L} - V_2}{2.\mathrm{Sinh}(\gamma.L)} \\ V_- = \dfrac{V_1.e^{-\gamma.L} - V_2}{-2.\mathrm{Sinh}(\gamma.L)} \end{cases}$$

Le tracé de L en fonction de la fréquence pour $\gamma.L = \pi$ est une hyperbole que l'on devine figure 19.

Le choix de la valeur optimale de L devient critique car :

- L doit être le plus grand possible afin d'avoir un maximum de fluctuation (en fonction de z) sur les tensions mesurées (notamment en basse fréquence) ;
- L doit être défini tel que : $\gamma.L < \pi$

Le choix de L se fera vis à vis de la fréquence à analyser car il n'existe pas de solution 'broadband'. C'est strictement le même problème que la détermination de la constante de propagation qui nécessite plusieurs longueurs de lignes afin de déboucher à une solution correcte en fonction de la fréquence.

Ce qui peut être fait, c'est de fixer L tel que γ.L=160° (il faut le prendre inférieur à 180° pour éviter les erreurs numériques) pour une fréquence égale à l'harmonique la plus haute considérée lors des mesures. Les fréquences les plus basses seront alors assujetties à des erreurs plus importantes.

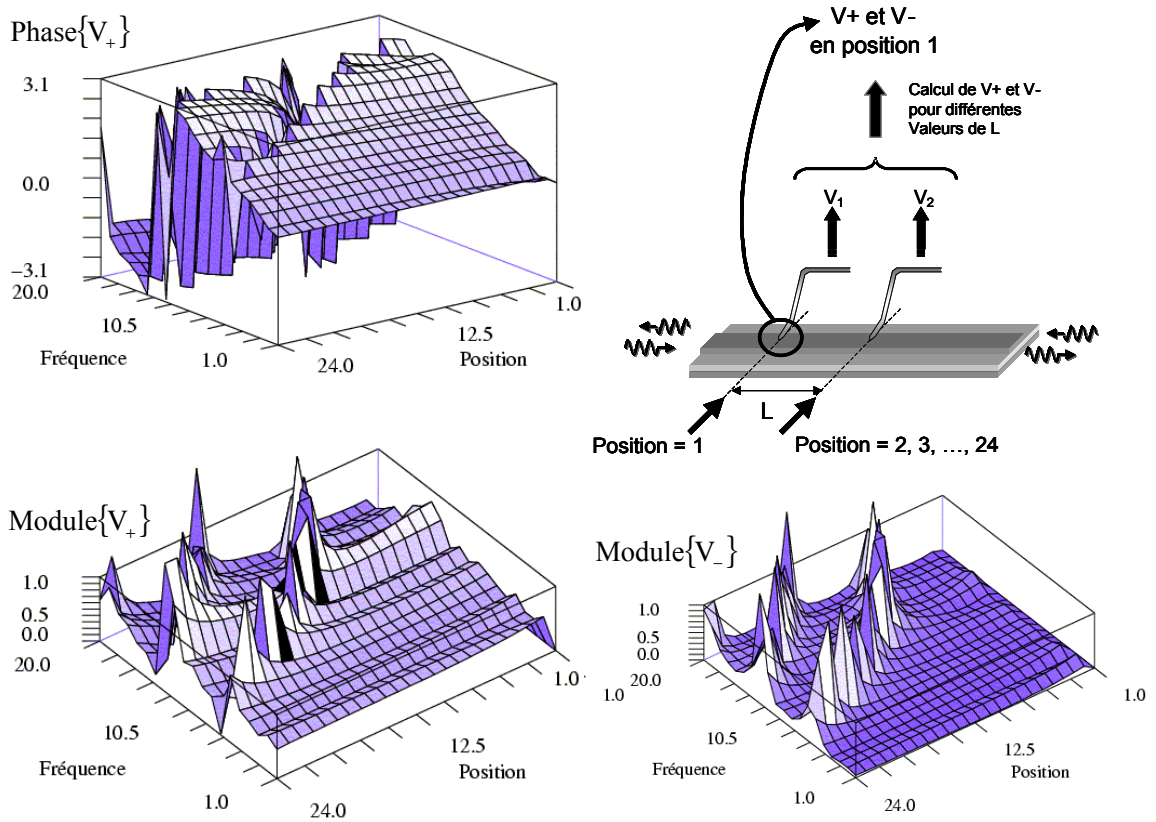


Fig. 19 - Analyse des coefficients V+ et V- en position 1 pour différentes valeurs de L sur une ligne 50Ω mesurée avec le pied de test Wiltron.
Retard linéique estimé : 105.24ps/cm ; Résolution spatiale : 0.3175mm ; Fréquence : 1 à 20 GHz

La mesure de tensions n'est donc pas la meilleure méthode pour déduire les courants. D'autant plus que pour sonder des lignes internes à un MMIC, les distances à considérer sont extrêmement courtes et peuvent conduire à des erreurs lors du calcul du courant (bruit de mesure non négligeables par rapport aux variations de tension).

Une autre alternative pour avoir v et i

Une autre méthode pour mesurer V et I consiste à utiliser, avec le LSNA des sondes non-contactives. Ces sondes permettent la mesure des champs électro-magnétiques E et H.

Avec de tel sondes, l'étalonnage du LSNA reste aussi simple qu'avec l'emploi d'un test-set classique. Des expérimentations ont été faites dans ce sens avec un VNA calibré en puissance par [3].

Le principal avantage est qu'un tel système permet la mesure en 1 point unique de la tension et du courant (le courant n'est pas déduit arithmétiquement comme avec les sondes HIP).En effet, les tensions et courants sont des expressions linéaires des champs E et H.

Deux défauts majeurs apparaissent cependant :

- Le positionnement des sondes non-contactives est extrêmement délicat. Les quantités mesurées vont dépendre très étroitement de la position spatiale des sonde (3D) alors qu'on a seulement 2 degrés de libertés avec les sondes HIP. En conséquence, les manipulations devront faire appel à des procédés de positionnement mécaniques très précis et en conséquence, fort onéreux.
- Ces sondes ne permettent pas la mesure du DC.

Cette méthode de mesure mériterait plus amples investigations. Elle serait particulièrement intéressante lorsque les sondes sont solidaires du substrat analysé (pour éviter toute dispersion mécanique sans pour autant disposer d'un appareillage mécanique sophistiqué) : ces sondes forment un coupleur de test-set localisé et peuvent remplacer le traditionnel coupleur distribué des test-sets pour des systèmes de mesure 'Load-pull' et 'Source-Pull'.

En effet, dans de tels systèmes utilisant de tuners mécaniques, les coefficients de réflexions diminuent fortement entre le plan du tuner et le plan du dispositif sous test. En plaçant de telles sondes au cœur même du tuner (en aval du connecteur d'entrée), on dispose d'un système classique (avec des coupleurs localisés, et non pas distribués) sans les pertes engendrées par les coupleurs distribués : les performances du tuner sont celles appliquées au dispositif sous test.

Dans une telle configuration, le LSNA s'étalonne de façon classique en SOLT ou en LRRM sans la moindre modification de code et sans étape supplémentaire.

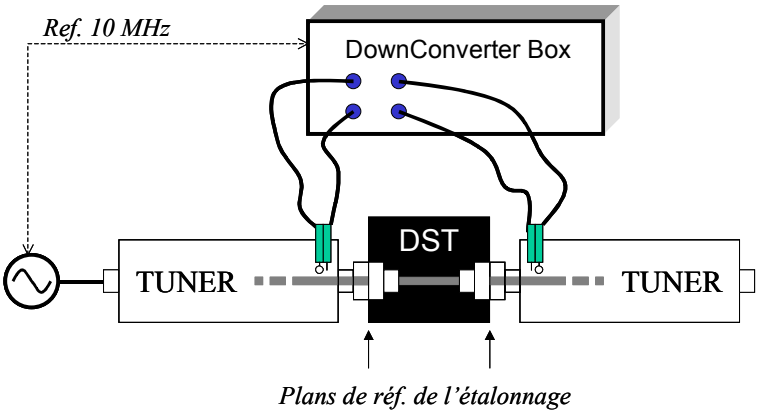


Fig. 20 - Utilisation de sondes non-contactives pour un banc de source-pull et de load-pull autour d'un LSNA.

ETALONNAGE DU SYSTEME DE MESURE LSNA COUPLE AUX SONDES HAUTES IMPEDANCES

Introduction

Nous avons, jusqu'à présent, considéré une hypothèse pour permettre un étalonnage simple et rapide des sondes HIP :

$$\widetilde{V}_{DST}(f) = \widetilde{K}(f).\widetilde{V}_{Mesuré}(f)$$

En partant de cette hypothèse, nous avons pu mettre en place différentes fonctions dans un nouveau module 'Mathematica' : le module « HIP` ». Ce module permet de réaliser l'étalonnage de sondes HIP et de faire des mesures dans des conditions similaires. Cela a été présenté dans le premier chapitre.

Nous disposons donc d'une procédure d'étalonnage et de mesure qui fonctionne uniquement lorsque les atténuateurs programmables internes reliés au sondes HIP sont dans leur position minimale d'atténuation (environ 0dB).

Le but de ce chapitre est de mettre en place un étalonnage complet du système LSNA+HIP qui prenne en compte les différentes positions possibles des atténuateurs programmables internes du 'downconverter box' : en présence d'amplificateurs de puissance, ces atténuateurs changerons de position, ce qu'il faut prendre en compte.

Il faut donc caractériser la fonction $\widetilde{K}(f)$ pour chaque position d'atténuateur possible. Cela reviendrais à établir 5 fonctions de transfert (le LSNA atténue les ondes RF de 0dB à 40dB par pas de 10dB). Mais la mise en place expérimentale manque de souplesse lors de la correction des mesures et n'est pas réalisable lors de l'étalonnage : elle nécessite une très forte puissance dans le plan du dispositif sous test pour conduire à une atténuation de 40 dB en aval de la sonde HIP.

Une seconde possibilité consiste à utiliser les fichiers de pré-caractérisation des atténuateurs internes du LSNA (les matrices [S]) pour appliquer la correction adéquate au moment de la mesure par sonde HIP. Cette procédure s'avère peu flexible et relativement lourde à développer, d'autant plus qu'un code similaire réalise déjà ce type de correction lors de l'utilisation classique de LSNA. Cela créerait une redondance dans le code 'Mathematica' de l'environnement LSNA.

Viens alors une 3ème possibilité : générer la matrice d'erreur du LSNA en fonction des $\widetilde{K}(f)$ relatifs aux HIP. Cette matrice d'erreur corresponds alors au système LSNA dont le port 1 du

‘test-set’ est composé de sondes HIP. Conformément au code ‘Mathematica’ déjà existant, la matrice d’erreur stockée sur disque (sous forme d’un fichier ASCII après la procédure d’étalonnage conventionnelle) n’est valable que lorsque les atténuateurs internes sont en position d’atténuation minimal lors de l’étalonnage : c’est le cas lors du calibrage des sondes HIP. En recréant de façon artificielle le fichier d’étalonnage du LSNA, on créé des avantages considérables :

- La correction relative aux atténuateurs internes programmables se fait automatiquement grâce au code déjà existant.
- Le coupleur d’entrée du ‘test-set’ est remplacé par les sondes HIP. Cela ne change rien pour l’utilisateur : il utilise l’environnement ‘Mathematica’ exactement comme s’il y avait un coupleur classique. L’utilisateur peut alors consulter en temps réel les formes temporelles des ondes v et éventuellement i, mais aussi leurs spectres comme d’habitudes.
- La gestion et l’organisation des mesures restent entièrement compatibles avec ce qui a été développé par NMDG. L’utilisation des ‘Data-Set’ est donc possible.

Aspects théoriques

Ce paragraphe détaille la structure de la matrice d’erreur du LSNA et en décrit sa transformation pour correspondre au système hybride qu’est le LSNA couplé à 2 sondes HIP. Cela va nous amener à considérer 3 méthodes d’étalonnage possibles.

La matrice d’erreur conventionnelle du LSNA

Dans le cadre de ce rapport, les ondes a et b sont des ondes de propagation en tension (ou pseudo-ondes normalisées en tension) V+ et V- dans un environnement 50 ohms (connecteurs du LSNA). Ainsi,

$$\begin{pmatrix} v_1 \\ i_1 \\ v_2 \\ i_2 \end{pmatrix} = \begin{pmatrix} a_1 + b_1 \\ \dfrac{a_1 - b_1}{50} \\ a_2 + b_2 \\ \dfrac{a_2 - b_2}{50} \end{pmatrix}$$

Conformément à la thèse de J. Verspecht (eq. 5-5.1) [4], la matrice d’erreur du LSNA s’exprime, pour une fréquence de l’étalonnage, de la façon suivante :

$$\begin{pmatrix} a_1 \\ b_1 \\ a_2 \\ b_2 \end{pmatrix}_{DUT} = \left|\widetilde{K}\right| . e^{j.\varphi\{\widetilde{K}\}} \bullet \begin{bmatrix} 1 & \beta_I & 0 & 0 \\ \gamma_I & \delta_I & 0 & 0 \\ 0 & 0 & \alpha_{II} & \beta_{II} \\ 0 & 0 & \gamma_{II} & \delta_{II} \end{bmatrix} \bullet \begin{pmatrix} a_1 \\ b_1 \\ a_2 \\ b_2 \end{pmatrix}_{\mathbf{Mesure}}$$

La relation est constituée des 7 termes d'erreurs relatifs issus d'une étalonnage type « VNA » associé aux étalonnages absolus en puissance et en phase. Elle peut s'écrire sous la forme simplifiée suivante :

$$\begin{pmatrix} a_1 \\ b_1 \\ a_2 \\ b_2 \end{pmatrix}_{DUT} = \begin{bmatrix} \alpha_1 & \beta_1 & 0 & 0 \\ \gamma_1 & \delta_1 & 0 & 0 \\ 0 & 0 & \alpha_2 & \beta_2 \\ 0 & 0 & \gamma_2 & \delta_2 \end{bmatrix} \bullet \begin{pmatrix} a_1 \\ b_1 \\ a_2 \\ b_2 \end{pmatrix}_{\mathbf{Mesure}}$$

En réalité, il apparaît que le LSNA déduit les formes d'ondes en tension et courant directement des mesures brutes, d'où l'expression :

$$\begin{pmatrix} v_1 \\ i_1 \\ v_2 \\ i_2 \end{pmatrix}_{DUT} = \begin{bmatrix} 1 & 1 & 0 & 0 \\ \frac{1}{50} & \frac{-1}{50} & 0 & 0 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & \frac{1}{50} & \frac{-1}{50} \end{bmatrix} \bullet \begin{bmatrix} \alpha_1 & \beta_1 & 0 & 0 \\ \gamma_1 & \delta_1 & 0 & 0 \\ 0 & 0 & \alpha_2 & \beta_2 \\ 0 & 0 & \gamma_2 & \delta_2 \end{bmatrix} \bullet \begin{pmatrix} a_1 \\ b_1 \\ a_2 \\ b_2 \end{pmatrix}_{\mathbf{Mesure}}$$

$$\begin{pmatrix} v_1 \\ i_1 \\ v_2 \\ i_2 \end{pmatrix}_{DUT} = \begin{bmatrix} (\alpha_1+\gamma_1) & (\beta_1+\delta_1) & 0 & 0 \\ \left(\frac{\alpha_1-\gamma_1}{50}\right) & \left(\frac{\beta_1-\delta_1}{50}\right) & 0 & 0 \\ 0 & 0 & (\alpha_2+\gamma_2) & (\beta_2+\delta_2) \\ 0 & 0 & \left(\frac{\alpha_2-\gamma_2}{50}\right) & \left(\frac{\beta_2-\delta_2}{50}\right) \end{bmatrix} \bullet \begin{pmatrix} a_1 \\ b_1 \\ a_2 \\ b_2 \end{pmatrix}_{\mathbf{Mesure}}$$

$$\begin{pmatrix} v1 \\ i1 \\ v2 \\ i2 \end{pmatrix}_{LSNA} = \begin{bmatrix} (\alpha_1+\gamma_1) & (\beta_1+\delta_1) & 0 & 0 \\ \left(\frac{\alpha_1-\gamma_1}{50}\right) & \left(\frac{\beta_1-\delta_1}{50}\right) & 0 & 0 \\ 0 & 0 & (\alpha_2+\gamma_2) & (\beta_2+\delta_2) \\ 0 & 0 & \left(\frac{\alpha_2-\gamma_2}{50}\right) & \left(\frac{\beta_2-\delta_2}{50}\right) \end{bmatrix} \bullet \begin{pmatrix} r1 \\ r2 \\ r3 \\ r4 \end{pmatrix}_{LSNA}$$

$$\begin{pmatrix} v1 \\ i1 \\ v2 \\ i2 \end{pmatrix}_{LSNA} = \begin{bmatrix} \alpha_1^{CAL} & \beta_1^{CAL} & 0 & 0 \\ \gamma_1^{CAL} & \delta_1^{CAL} & 0 & 0 \\ 0 & 0 & \alpha_2^{CAL} & \beta_2^{CAL} \\ 0 & 0 & \gamma_2^{CAL} & \delta_2^{CAL} \end{bmatrix} \bullet \begin{pmatrix} r1 \\ r2 \\ r3 \\ r4 \end{pmatrix}_{LSNA}$$

Cette matrice dont huit termes sont non-nuls est la « matrice Cal ». Cette matrice de calibrage est générée par Mathematica à la suite d'un étalonnage (fonction **SaveCal()**) et établit la relation entre les {V ; I} dans le plan du DUT et les ondes mesurées de façon brute (Raw Data).

Le principe de l'implémentation de l'étalonnage des sondes HIP dans l'environnement LSNA consiste à modifier la matrice de calibrage obtenue par un étalonnage conventionnel (SOLT ou LRRM) afin de permettre la mesure calibrée avec un système LSNA hybride (HIP/SOLT) ou (HIP/LRRM) de façon transparente pour l'utilisateur.

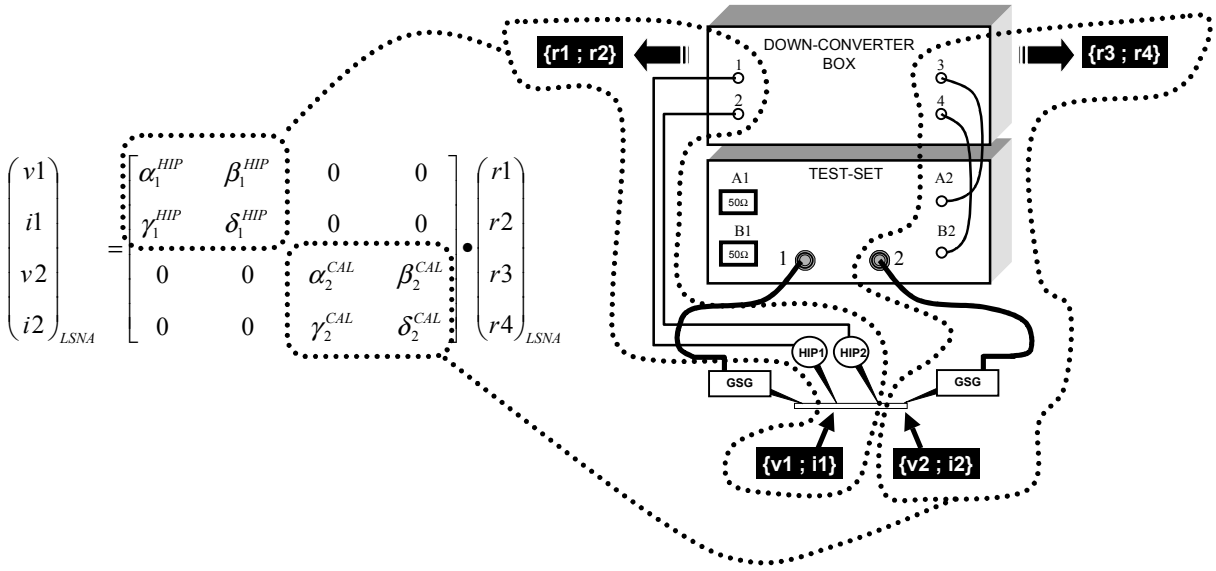


Fig. 21 - Synoptique du sytème de mesure complet associé à sa matrice d'erreur

Méthodes 1 et 2 : modèle d'erreur simple (fonction de transfert en tension)

Ce modèle d'erreur est celui qui a toujours été considéré jusqu'à présent, à savoir que l'on émet l'hypothèse d'une fonction de transfert en tension reliant les quantités réelles dans le plan de la pointe de la sonde haute-impédance aux quantités mesurées après échantillonnage tel que :

$$\widetilde{V}_{DUT}(f) = \widetilde{K}(f).\widetilde{V}_{Mesure}(f)$$

De façon pragmatique, cela revient à définir $\widetilde{K}(f)$ dans le système de mesure LSNA afin que Mathematica puisse fournir la valeur de tension désirée : $\widetilde{V}_{LSNA}(f) = \widetilde{K}(f).R1(f)$ par exemple pour une sonde HIP connectée au port 1 du Downconverter Box.

La caractérisation de $\widetilde{K}(f)$ a déjà fait l'objet d'une description détaillée.

Méthode 1 : l'implémentation rapide

La façon la plus simple d'implémenter la réponse du système de mesure connecté aux sondes HIP 1 et 2, est de définir la matrice d'erreur telle que :

$$\begin{pmatrix} v1 \\ i1 \\ v2 \\ i2 \end{pmatrix}_{LSNA} = \begin{bmatrix} \widetilde{K}_1 & 0 & 0 & 0 \\ 0 & \widetilde{K}_2 & 0 & 0 \\ 0 & 0 & \alpha_2^{CAL} & \beta_2^{CAL} \\ 0 & 0 & \gamma_2^{CAL} & \delta_2^{CAL} \end{bmatrix} \bullet \begin{pmatrix} r1 \\ r2 \\ r3 \\ r4 \end{pmatrix}_{LSNA}$$

L'utilisateur peut alors bénéficier des tension mesurées avec les sondes HIP 1 et 2 grâce aux fonctions associées aux ondes V1 et I1. Si V1 correspond bien à la mesure corrigée provenant de la sonde 1, I1 n'est pas un courant, mais la tension corrigée issue de la sonde 2.
Les mesures faites dans un tel environnement ne permettent pas d'utiliser les ondes a1 et b1 : elles sont calculées à partir de V1 et I1, et n'ont donc aucun sens dans cette configuration.
Cette méthode d'étalonnage peut être utile pour mesurer uniquement des tensions (avec des sondes indépendantes). Les ondes a1 et b1 ainsi que le courant I1 peuvent être calculés par l'utilisateur à partir de 2 mesures séquentielles ou simultanée, mais n'est pas fourni par une mesure en temps réel.

Méthode 2 : implémentation conventionnelle

L'utilisation de 2 sondes HIP doit permettre l'obtention en temps réel des ondes de propagation en tension a et b (V+ et V-) par le LSNA. Cela est possible en considérant les équations des télégraphistes.

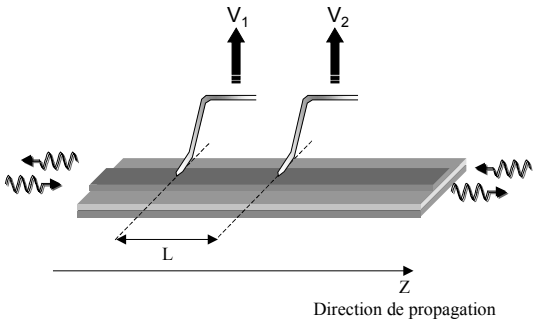


Fig. 22 - Mesures simultanées afin d'obtenir directement V1 et I1 (et donc a1 et b1)

Au point de mesure de V1, on a :

$$\begin{cases} V^{+}_{\text{corrigé}} = a_{DUT} = \dfrac{\tilde{K}_1.\tilde{R}_1.e^{+\gamma.L} - \tilde{K}_2.\tilde{R}_2}{2.\text{Sinh}(\gamma.L)} \\ V^{-}_{\text{corrigé}} = b_{DUT} = \dfrac{\tilde{K}_1.\tilde{R}_1.e^{-\gamma.L} - \tilde{K}_2.\tilde{R}_2}{-2.\text{Sinh}(\gamma.L)} \end{cases}$$

Il vient alors la matrice d'erreur entre les onde a et b dans le plan de mesure et les mesures brutes R1 et R2.

$$\begin{pmatrix} a \\ b \end{pmatrix}_{LSNA} = \frac{1}{2.\text{Sinh}(\gamma.L)} \begin{bmatrix} \tilde{K}_1.e^{+\gamma.L} & -\tilde{K}_2 \\ -\tilde{K}_1.e^{-\gamma.L} & \tilde{K}_2 \end{bmatrix} \bullet \begin{pmatrix} R1 \\ R2 \end{pmatrix}_{LSNA}$$

La matrice de calibrage correspondant au port 1 du LSNA devient alors :

$$\begin{pmatrix} v1 \\ i1 \end{pmatrix}_{LSNA} = \frac{1}{2.\text{Sinh}(\gamma.L)} \begin{bmatrix} \tilde{K}_1.e^{+\gamma.L} - \tilde{K}_1.e^{-\gamma.L} & 0 \\ \dfrac{\tilde{K}_1.e^{+\gamma.L} + \tilde{K}_1.e^{-\gamma.L}}{Z_C} & \dfrac{-2.\tilde{K}_2}{Z_C} \end{bmatrix} \bullet \begin{pmatrix} R1 \\ R2 \end{pmatrix}_{LSNA}$$

où γ, Z_C et L représentent respectivement la constante de propagation de la ligne sondée, son impédance caractéristique et la distance entre les sonde HIP lors de la mesure.

En imposant la matrice de calibrage suivante au logiciel pilotant le LSNA, l'utilisation des sondes HIP deviens transparente et s'apparente à un test-set normal. Ainsi l'onde a1

correspond à V+ et b1 à V-. Les données V1 et I1 correspondent aux tensions et courants corrigés dans le plan de mesure de la sonde HIP numéro 1.

$$\begin{pmatrix} v1 \\ i1 \\ v2 \\ i2 \end{pmatrix}_{LSNA} = \begin{bmatrix} \widetilde{K}_1 & 0 & 0 & 0 \\ \left(\dfrac{\widetilde{K}_1.Coth(\gamma.L)}{Z_C}\right) & \left(\dfrac{-\widetilde{K}_2}{Z_C.Sinh(\gamma.L)}\right) & 0 & 0 \\ 0 & 0 & \alpha_2^{CAL} & \beta_2^{CAL} \\ 0 & 0 & \gamma_2^{CAL} & \delta_2^{CAL} \end{bmatrix} \bullet \begin{pmatrix} r1 \\ r2 \\ r3 \\ r4 \end{pmatrix}_{LSNA}$$

Voici donc une approche relativement souple qui part de la détermination des fonctions de transfert en tension $\widetilde{K}_1(f)$ et $\widetilde{K}_2(f)$ et qui permet à l'utilisateur d'obtenir toutes les formes d'ondes de façon strictement identique à une utilisation du LSNA via un 'test-set'. Cette approche reviens à considérer le couple de sondes HIP comme un coupleur localisé et non plus distribué. On prolongeant cette dernière considération, on peut imaginer un nouvel étalonnage qui caractérise ce coupleur localisé, nous permettant alors d'éliminer l'hypothèse de départ $\widetilde{V}_{DST}(f) = \widetilde{K}(f).\widetilde{V}_{Mesuré}(f)$: c'est l'objet d'une 3ème méthode décrite ci-après.

Méthode 3 : modèle d'erreur étendu (fonction de transfert en onde de propagation)

Principe

D'une façon générale, en présence d'un 'test-set' composé d'un coupleur distribué (sondes HIP), la matrice d'erreur du LSNA s'écrit de la sorte :

$$\begin{pmatrix} v1 \\ i1 \\ v2 \\ i2 \end{pmatrix}_{LSNA} = \begin{bmatrix} \alpha_1^{HIP} & \beta_1^{HIP} & 0 & 0 \\ \gamma_1^{HIP} & \delta_1^{HIP} & 0 & 0 \\ 0 & 0 & \alpha_2^{CAL} & \beta_2^{CAL} \\ 0 & 0 & \gamma_2^{CAL} & \delta_2^{CAL} \end{bmatrix} \bullet \begin{pmatrix} r1 \\ r2 \\ r3 \\ r4 \end{pmatrix}_{LSNA}$$

Jusqu'à présent, nous émettions l'hypothèse $\widetilde{V}_{DST}(f) = \widetilde{K}(f).\widetilde{V}_{Mesuré}(f)$. Cette relation est accrédité par une vérification faite par le NIST [1] mais elle est restrictive.

En effet, d'une part, on ne tient pas compte de la réponse de la sonde en fonction de l'impédance caractéristique de la ligne mesurée. Classiquement l'étalonnage de la sonde HIP se fait sur une ligne 50Ω. Cela implique que :

- si $\widetilde{K}(f)$ dépend fortement de l'impédance caractéristique, les impédances des substrats mesurés soient proches de 50 Ω ;
- ou émettre l'hypothèse que $\widetilde{K}(f)$ dépend peu de l'impédance caractéristique mesurée.

D'autre part, lorsque les sondes sont caractérisées (déduction de $\widetilde{K}(f)$) pour être ensuite utilisées de façon couplée (méthode 2), le terme d'erreur e_{12} est égal à zéro. Cela s'explique physiquement par l'absence de couplage entre les 2 sondes utilisées simultanément. Ceci est

normal car les sondes HIP sont caractérisées individuellement selon une analyse de signaux (la tension). Mais leur utilisation en mesure par la méthode 2 les présente comme un coupleur (analyse en ondes : a et b) dans lequel peut exister un couplage. Ainsi le terme e_{12} peut ne pas être nul. Cela est vérifiable en étalonnant le LSNA en présence des 2 sondes HIP mesurant simultanément.
On serait tenter pour cet étalonnage de reprendre les procédures classique SOLT ou LRRM (les sondes HIP jouant le rôle de coupleur) mais une inconsistance physique nous limite : il faut pouvoir placer 1 sonde dans le plan de référence des standards et la seconde en retrait sur une ligne débouchant sur le même standard. Cet étalonnage n'est donc pas possible avec un kit d'étalonnage classique car nous ne disposons pas du kit approprié. Une astuce permet néanmoins d'étalonner le système avec des standards classique : c'est l'objet du paragraphe suivant.

Etalonnage : voir les sondes HIP comme un coupleur localisé

En considérant les 2 sondes comme un « test-set » localisé, on peut réaliser un étalonnage permettant de déterminer les termes d'erreur α_1^{HIP}, β_1^{HIP}, γ_1^{HIP} et δ_1^{HIP}. Le principe de l'étalonnage est le même que ceux utilisé sur les VNA traditionnels : on doit disposer de suffisamment de standards parfaitement connus, que l'on mesure, afin d'identifier les termes d'erreurs. Cependant aucun standard n'est approprié pour la configuration du port 1 du système de mesure car il est constitué de 2 sondes disjointes. C'est pourquoi, le LSNA s'avère particulièrement intéressant. En effet, on peut utiliser le port 2, préalablement étalonné en LRRM, afin de constituer nos standards en mesurant les ondes v2 et i2.

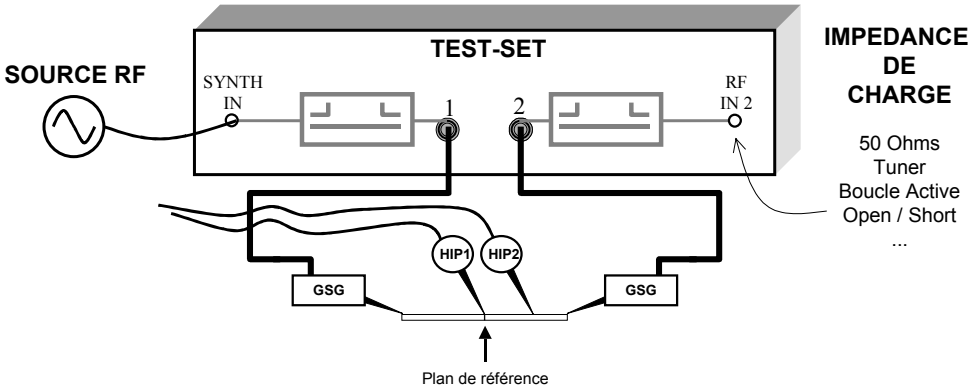


Fig. 23 - Configuration du 'test-set' lors de l'étalonnage par la méthode 3.

Dans le plan de référence d'une ligne « Thru », les ondes v1 et v2 ainsi que i1 et i2 sont identiques entre elles. Une mesure du port 2 peut nous permettre de déterminer 2 des 4 termes d'erreurs recherchés (HIP). L'idée de l'étalonnage consiste a mesurer v2 et i2 pour différentes perturbations afin de permettre la détermination des 4 termes d'erreur. Pour cela on place différentes impédances coaxiales 2.4mm dans le port « RF IN2 » du « test-set ». Ces impédances ramenées dans le plan de référence de la ligne « Thru » peuvent être entièrement mesurées par le port 2 du LSNA et constituer nos standards pour l'étalonnage du port 1 (les 2 sondes HIP). On peut imaginer utiliser le kit de calibrage 2.4mm (Short / Open / Load) sur le port « RF IN2 » pour une telle procédure. Deux conditions de mesures (par exemple Short et Open) sont nécessaires à la détermination des 4 termes d'erreur recherchés.

Ces 4 termes d'erreur sont représentés par la sous matrice suivante :

$$\begin{pmatrix} v1 \\ i1 \end{pmatrix}_{LSNA} = \begin{bmatrix} \alpha_1^{HIP} & \beta_1^{HIP} \\ \gamma_1^{HIP} & \delta_1^{HIP} \end{bmatrix} \bullet \begin{pmatrix} r1 \\ r2 \end{pmatrix}_{LSNA}$$

Pour 2 conditions de mesures C1 et C2, il vient :

$$\alpha_1^{HIP} = \frac{r2_{C2}.v1_{C1} - r2_{C1}.v1_{C2}}{r1_{C1}.r2_{C2} - r1_{C2}.r2_{C1}}$$

$$\beta_1^{HIP} = \frac{r1_{C2}.v1_{C1} - r1_{C1}.v1_{C2}}{r1_{C2}.r2_{C1} - r1_{C1}.r2_{C2}}$$

$$\gamma_1^{HIP} = \frac{r2_{C2}.i1_{C1} - r2_{C1}.i1_{C2}}{r1_{C1}.r2_{C2} - r1_{C2}.r2_{C1}}$$

$$\delta_1^{HIP} = \frac{r1_{C2}.i1_{C1} - r1_{C1}.i1_{C2}}{r1_{C2}.r2_{C1} - r1_{C1}.r2_{C2}}$$

Le système ainsi calibré permet de réaliser des mesures sans connaissance « a priori », jusqu'ici nécessaire, de la constante de propagation (γ) et de l'impédance caractéristique (Z_C) si et seulement si le substrat d'étalonnage et de mesure ont les même propriétés physiques. Ces paramètres (γ et Z_C) sont inclus dans la matrice d'erreur issue de l'étalonnage.

En effet, il faut noter que théoriquement cet étalonnage dépend étroitement de la distance entre les sondes HIP (L), de la constante de propagation de la ligne « Thru » utilisée (γ) ainsi que de l'impédance caractéristique de cette même ligne (Z_C). Néanmoins une connaissance parfaite de ces 3 paramètres peut conduire à la possibilité de réaliser un étalonnage et des mesures sur des lignes différentes en définissant des termes d'erreur « indépendants » de ces 3 paramètres : K_1^+, K_1^-, K_2^+, K_2^- qui sont les corrections à apporter sur l'ondes incidente (a = V+) et réfléchie(b = V-) mesurée par les sondes HIP (1 et 2).

Cela constitue une nouvelle hypothèse sur la transformation des ondes mesurées par le système. Au lieu de considérer une fonction de transfert en tension, on considère une fonction de transfert en onde de propagation. Ainsi une sonde HIP (HIP 1 par exemple) est caractérisée par 2 fonctions de transferts : K_1^+ pour l'onde incidente et K_1^- pour l'onde réfléchie.

Associer un étalonnage et une mesure réalisés sur des substrats différents

En utilisant les 4 nouveaux termes décris précédemment, on peut exprimer les ondes de propagation en tension corrigées et obtenues par mesures brutes sur le LSNA (R1 et R2). Cela conduit à la relation matricielle suivante :

$$\begin{pmatrix} a \\ b \end{pmatrix}_{LSNA} = \frac{1}{2.\mathrm{Sinh}(\gamma.L)} \begin{bmatrix} \tilde{K}_1^+.e^{+\gamma.L} & -\tilde{K}_2^+ \\ -\tilde{K}_1^-.e^{-\gamma.L} & \tilde{K}_2^- \end{bmatrix} \bullet \begin{pmatrix} R1 \\ R2 \end{pmatrix}_{LSNA}$$

Il vient alors la matrice de calibrage totale du LSNA utilisé avec 2 sondes HIP :

$$\begin{pmatrix} v1 \\ i1 \\ v2 \\ i2 \end{pmatrix}_{LSNA} = \begin{bmatrix} \left(\dfrac{\widetilde{K}_1^+.e^{+\gamma.L} - \widetilde{K}_1^-.e^{-\gamma.L}}{2.Sinh(\gamma.L)}\right) & \left(\dfrac{-\left(\widetilde{K}_2^+ - \widetilde{K}_2^-\right)}{2.Sinh(\gamma.L)}\right) & 0 & 0 \\ \left(\dfrac{\widetilde{K}_1^+.e^{+\gamma.L} + \widetilde{K}_1^-.e^{-\gamma.L}}{2.Z_C.Sinh(\gamma.L)}\right) & \left(\dfrac{-\left(\widetilde{K}_2^+ + \widetilde{K}_2^-\right)}{2.Z_C.Sinh(\gamma.L)}\right) & 0 & 0 \\ 0 & 0 & \alpha_2^{CAL} & \beta_2^{CAL} \\ 0 & 0 & \gamma_2^{CAL} & \delta_2^{CAL} \end{bmatrix} \bullet \begin{pmatrix} r1 \\ r2 \\ r3 \\ r4 \end{pmatrix}_{LSNA}$$

Ainsi, il devient possible d'extraire les termes K_1^+, K_1^-, K_2^+, K_2^- lors de l'étalonnage et de construire « artificiellement » la matrice d'erreur utilisée lors de mesures (où Z_C et γ sont « a priori » différents par rapport à l'étalonnage). Pour 2 conditions de mesures différentes : C1 et C2 (par exemple « Open » et « Short »), il vient :

$$K_1^+ = \frac{e^{-\gamma.L}.(-r2_{C2}.(v1_{C1} + i1_{C1}.Z_C) + r2_{C1}(v1_{C2} + i1_{C2}.Z_C)).Sinh(\gamma.L)}{r1_{C2}.r2_{C1} - r1_{C1}.r2_{C2}}$$

$$K_1^- = \frac{e^{+\gamma.L}.(r2_{C2}.v1_{C1} - r2_{C1}.v1_{C2} + i1_{C2}.r2_{C1}.Z_C - i1_{C1}.r2_{C2}.Z_C).Sinh(\gamma.L)}{r1_{C2}.r2_{C1} - r1_{C1}.r2_{C2}}$$

$$K_2^+ = \frac{(-r1_{C2}.(v1_{C1} + i1_{C1}.Z_C) + r1_{C1}(v1_{C2} + i1_{C2}.Z_C)).Sinh(\gamma.L)}{r1_{C2}.r2_{C1} - r1_{C1}.r2_{C2}}$$

$$K_2^- = \frac{(r1_{C2}.v1_{C1} - r1_{C1}.v1_{C2} + i1_{C2}.r1_{C1}.Z_C - i1_{C1}.r1_{C2}.Z_C).Sinh(\gamma.L)}{r1_{C2}.r2_{C1} - r1_{C1}.r2_{C2}}$$

Notons que lors du second étalonnage (le premier étant le LRRM classique étalonnant le port 2), les termes v1 et i1, présents dans les expressions ci-dessus, sont mesurés par le port 2 (il s'agit en fait de v2 et i2).

Si l'on souhaite exprimer ces coefficients à partir de la sous-matrice [2*2] produite par la nouvelle procédure d'étalonnage, on a :

$$K_1^+ = e^{-\gamma.L}.Sinh(\gamma.L).(\alpha_1^{HIP} + Z_C.\gamma_1^{HIP})$$
$$K_1^- = e^{+\gamma.L}.Sinh(\gamma.L).(-\alpha_1^{HIP} + Z_C.\gamma_1^{HIP})$$
$$K_2^+ = Sinh(\gamma.L).(-\beta_1^{HIP} - Z_C.\delta_1^{HIP})$$
$$K_2^- = Sinh(\gamma.L).(\beta_1^{HIP} - Z_C.\delta_1^{HIP})$$

Pratiquement, on utilisera cette solution sous 'Mathematica'. La figure 24 est un synoptique illustrant les opérations qui sont prévue dans le cas de mesures avec cette nouvelle procédure d'étalonnage.

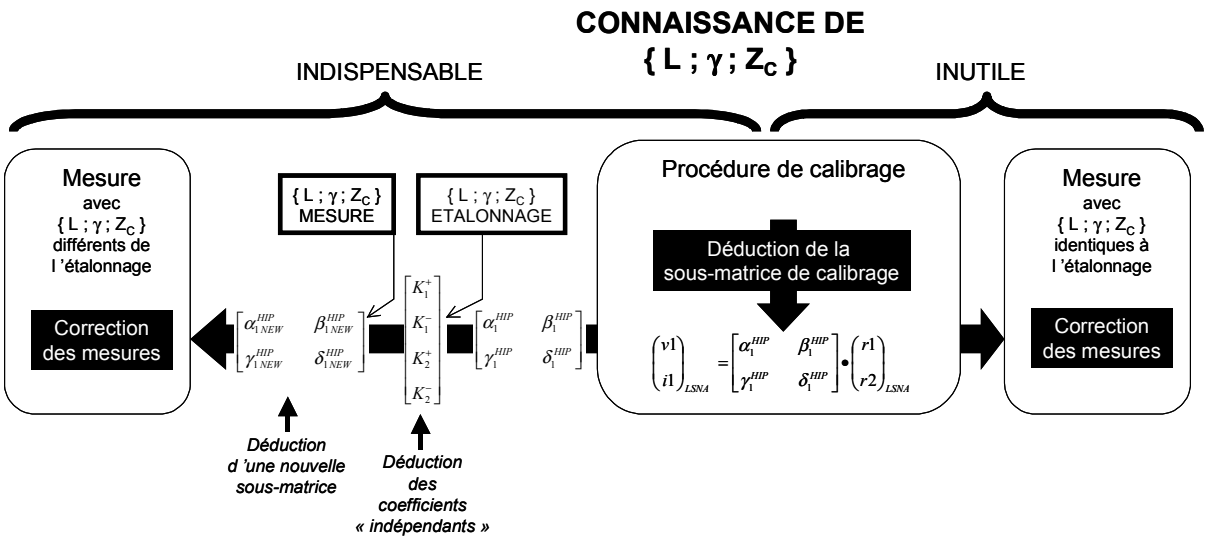


Fig. 24 - Etalonnage et mesure en utilisant la méthode 3 : les paramètres caractéristiques des substrats ont besoin d'être connus lorsque l'étalonnage et les mesures ne se font pas dans des conditions physiques identiques.

La relation gérant la transformation de la sous-matrice de calibrage d'un environnement de calibrage $\{L_{CAL}\ ;\ \gamma_{CAL}\ ;\ Z_{C\,CAL}\}$ vers un environnement de mesure $\{L_{MES}\ ;\ \gamma_{MES}\ ;\ Z_{C\,MES}\}$ peut être exprimée suivant la sous matrice d'erreur dont les coefficients sont les suivants :

$$\alpha_{1\,NEW}^{HIP} = \frac{\mathrm{Sinh}(\gamma_{CAL}.L_{CAL})}{\mathrm{Sinh}(\gamma_{MES}.L_{MES})}.\left[\left(\alpha_1^{HIP}.\mathrm{Cosh}((\gamma_{CAL}.L_{CAL})-(\gamma_{MES}.L_{MES}))\right)-\left(Z_{C\,CAL}.\gamma_1^{HIP}.\mathrm{Sinh}((\gamma_{CAL}.L_{CAL})-(\gamma_{MES}.L_{MES}))\right)\right]$$

$$\beta_{1\,NEW}^{HIP} = \beta_1^{HIP}.\frac{\mathrm{Sinh}(\gamma_{CAL}.L_{CAL})}{\mathrm{Sinh}(\gamma_{MES}.L_{MES})}$$

$$\gamma_{1\,NEW}^{HIP} = \frac{1}{Z_{C\,MES}}.\frac{\mathrm{Sinh}(\gamma_{CAL}.L_{CAL})}{\mathrm{Sinh}(\gamma_{MES}.L_{MES})}.\left[\left(Z_{C\,CAL}.\gamma_1^{HIP}.\mathrm{Cosh}((\gamma_{CAL}.L_{CAL})-(\gamma_{MES}.L_{MES}))\right)-\left(\alpha_1^{HIP}.\mathrm{Sinh}((\gamma_{CAL}.L_{CAL})-(\gamma_{MES}.L_{MES}))\right)\right]$$

$$\delta_{1\,NEW}^{HIP} = \delta_1^{HIP}.\frac{Z_{C\,CAL}}{Z_{C\,MES}}.\frac{\mathrm{Sinh}(\gamma_{CAL}.L_{CAL})}{\mathrm{Sinh}(\gamma_{MES}.L_{MES})}$$

Synthèse des 3 configurations « Calibrage / Mesure » proposées

La figure 25 résume ce qui a été présenté dans ce chapitre.

Il existe donc 2 façons de calibrer les sondes HIP.

La première façon revient à déterminer leurs fonctions de transfert en tension. Typiquement le calibrage se fait de façon indépendante sur chacune des sondes et sur une ligne 50Ω. Travailler avec un LSNA simplifie fortement la procédure d'étalonnage par rapport à ce qui a pu être mis en place par le NIST. Mais fondamentalement, les hypothèses de travail sont les mêmes : il n'y pas eu à l'heure actuelle de caractérisation des sondes HIP paramétrée en impédance caractéristique (elle est supposée voisine de 50Ω).
Une fois étalonnées de la sorte, les sondes HIP peuvent être utilisées indépendamment (méthode 1) pour une analyse de signal ou couplées (méthode 2) pour une analyse d'onde. L'analyse de signal ne permet d'obtenir que la tension en temps réel avec le LSNA mais reste

cohérente avec l'étalonnage (lui aussi structuré autour d'un modèle d'analyse de signal). Si on désire avoir les ondes a, b ou le courant, cela nécessite 2 mesures puis un calcul fait par l'utilisateur. L'analyse d'onde permet d'obtenir les ondes v, i ,a et b mais est incohérente vis à vis de l'étalonnage. Cela nous amène donc à une seconde procédure d'étalonnage spécifique à l'emploi de 2 sondes HIP simultanément.

La seconde procédure d'étalonnage des HIP reviens à considérer les 2 sondes comme un coupleur discret. Contrairement à la première méthode, le modèle d'erreur des sondes n'est pas une fonction de transfert en tension mais en ondes (a et b). Cela permet d'introduire dans la matrice d'erreur un couplage existant entre 2 sondes posées côte à côte. La réponse d'une sonde à une onde incidente n'est plus la même que la réponse à l'onde réfléchie (cela incluse les perturbations engendrées par la seconde sonde). Cette procédure d'étalonnage est la plus rigoureuse mais elle nécessite tous les paramètres physique des structures sondées lors de l'étalonnage et de la mesure car elle permet la mesure d'un substrat dont l'impédance caractéristique est fort différente de celui de l'étalonnage.

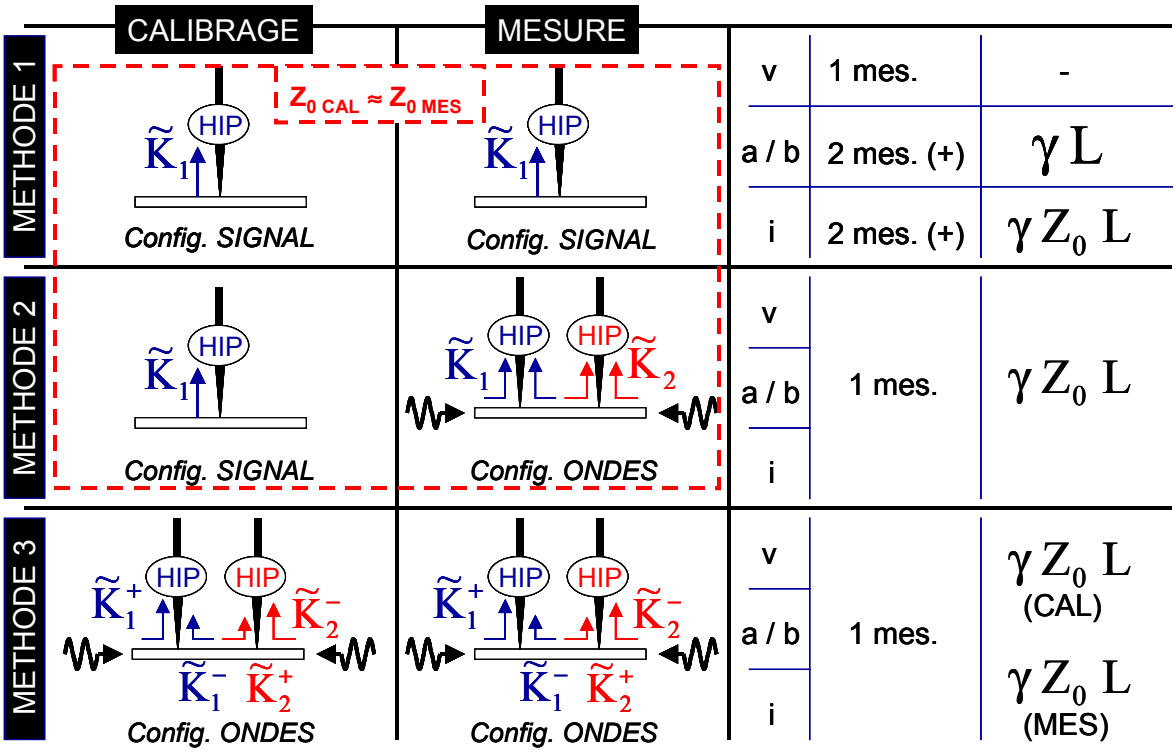


Fig. 25 - Synthèse des méthodes proposées

Aspects pragmatiques sur la correction des ondes mesurées par le LSNA

Ce paragraphe présente le fichier de calibrage issu d'un étalonnage standard du LSNA ainsi que son utilisation lors de mesures. Une telle connaissance nous permettra alors de construire artificiellement une matrice de calibrage unique pour le système LSNA associé aux sondes HIP.

Fichier de calibrage standard du LSNA

Les « user-setup »

Typiquement, après avoir complété une procédure d'étalonnage, on sauvegarde la matrice d'erreur issue de cet étalonnage en exécutant sous Mathematica l'instruction :

```
SaveCal["D:\Public\Tibo\my_cal.m"]
```

Cela créé un fichier ASCII qui contient entre autre la matrice d'erreur du LSNA pour l'étalonnage qui vient d'être fait.

Le logiciel pilotant le LSNA sous Mathematica permet de gérer plusieurs configurations de mesures dans un même étalonnage. Par exemple si on souhaite faire des mesures pour différentes fréquences fondamentales, cela peut être spécifié dans la configuration utilisateur (ou « user setup »). En conséquence, on réalisera 1 seul étalonnage. En réalité, pour chaque standard d'étalonnage, toutes les configurations utilisateur sont balayées ce qui transparent pour l'utilisateur et produit un gain de temps considérable. La sauvegarde de ces « multiples étalonnage en un » sont sauvegardées dans un fichier ASCII unique.

Le détail du contenu du fichier de calibrage va être présenté. L'utilisation de plusieurs « user-setup » n'est pas prévu dans les nouvelles fonctionnalités du module « HIP` » dont le listing est fourni en annexe. En conséquences, tous les étalonnages utilisés dans le cadre des sondes HIP ne doivent comporter qu'un seul « user setup ».

Structure du fichier de calibrage

La description du fichier de calibrage issu du mode LSNA est faite ici dans un cas général (plusieurs user-setup). Les modifications de fichiers de calibrage proposées dans le module « HIP` » ne fonctionnent pour l'instant que pour l'utilisation d'un user-setup unique. Ce fichier est un fichier ASCII contenant des listes au format 'Mathematica'. Il contient les informations suivantes :

1. **Les spécifications des user setup**
   C'est une matrice contenant n lignes (n étant le nombre de user setup) et 5 colones. Chaque vecteur en ligne (qui correspond à 1 user setup) est structuré de la sorte :

   { Fondamental, Nombre d'harmonique, dynamique, { }, { } }

   La dynamique est un coefficient relatif à la longueur d'enregistrement des ondes échantillonnées par les cartes ADC. Exemple : {{2.15*^9, 9, 0, {}, {}}}.

2. **Les états internes du LSNA correspondant au user-setup**
   C'est une matrice de n lignes et de 6 colonnes. Chaque ligne est de la forme :

   { Fondamental, FracN, Nombre de points ADC, { Positions des raies compressées },
   { Positions des atténuateurs }, { Echelle des ADC }, { { },{ } } }

Les positions sont des sous matrices. La position des raies compressées est l'emplacement des raies harmoniques dans la FFT issue des ADC. Elle est évidemment liée à la fréquence du Fondamental (1er paramètre) et du FracN (2ème paramètre).
Exemple :
{{2.15*^9, 1.990351924189815*^7, 2048, {86, 172, 258, 344, 430, 516, 602, 688, 774}, {{0, 0, 0, 0}, {0.0625, 0.0625, 0.0625, 0.0625}}, {{}, {}}}}

3. Les matrices d'erreurs du LSNA

La figure 26 illustre par un exemple comment sont stockées les informations relatives aux matrices d'erreurs du LSNA. Ces matrices d'erreurs sont les relations linéaires entre les quantités {v1, i1, v2, i2} et les mesures brutes non corrigées {r1, r2, r3, r4}.
L'information d'étalonnage est stockée dans un vecteur Mathematica. Ce vecteur contient les matrices d'erreurs correspondant à chaque user-setup.
Pour chaque user-setup, on dispose d'autant de matrices d'erreurs qu'il y a d'harmoniques considérées. Pour chaque fréquence, l'information enregistrée est la suivante :

{ Matrice de Calibrage, Null, Null }

soit :

$$\{ \{ \{\alpha_1, \beta_1, 0, 0\} , \{\gamma_1, \delta_1 , 0, 0\} , \{0, 0, \alpha_2, \beta_2\} , \{0, 0, \gamma_2, \delta_2 \} \} , \text{Null} , \text{Null} \}$$

4. Résistance DC des ports 1 et 2

C'est un vecteur de 2 éléments qui permet d'avoir de façon calibrée les tensions et courants au continu lors des mesures. Cela nécessite une alimentation Agilent que nous n'avons pas au CNES. Cette information n'est donc pas utilisée dans notre application.

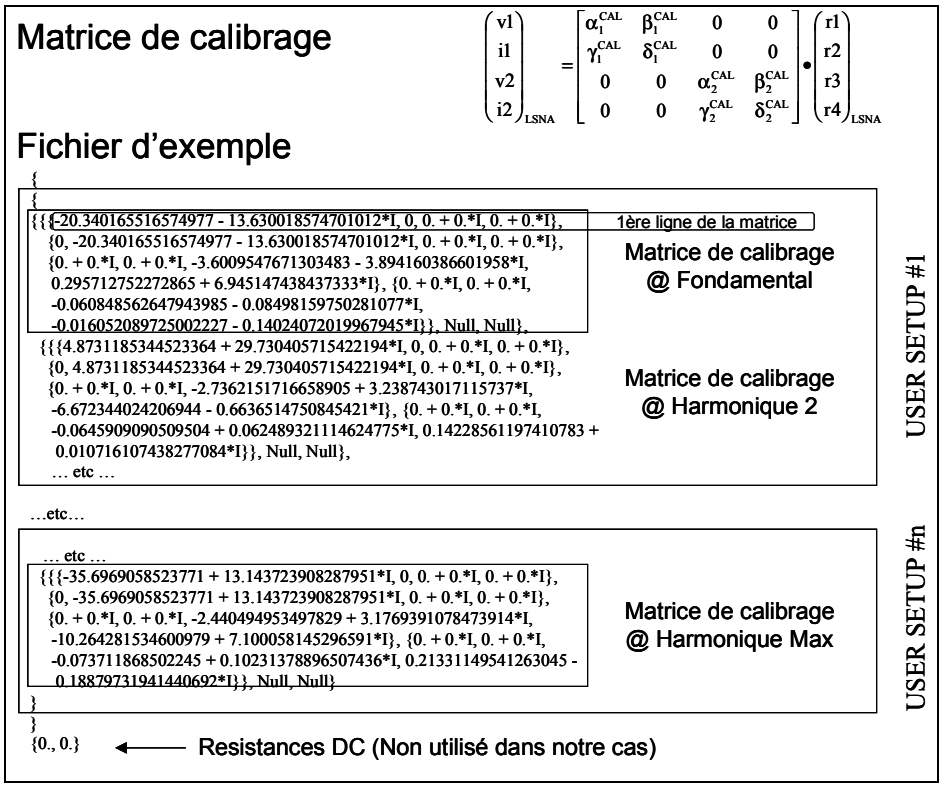


Fig. 26 - Illustration par l'exemple de la structure des matrices d'erreur dans un fichier de calibrage du LSNA

Correction « temps réel » des mesures par le LSNA

La matrice d'erreur obtenue lors de l'étalonnage du LSNA n'est valable que lorsque les atténuateurs programmables internes du 'downconverter box' sont en position minimale (environ 0 dB).
Or, au fil des mesures les positions de ces atténuateurs changent. En effet, à chaque instruction « Measure; » sous l'environnement 'Mathematica', le logiciel va contrôler le niveau obtenu en sortie des convertisseurs analogique/numérique. Si ce niveau est trop élevé, une incrémentation du positionnement des atténuateurs programmables est opérée.
Les positions possibles des atténuateurs programmables vont d'environ 0 dB à 40 dB par pas de 10 dB. Les paramètres [S] des ces atténuateurs programmables ont été pré-caractérisés très précisément lors de l'assemblage du 'downconverter-box' par Agilent. En les considérant de manière relative, il devient possible de déduire la matrice d'erreur du LSNA pour une position d'atténuateur quelconque en connaissant celle issue de la configuration d'étalonnage (0 dB).

L'utilisateur peut contrôler à tout moment la matrice d'erreur du LSNA. On peut ainsi comprendre expérimentalement et très facilement les transformations de matrice d'erreur produites par le programme 'Mathematica' pilotant le LSNA.

La matrice d'erreur du LSNA est disponible en « temps réel » à travers la variable NNMS`CalMatrix.

La mise en évidence des transformations de matrices d'erreurs (gestion automatique de la correction des atténuateurs programmables) peut être faite de la façon suivante :

- On réalise un étalonnage. La matrice d'erreur est sauvegardée dans un fichier semblable à celui décris dans le paragraphe précédent.
- On réalise une mesure simple (CW à 0 dBm sur une ligne par exemple) en s'assurant que les atténuateurs internes sont à 0 dB.

```
SelectRange[1,1]
Ranges
        {0.794328, 0.794328}
Measure
```

- On contrôle alors la matrice de calibrage courante qui établie les relations entre {v ; i } et les données brutes {r1, r2, r3, r4}.

```
NNMS`CalMatrix
```

La matrice est strictement identique à celle qui a été sauvegardée en fichier lors de la procédure d'étalonnage. On réalise maintenant la même opération lorsque la configuration des atténuateurs programmables est différente (dans cet exemple, le port 1 est atténué d'environ 10 dB et le port 2 d'environ 20 dB).

```
SelectRange[2,3]
Ranges
        {2.51189,7.94328}
Measure
NNMS`CalMatrix
```

Dès lors la matrice d'étalonnage est différente de celle sauvegardée dans le fichier d'étalonnage. Les variations relatives des S_{21} des atténuateurs programmables par

rapport à leur position nominale est prise en compte de façon transparente pour l'utilisateur.

Génération d'un fichier de calibrage hybride pour le LSNA

Le fichier de calibrage, afin d'être compatible avec l'environnement 'Mathematica' du LSNA et de permettre une correction automatique des atténuateurs programmables, doit être obtenu pour des positions nominales d'atténuateurs internes du 'downconverter box'.

La matrice de calibrage est constituée de 2 sous-matrices : chacune d'entre elle correspondant aux erreurs obtenues sur les ports 1 et 2 du LSNA.
Avec la configuration adoptée du LSNA associé aux sondes HIP, la sous matrice d'étalonnage du port 2 est celle obtenue par une procédure d'étalonnage conventionnelle LRRM ou SOLT. Cette procédure se fait donc pour des atténuateurs en position minimale sur les entrées 3 et 4 du 'downconverter box'.

En revanche il convient de s'assurer personnellement de l'atténuation minimale des atténuateurs reliés aux entrées 1 et 2 du 'downconverter box' lors de la détermination de la sous matrice relative au port 1 (caractérisant les sondes HIP).
Pour l'étalonnage du port 1, nous avons considéré 3 méthodes. Les procédures de caractérisations différent suivant les méthodes : détermination de $\widetilde{K}(f)$ pour les méthodes 1 et 2 ou obtention directe de la sous matrice reliant les quantités $\begin{pmatrix} v1 \\ i1 \end{pmatrix}$ et $\begin{pmatrix} r1 \\ r2 \end{pmatrix}$ pour la 3ème méthode.
Quelque soit la méthode utilisée, il conviendra, dans un premier temps, de former la sous matrice d'étalonnage relative au port 1 puis, dans un deuxième temps, d'associer cette nouvelle expression à la sous matrice d'étalonnage relative au port 2 obtenue lors de l'étalonnage classique LRRM.

Cette gestion des fichiers de calibrage est particulièrement intéressante car elle permet de disposer d'un système de mesure calibré hybride. On peut par exemple disposer d'un port 1 sous pointes GSG et d'un port 2 en SMA.
Cela peut permettre la mesure des paramètres [S] d'une sonde de façon immédiate et extrêmement simple, alors que traditionnellement cette mesure requière une procédure lourde de de-embedding [2] [7].

Dans le cas particulier de l'utilisation des sondes HIP avec le LSNA, des procédures de génération de matrices de calibrage hybrides ont été développées et inclues dans le module « HIP` » :

Caractérisation des sondes

DefineK1[nombre d'harmoniques]
Caractérisation de l'entrée n°1 du 'downconverter box' : K1 (méthode 1 et 2).

DefineK2[nombre d'harmoniques]
Caractérisation de l'entrée n°2 du 'downconverter box' : K2 (méthodes 1 et 2).

Cal2HIP[nombre d'harmoniques]

Caractérisation des 2 sondes HIP couplées (méthodes 3).

Génération de la sous-matrice de calibrage du port 1

MakeSubMatrix1[K1,K2]

Création de la sous matrice par la méthode 1. K1 et K2 sont des vecteurs préalablement déterminés dont la dimension correspond au nombre d'harmoniques.

MakeSubMatrix1[K1,K2,γL,Z_C]

Création de la sous matrice par la méthode 2. Les 4 paramètres d'entrée dont des vecteurs de même taille (grille de fréquences harmoniques). Z_C est l'impédance caractéristique de la ligne sondée en fonction de la fréquence. ,γL est le produit de la constante de propagation par la distance entre les sondes HIP exprimé en fonction de la fréquence.

La fonction DefineGammaL1[fondamental,nombre d'harmonique,delay,L] , également présente dans le module HIP`permet d'avoir une estimation rapide du produit γL en supposant que la portion de ligne sondée ne présente qu'un retard linéique 'Delay' exprimé en ps/cm. L est exprimé en mm.

MakeSubMatrix3[Sous-matrice d'étalonnage,γL_{CAL},$Z_{C\ CAL}$, ,γL_{MESURE},$Z_{C\ MESURE}$]

Transformation de la sous-matrice de calibrage obtenue lors de l'étalonnage par la méthode 3 lorsque les conditions de mesure et d'étalonnage sont différentes.

Génération de la matrice d'étalonnage unique du système complet (LSNA+HIP)

GenerateHybridCalFile[nom fichier de cal 1, sous-matrice , nom fichier de cal 2]

Charge les coefficients de l'étalonnage complet et classique sauvegardés dans le fichier de calibrage n°1. La sous-matrice correspondant au port 1 est alors remplacée par la sous-matrice précédemment calculée. Le tout est sauvegardée sur disque dans un fichier de calibrage numéro 2.

Le listing de cette procédure proposé en annexe n'est valable que lorsque l'étalonnage classique ne contient qu'un seul « user-setup ».

Ce fichier de calibrage numéro 2 est entièrement compatible avec les fichier de calibrage traditionnels. Une fois la phase d'étalonnage achevée, l'utilisateur démarrera sa phase de mesure par l'instruction habituelle : LoadCal[fichier de cal 2].

Des exemples d'utilisation sous 'Mathematica' de ces nouvelles fonctions seront présentés en fin de chapitre.

Avantages des méthodes utilisées

Le fait de modifier le fichier des coefficients de calibrage possède de nombreux avantages.

En premier lieu, cela à permis de faire un développement rapide d'une nouvelle procédure d'étalonnage afin de conserver les corrections automatiques relatives aux atténuateurs programmables internes du LSNA.

Ensuite, l'environnement classique de pilotage du LSNA reste strictement le même. Une fois le calibrage effectué, l'utilisateur réalise ses mesures de façon strictement identique à celle employée par l'usage d'un test-set classique.

Enfin, cela permet de conserver toutes les fonctionnalités de l'environnement 'Mathematica' du LSNA :

- mesures quasi-temps réel des formes d'ondes ;
- choix des synchronisation temporelles (forme d'onde et fréquence), même sur les formes issues des sondes HIP ;
- l'usage des data-sets pour la gestion et la sauvegarde des mesures reste possible.

Comparaison des matrices d'erreurs issues des méthodes 2 et 3

Ce paragraphe propose un début d'investigation comparative entre les procédés d'étalonnage correspondants aux méthodes 2 et 3.
On s'intéressera aux termes de la sous-matrice de calibrage associée au port 1 :

$$\begin{pmatrix} v1 \\ i1 \end{pmatrix} = \begin{bmatrix} \alpha & \beta \\ \gamma & \delta \end{bmatrix} \cdot \begin{pmatrix} r1 \\ r2 \end{pmatrix}$$

Mesures expérimentales

Après avoir étalonné le système LSNA conventionnel sur un kit LRRM 'Cascade', on mesure une grande ligne coplanaire (environ 5 mm). La seconde partie de l'étalonnage se fait sur cette nouvelle ligne. Le standard en tension est imposé par la mesure obtenue via le port 2 (sonde GSG). On réalise alors les 2 techniques possibles de caractérisation des sondes HIPs : la détermination de $\widetilde{K}(f)$ et la sous-matrice à 4 termes d'erreurs caractérisant les sondes couplées pour L = 3.175mm (la distance entre les sondes).
A partir de ces caractérisations des sondes HIPs, on peut en déduire les sous matrices d'étalonnage qui nous intéressent.

La sous matrice correspondant à la méthode 3 est obtenue directement lors de l'étalonnage. Celle correspondant à la méthode 2 fait intervenir des paramètres supplémentaires : la connaissance exacte de la distance entre les sondes, l'impédance caractéristique et la constante de propagation. Si la distance L est connue, les 2 autres paramètres sont estimés et sont à l'origine des différences entre les coefficients des sous-matrices d'erreurs obtenus par les méthodes 2 et 3.

Le synoptique des opérations réalisées est représenté figure 27.

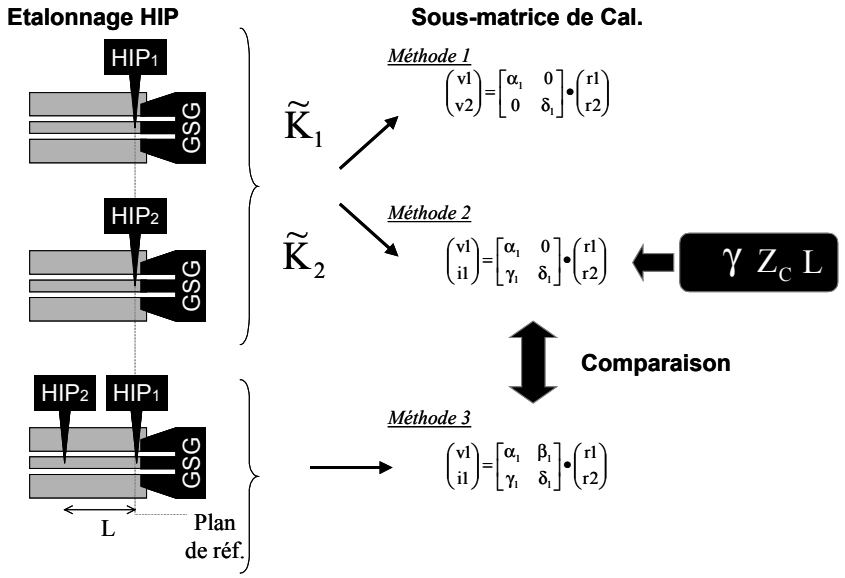


Fig. 27 - Procédures d'étalonnages des sondes HIP réalisées pour permettre la comparaison des méthodes 2 et 3.

Résultats

Les figures 28, 29 et 30 présentent les coefficients d'erreurs obtenus lors des 2 types d'étalonnages (sondes couplées ou indépendantes) ainsi que ceux issus de la sous matrice de calibrage via la méthode 2.

On pourra constater que dans cette configuration (L = 3.175mm) le couplage entre les sondes est d'environ 20 dB. Ce couplage dépend très étroitement de la distance entre les sondes et devient plus important lorsque cette distance diminue.

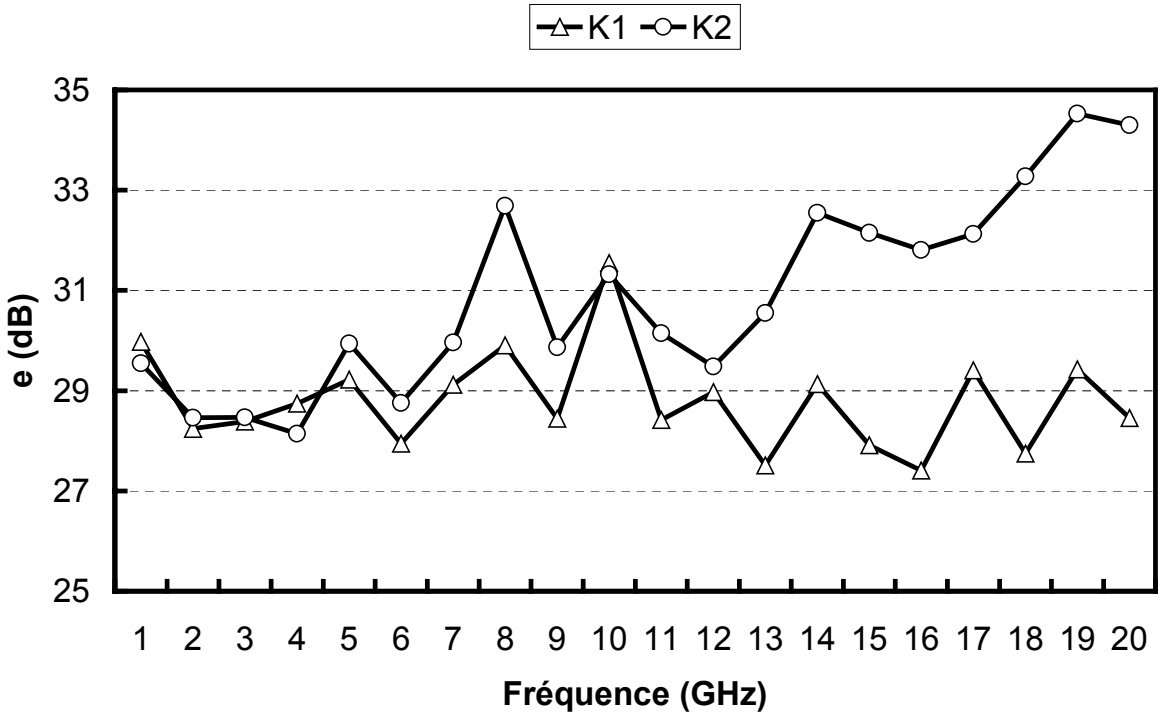


Fig. 28 - Modules en dB des réponses en fréquence des sondes HIP. Cela correspond aux termes d'erreurs α et δ (les autres termes étant nuls) dans la sous matrice d'étalonnage produite par la méthode 1.

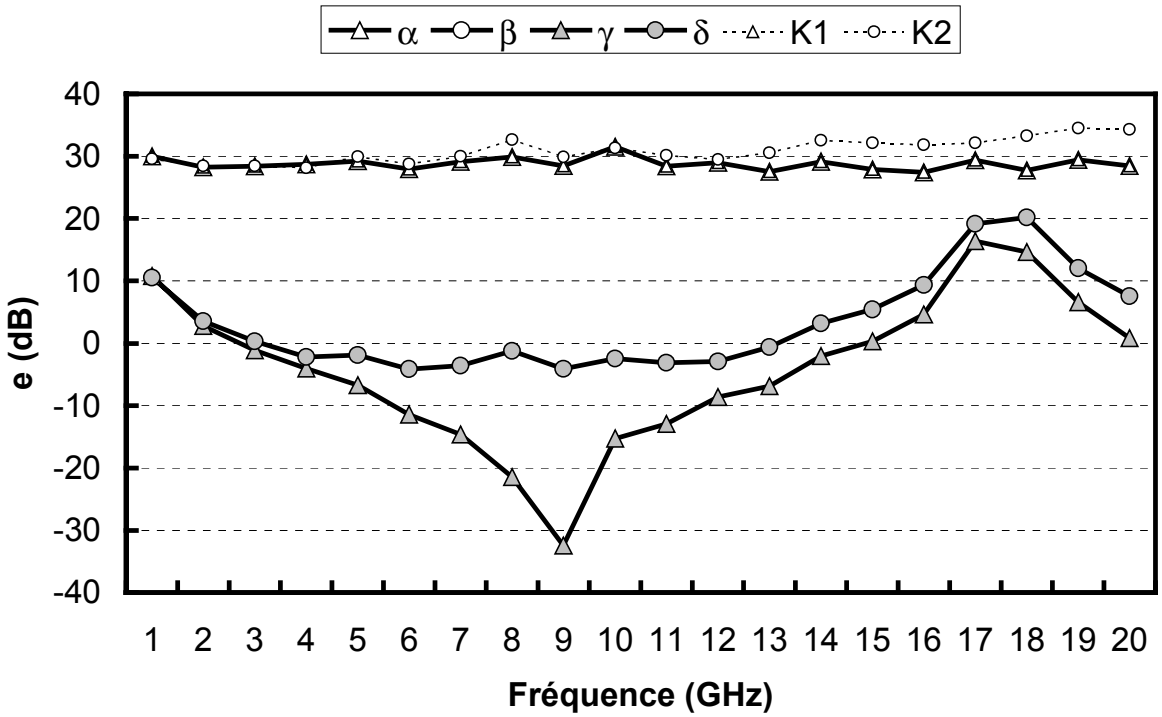


Fig. 29 - Méthode 2 : Représentation graphique des modules en dB des 4 termes d'erreur de la sous matrice d'étalonnage. Le terme β n'est pas représenté car il est nul. Dans cette configuration $\alpha=K_1$. Les autres termes sont <u>estimés</u> tel que $Z_C=50\Omega$, L=3.175mm (10 tours) et une constante de propagation sans pertes correspondant à un retard linéique de 90ps/cm.

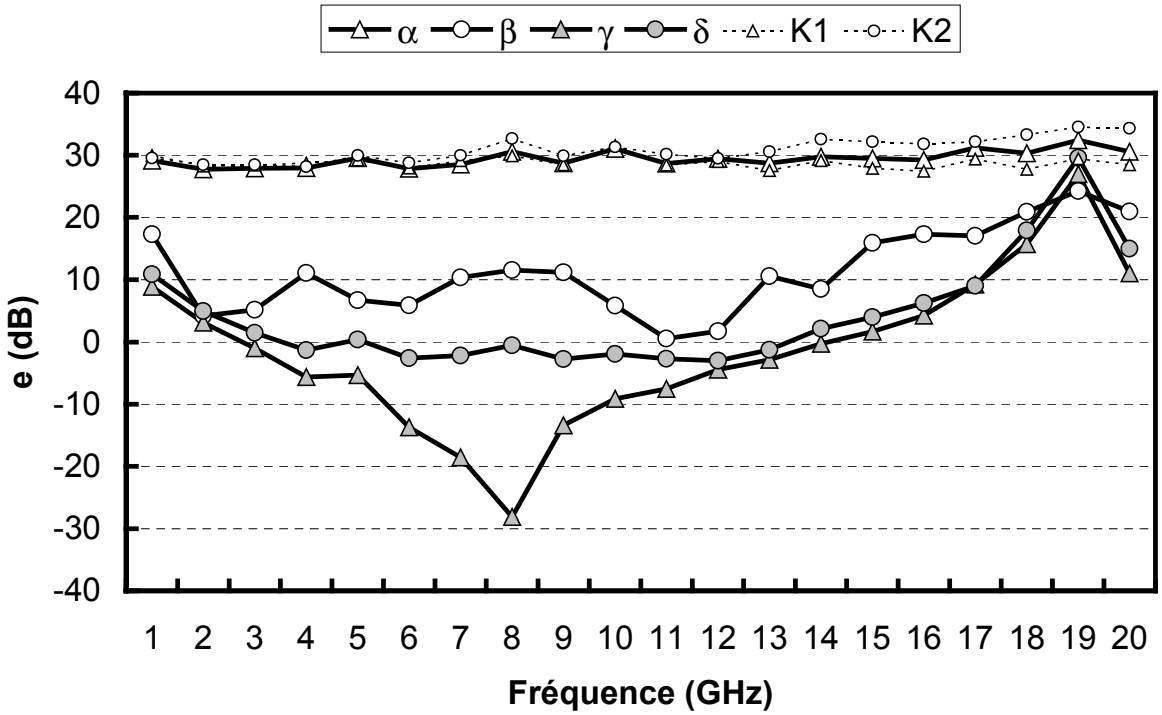


Fig. 30 - Méthode 3 : Les termes sont déduit de l'étalonnage réalisé pour L=3.175mm (10 tours). Aucune estimation n'est nécessaire. On constatera le couplage (β) entre les sondes (pour cette valeur de L) vaut environ 20 dB.

Utilisation sous ‘Mathematica’

Ce paragraphe propose 3 exemples d’étalonnage de sondes HIP sous l’environnement ‘Mathematica’ pilotant le LSNA. Chaque exemple illustre une méthode d’étalonnage précédemment décrite.

Les premières lignes de commandes sont communes aux 3 méthodes. Après avoir fait un calibrage conventionnel du LSNA, on impose comme référence la mesure de tension issue du port 2. Il faut s’assurer que les atténuateurs internes du port 1 soient toujours à leur niveau minimal durant le calibrage des HIP.

```
LoadCal["D:\\Public\\Tibo\\CAL_LRRM.m"]

SetPhaseReferenceQuantity["v2"];

PhaseReferenceFrequency = 1;

SetSourcePower[0]; SetSourceFreq[Fundamental]; SetSourceOutput[On];
AutoRange;

Ranges // dB

{-2., -2.}
```

Méthode 1

Pour les méthodes 1 et 2, les sondes HIP sont caractérisées séquentiellement de façon indépendante en les plaçant au plus près du plan de référence de l’étalonnage conventionnel du LSNA (milieu de la ligne « thru » pour le pied de test, ou au plus près de la sonde GSG pour le banc ‘Cascade’).
L’exemple ci-dessous est un étalonnage de 2 sondes HIP suivant la méthode 1 pour 9 harmoniques. Le fichier de calibrage final est alors « CAL_HYBRID.m ». Il suffira simplement de la charger grâce à l’instruction LoadCal[] pour disposer d’un instrument de mesure LSNA+HIP totalement calibré. L’onde nommée v1 correspondra alors à l’onde de tension mesurée (et corrigée) par la première sonde. L’onde nommée i1 quand à elle n’est pas un courant mais l’onde de tension corrigée mesurée par la seconde sonde.

```
SetSourceOutput[On];

myK1 = DefineK1[9];

myK2 = DefineK2[9];

myMatrix = MakeSubMatrix1[myK1, myK2];

GenerateHybridCalFile["D:\\Public\\Tibo\\CAL_LRRM.m", myMatrix,
  "D:\\Public\\Tibo\\CAL_HYBRID.m"];

SetSourceOutput[Off];

SelectRange[5, 5];
```

Méthode 2

L'utilisation de la seconde méthode est semblable à la première expliquée précédemment. L'exemple ci-dessous correspond à un étalonnage comportant 20 harmoniques dont le fondamental est à 1 GHz. Pour l'usage de cette méthode, il faut spécifier les vecteurs Z_C (ici uniformément égal à 50Ω) et γL (ici un retard linéique de 105.2446 ps/cm pour une distance L=0.3175mm).

```
SetSourceOutput[On];

myK1 = DefineK1[20];

myK2 = DefineK2[20];

myZc = Table[50, {i, 20}];

myGL = DefineGammaL1[1*10^9, 20, 105.2446, 0.3175];

myMatrix = MakeSubMatrix2[myK1, myK2, myGL, myZc];

GenerateHybridCalFile["D:\\Public\\Tibo\\CAL_LRRM.m", myMatrix,
  "D:\\Public\\Tibo\\CAL_HYBRID.m"];

SetSourceOutput[Off];

SelectRange[5, 5];
```

Avec cette méthode, v1 et i1 correspondent bien aux tension et courant corrigés dans le plan de la pointe HIP numéro 1.

Méthode 3

La 3ème méthode fournie directement la sous-matrice d'étalonnage. Cette sous matrice n'est valable que si les conditions de mesures sont rigoureusement identiques à celles de l'étalonnage.
L'instruction Cal2HIP[nombre d'harmonique] invite l'utilisateur à changer les standards présents sur le port « RF IN 2 » du Test-Set.

```
SetSourceOutput[On];

myMatrix = Cal2HIP[20];

GenerateHybridCalFile["D:\\Public\\Tibo\\CAL_LRRM.m", myMatrix,
  "D:\\Public\\Tibo\\CAL_HYBRID.m"];

SetSourceOutput[Off];

SelectRange[5, 5];
```

Lorsque les conditions de mesure ne sont pas les mêmes que les conditions de l'étalonnage, il convient de transformer la sous-matrice de calibrage avant de lancer l'instruction GenerateHybridCalFile[]. Ceci est faisable très facilement au moyen de la commande MakeSubMatrix3[].

{{Nouvelle sous-matrice}} =
MakeSubMatrix3 [{{Sous matrice d'étalonnage}} , $\{\gamma L_{CAL}\}$, $\{Z_{C\ CAL}\}$, $\{\gamma L_{MESURE}\}$,
$\{Z_{C\ MESURE}\}$]

Il convient alors de connaître très précisément les impédances caractéristiques et les constantes de propagations des 2 substrats utilisés.

Notons que la 3ème méthode peut être plus amplement développée afin de devenir une méthode à part entière pour déterminer la constante de propagation (nécessite de nombreuses positions de mesure pour la HIP) et l'impédance caractéristique (nécessite un 3ème standard dans le plan 'RF IN 2' du test-set afin de solutionner un système dont l'impédance caractéristique est inconnue).

Le temps a profondément manqué pour poursuivre plus en avant ces aspects de métrologie, d'autant plus qu'ils s'apparentent plus aux besoins de laboratoires comme le NIST qu'aux réels besoins ressentis par le CNES et l'IRCOM actuellement.

MESURES APPLICATIVES

Nous disposons à ce stade d'un système de mesure de forme d'ondes temporelles au moyen de sondes HIP. Ce système possède une procédure d'étalonnage rigoureuse qui a fait l'objet des chapitres précédents.
Dans ce nouveau chapitre, nous allons voir comment réaliser des mesures corrigées par sondes HIP positionnées à des emplacements arbitraires à l'intérieur même de circuits hybrides ou MMICs.
Ce chapitre se décompose en 3 parties.
Une première partie expliquera de façon explicite comment obtenir des mesures de formes d'ondes temporelles corrigées via sondes HIP. Cette explication se fera à travers un script d'exemple utilisé dans l'environnement 'Mathematica' du LSNA pour une campagne de mesure.
Une seconde partie traitera d'une campagne de mesure dont l'objet à été la vérification d'une topologie optimale d'un amplificateur en classe haut-rendement (classe F) : il s'agit d'une application de notre système de mesure sur la vérification et la validation d'une conception MMIC.
Enfin la troisième et dernière partie s'attardera sur l'analyse de phénomènes dynamiques non-linéaires d'un amplificateur présentant de nombreux comportements paramétriques face à une simple excitation en CW : il s'agit d'une application axée sur l'analyse de stabilité.

Mesurer des ondes de tensions internes à un circuit MMIC ou hybride

Pour pouvoir mesurer des formes d'ondes temporelles avec le système LSNA associé aux sondes HIP, il suffit de suivre un certain nombre d'étapes. Ce paragraphe n'est autre qu'un exemple commenté d'un script 'Mathematica' permettant des mesures de circuit MMIC. L'étalonnage se fait suivant la méthode 1 : les sondes HIP sont indépendantes et peuvent être placées de façon arbitraire sur les véhicules de test. En contre partie nous ne disposons que de la forme temporelle de la tension au point de mesure : nous n'avons accès ni au courant ni aux ondes de propagation.

1. Initialisation du LSNA

L'initialisation du LSNA est classique. Il faut juste penser à intégrer le module « HIP` ».

```
<<D:\Public\Tibo\Masterfile.m;
OpenModule["General`","NNMS`","Source`","CalibrationMgr`","VNA`","HIP`"];
```

2. Calibrage du LSNA en LRRM

La seconde étape consiste à réaliser un étalonnage conventionnel du LSNA. Dans cet exemple, il s'agit d'un LRRM sous pointes pour 9 harmoniques dont le fondamental et à 2.15 GHz. La matrice d'erreur de cet étalonnage est sauvegardée dans le fichier « CAL_LRRM.m ».

```
StartCal["LRRM`",{2.15*10^9,9,0}]
SaveCal["D:\\Public\\Tibo\\CAL_LRRM.m"]
```

3. Mesure avec le LSNA calibré en LRRM

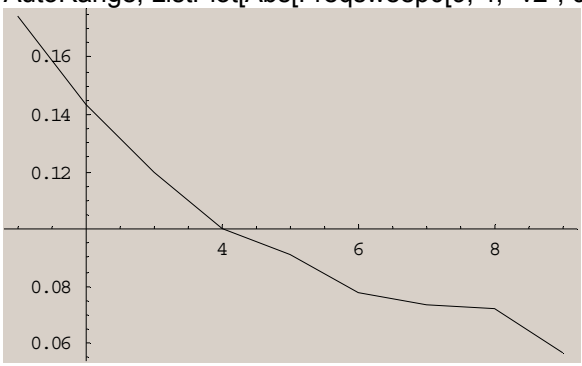
Une fois l'étalonnage réalisé, on se place en configuration mesure. Le LSNA se trouve toujours en configuration conventionelle, sous pointes GSG, en présente de la ligne « thru ».

```
LoadCal["D:\\Public\\Tibo\\CAL_LRRM.m"]
SetPhaseReferenceQuantity["v2"];
PhaseReferenceFrequency=1;
```

3.1 Vérification du calibrage LRRM

La première chose à réaliser et de vérifier l'étalonnage (ici LRRM) du LSNA. Un 'sweep-sin' doit conduire à une courbe décroissante en fonction de la fréquence : c'est la réponse du « test-set ».

```
SetSourcePower[0]; SetSourceFreq[Fundamental]; SetSourceOutput[On];
AutoRange; ListPlot[Abs[Freqsweep0[9, 4, "v2", 0]], PlotJoined -> True, PlotRange -> All];
```




```
SetSourcePower[-20]; SetSourceOutput[Off];
```

3.2 Correction de la matrice d'erreur LRRM

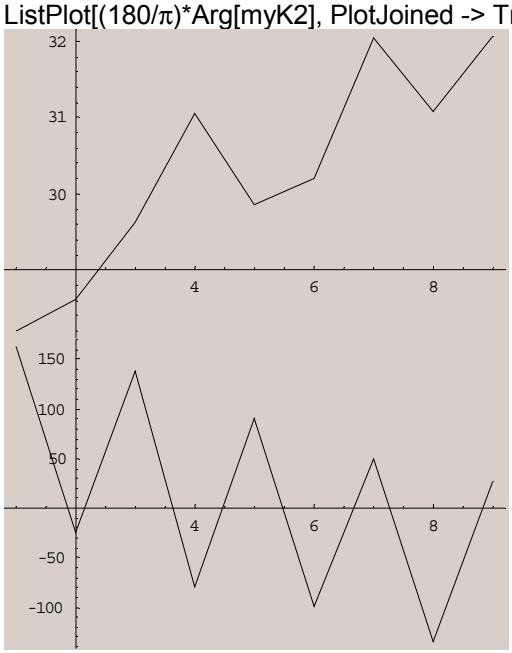
Si la courbe présente des pics ou ne parait pas « lissée », il est conseillé de pratiquer un second étalonnage (retour à l'étape 2). Si la courbe est identique à nouveau il est probable que l'étalonnage soit correct. Si les courbes diffèrent, on peut pratiquer alors plusieurs étalonnages puis construire artificiellement une matrice d'erreur correcte à partir de ces différents étalonnages au moyen de la fonction LRRMDebugCalfile[]. Ceci à été expliqué en début de rapport.

```
LRRMDebugCalFile["D:\\Public\\Tibo\\CAL_LRRM_1.m", "D:\\Public\\Tibo\\CAL_LRRM_2.m", 4,
   "D:\\Public\\Tibo\\CAL_LRRM.m"];
```

4. Etalonnage des sondes HIPs

Il vient ensuite l'étalonnage des sondes HIP. Les entrées 1 et 2 du 'downconverter box' sont déconnectées du 'test-set' puis reliées aux sondes HIP. Il faut penser à placer un 'DC block' si une tension DC est attendue. Ensuite les sondes sont positionnées indépendamment et séquentiellement dans le plan de référence (au plus près de la sonde GSG du port 2) afin d'être caractérisées. Cette manipulation se fait sur la ligne « thru » du kit d'étalonnage LRRM. Il faut vérifier impérativement que les atténuateurs internes du 'downconverter box' relatifs au port 1 sont à leurs position minimale (on peut utiliser la commande « Ranges » qui doit donner environ 0 dB, ou la commande « RangeIndex » qui doit renvoyer « 1 »).
Un nouveau fichier de calibrage est finalement généré : « CAL_HYBRID.m ».

```
SetSourceOutput[On];
AutoRange;
Ranges //dB
{-2.,-2.}
myK1=DefineK1[9];
myK2=DefineK2[9];
ListPlot[20*Log[10, Abs[myK2]], PlotJoined -> True, PlotRange -> All];
ListPlot[(180/π)*Arg[myK2], PlotJoined -> True,   PlotRange -> All];
```



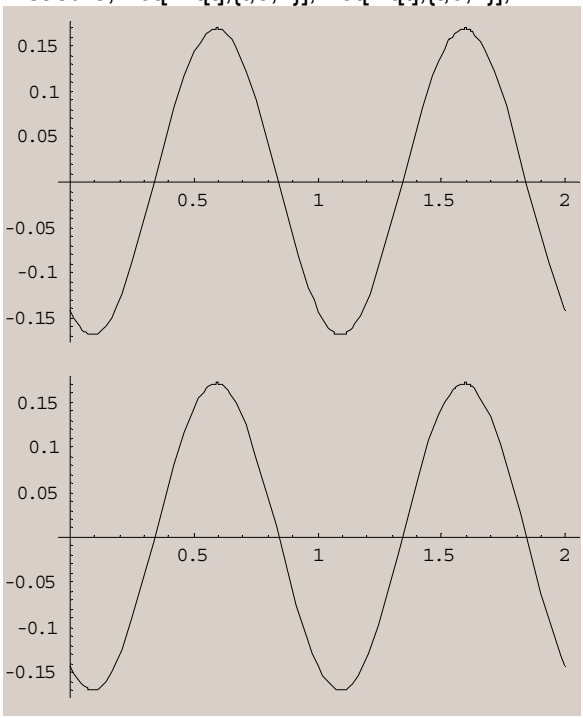

```
myMatrix=MakeSubMatrix1[myK1,myK2];
GenerateHybridCalFile["D:\\Public\\Tibo\\CAL_LRRM.m",myMatrix,
   "D:\\Public\\Tibo\\CAL_HYBRID.m"];
SetSourceOutput[Off]
```

5. Mesure via sonde HIP sur le substrat d'etalonnage

Juste après l'étalonnage des sondes HIP, on peut vérifier la cohérence entre les mesures offertes par la sonde GSG du port 2 et la 2ème sonde HIP toujours positionnée dans le plan de référence.
La nouvelle matrice d'erreur doit donc être chargée dans l'environnement 'Mathematica', et les paramètres de synchronisation mis à jour.

Utilisant la méthode 1, la tension dans le plan de la pointe HIP peut être consultée à travers les commandes relatives à i1.

```
LoadCal["D:\\Public\\Tibo\\CAL_HYBRID.m"]
SetPhaseReferenceQuantity["v2"];
PhaseReferenceFrequency=1;
SetSourcePower[0];SetSourceFreq[Fundamental];SetSourceOutput[On];
AutoRange;
Measure;Plot[v2t[t],{t,0,2}];Plot[i1t[t],{t,0,2}];
```




```
Ranges //dB
{-2.,-2.}
RangeIndex
{1,1}
SelectRange[5,5];
Ranges //dB
{38.,38.}
SetSourcePower[-20];SetSourceOutput[Off];
```

6. Mesures via sonde HIP sur un circuit MMIC

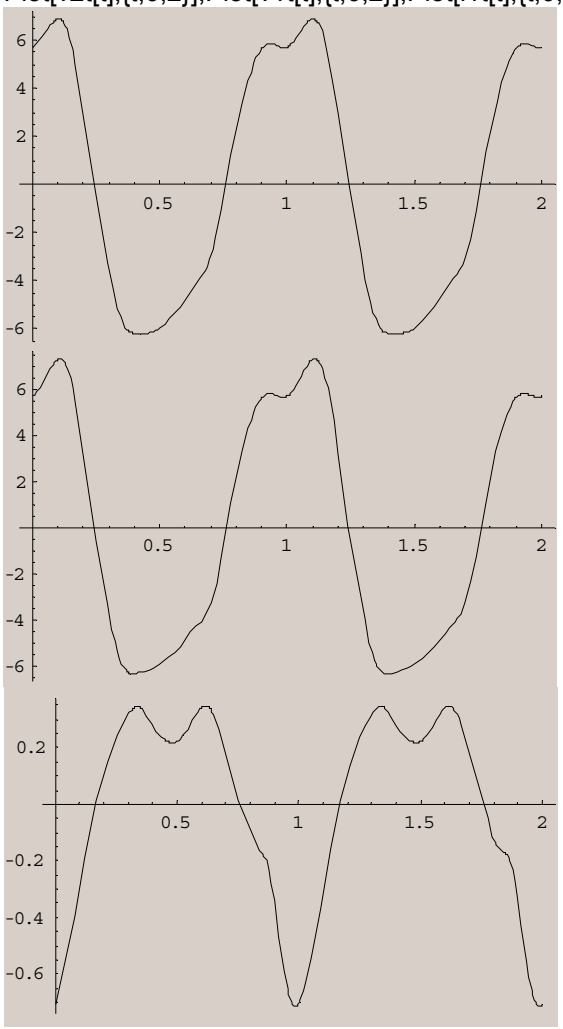
Un circuit MMIC peut dès lors être mesuré. Dans cet exemple la sonde HIP n°1 (v1) mesure la tension dans le même plan que la sonde GSG du port 2 (v2) ; la sonde HIP n°2 (i1) mesure la tension en entrée d'un transistor HBT. Le DC n'est pas inclus, ce qui explique que les formes d'ondes ont une valeur moyenne nulle.

Avant chaque déplacement de sonde HIP, il faut penser à remettre les atténuateurs programmables dans leur atténuation maximale (environ 40 dB) via l'instruction SelectRange[]. Cela permet d'éviter des risques destructifs sur le 'downconverter box' lorsque l'on passe de l'entrée à la sortie d'un amplificateur de puissance par exemple.

L'utilisation des 'Data-Set' reste possible. Cela permet d'organiser facilement une campagne de mesure [6]. Mais il faut faire attention : d'une part les ondes a et b, également sauvegardées dans les data-set, ne sont pas physiques car « i1 » traduit une tension et non pas le courant du port 1 ; d'autre part les fonctions associées au puissances ne sont plus valables.

Il faut donc bien noter ce qui se trouve dans les ‘data-set’ car leurs réutilisation ultérieure n’est pas immédiate.

```
SelectRange[5,5];
SetSourcePower[10];SetSourceFreq[Fundamental];SetSourceOutput[On];
Ranges // dB
{38.,38.}
AutoRange;
Measure;
Plot[v2t[t],{t,0,2}];Plot[v1t[t],{t,0,2}];Plot[i1t[t],{t,0,2}]
```




```
CreateNNMSDataSet["MesMesures"]
AddToNNMSDataSet["MesMesures"]

…etc…

SetSourcePower[-20]; SetSourceOutput[Off];
Quit[] ;
```

Vérification d'une topologie optimale pour un circuit MMIC

De nombreux dispositifs non-linéaires ont été investigués par le LSNA associé aux sondes HIP. L'un d'entre eux est un amplificateur à 2 transistors HBT (1 seul étage) optimisé pour une classe de fonctionnement en haut rendement. Il a été conçu par A. Mallet lors de sa thèse de doctorat [8].
L'objet des mesures consiste non seulement à visualiser les formes d'ondes en sortie de la puce par des mesures classique du LSNA via sonde GSG mais également les formes d'ondes à l'intérieur de la puce soit en entrée et sortie des transistors.

Présentation de l'amplificateur

L'amplificateur présente un gain d'environ 15 dB. Sa conception a été optimisée à 2.15 GHz pour une classe F : c'est à dire que les formes temporelles en tension aux sorties des transistors sont quasi-carrées. Cela a été possible en plaçant un réseau d'adaptation qui, entre autre, annule la 2ème harmonique du signal de sortie. En conséquence, les performances des transistors présentent un très haut rendement (plus de 70%).

Le layout du circuit est représenté figure 31. Ses paramètres de fonctionnement sont décris ci-dessous :

<u>Fréquence de fonctionnement</u>
f_0 = 2.15 GHz

<u>Classe de fonctionnement :</u>
Classe B (demi-période de conduction)
Optimisé en Classe F pour f = 2.15 GHz

<u>Polarisation :</u>
Vbeo = 1 volt
Vceo = 9 volts
Ibo = 0 mA
Ico = 0 mA

Banc de mesure

Pour les mesures sur cet amplificateur, nous avons utilisé une source extérieure (Anritsu 69047B) connectée directement à la sonde GSG d'entrée du circuit afin de disposer d'une puissance disponible suffisante à 2.15 GHz. Les niveaux de puissance disponibles qui ont été émis sont : 5, 8, 10, 12, 13, 14 et 15 dBm. Tous les résultats graphiques présentés ultérieurement portants sur cet amplificateur, sont paramétré suivant ces valeurs.
Une seule sonde suffit pour mesurer les formes d'onde de façon séquentielle.
Le banc de mesure utilisé est illustré figure 32.

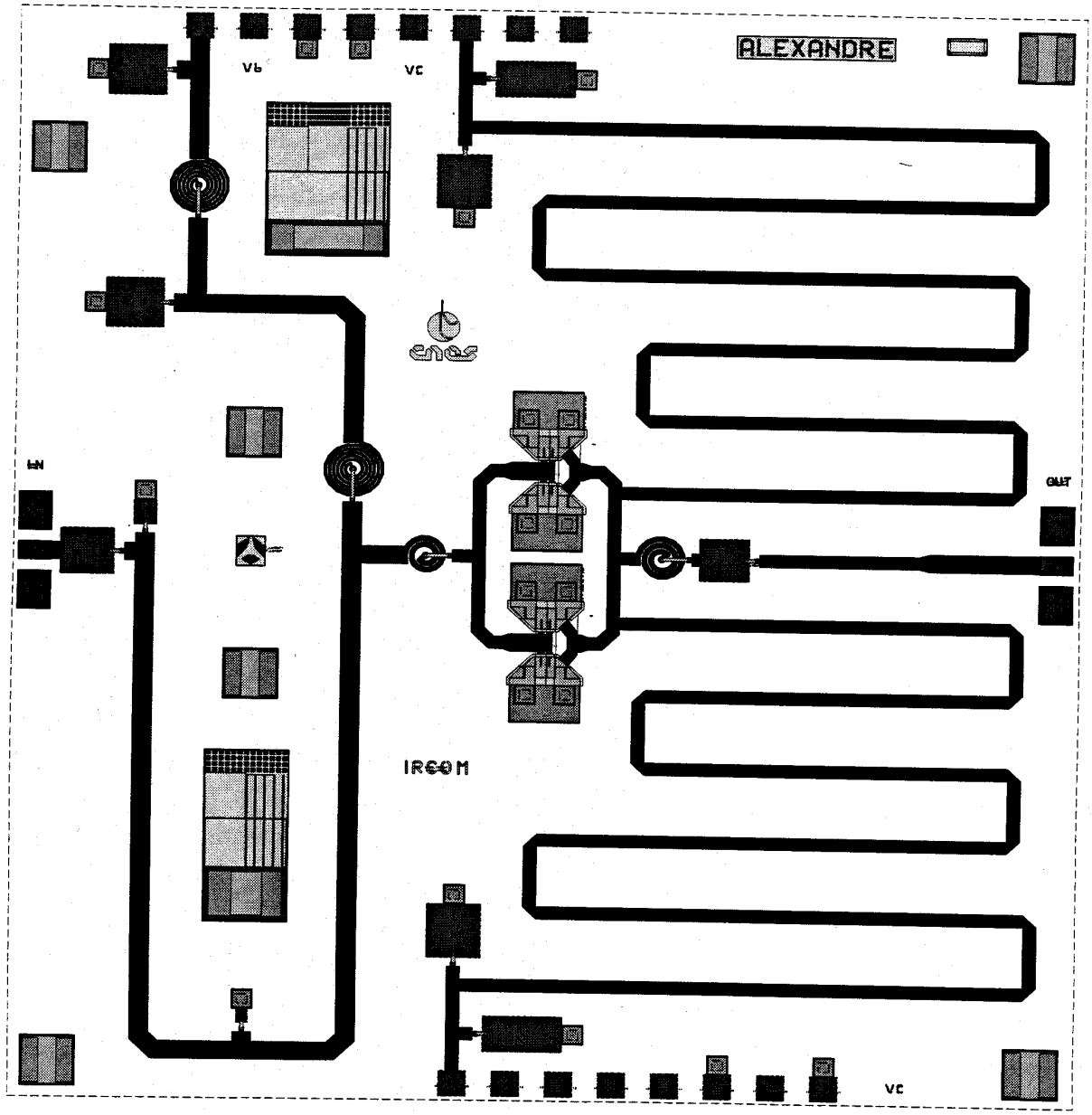


Fig. 31 - Masque du circuit sous test : Amplificateur à 2 transistors HBT optimisés en classe F

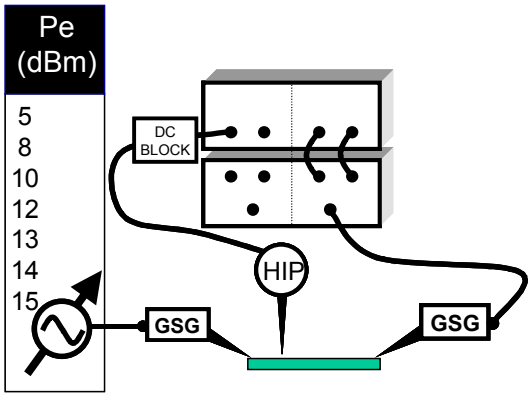


Fig. 32 - Banc de mesure : configuration du système pour les mesures sur l'amplificateur en classe F

Résultats de mesures

On représente toutes les formes d'ondes temporelle de la tension en différents points du circuit pour un fenêtrage temporel de 930ps, soit 2 périodes du fondamental. Les courbes sont paramétrées suivant le niveau de puissance appliqué.

Mesures en sortie du MMIC par la sonde GSG

Les premières mesures effectuées sont celles provenant de la sonde GSG placée en sortie du circuit MMIC. Ces mesures sont corrigées par la sous-matrice d'étalonnage LRRM du LSNA. Ce sont les ondes mesurables par n'importe quel système LSNA. Elle vont donc constituer une base comparative à des mesures faite via sonde HIP.
On constate qu'en sortie de circuit MMIC, même pour une puissance disponible élevée, la forme n'est pas complètement carrée : en effet il existe un filtrage par rapport aux signaux produit par les HBT optimisés en classe F (ondes a priori « carrées »).

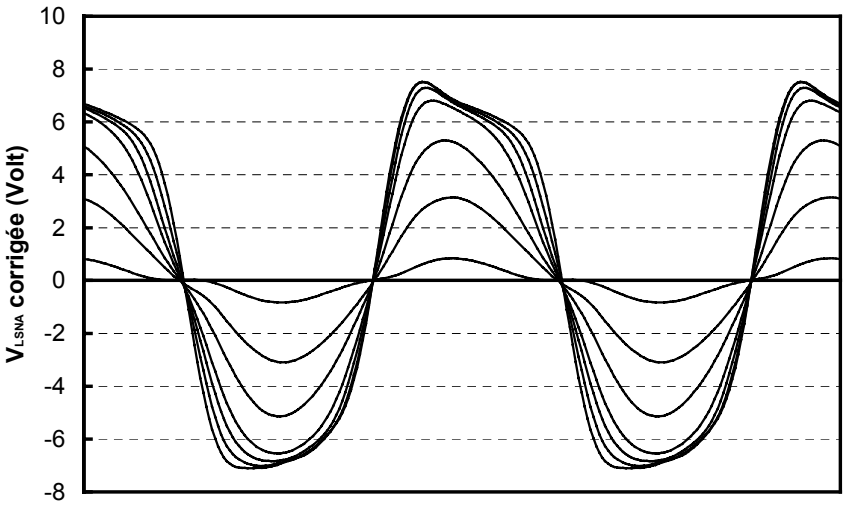


Fig. 33 - Mesures en sortie du circuit MMIC dans le plan de la sonde GSG du port 2.

Mesures en sortie du MMIC par la sonde HIP

Des mesures ont été faites en sortie du circuit MMIC avec une sonde HIP. Le plan de mesure de la sonde HIP est en amont de celui de la sonde GSG. Cela explique les petites différences entre les figures 33 et 34. Cependant, la correspondance des courbes reste forte entre les mesures réalisées par un LSNA conventionnel (figure 33) et celles obtenues par un LSNA associé aux sondes HIP (figure 34).
La classe F n'est pas visualisée car le plan de mesure est encore ici un plan « extrinsèque » alors que l'optimisation lors de la conception se fait dans un plan « intrinsèque ».

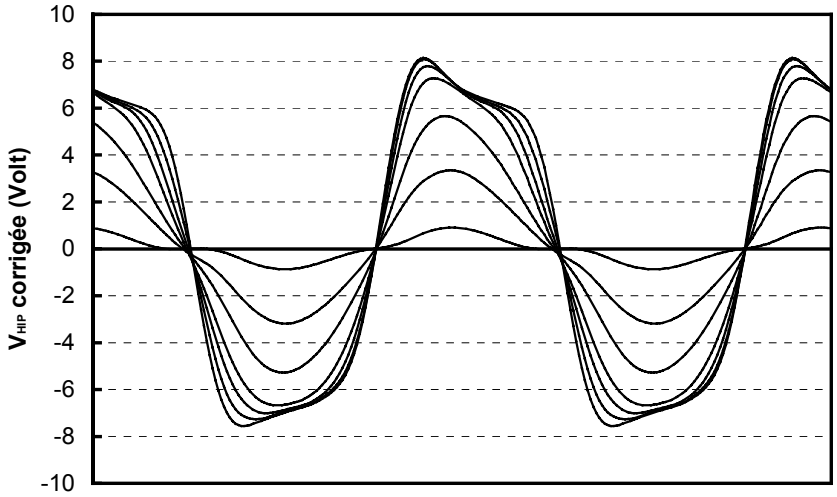


Fig. 34 - Mesures en sortie du circuit MMIC dans le plan de la sonde HIP voisin du plan de la sonde GSG.

Il devient alors intéressant de déplacer la sonde HIP pour vérifier les formes d'ondes dans les plans aux accès même d'un transistor HBT.

Mesures en entrée du HBT « du haut » par la sonde HIP

En plaçant la sonde HIP en entrée du transistor HBT du « haut » du circuit MMIC, nous pouvons visualiser la forme d'onde de la tension dans un plan intrinsèque. Une telle forme correspond complètement aux simulations faites lors de la conception du circuit [8]. On visualise en réalité la tension provenant par la diode d'entrée du modèle de HBT. C'est une première étape vers une double accréditation : la validation des mesures vis à vis des simulations et la validation du modèle thermique de HBT vis à vis des mesures.
Ces formes d'ondes de tension, toujours paramétrées en fonction de différents niveaux de puissance disponible, sont représentés figure 35.

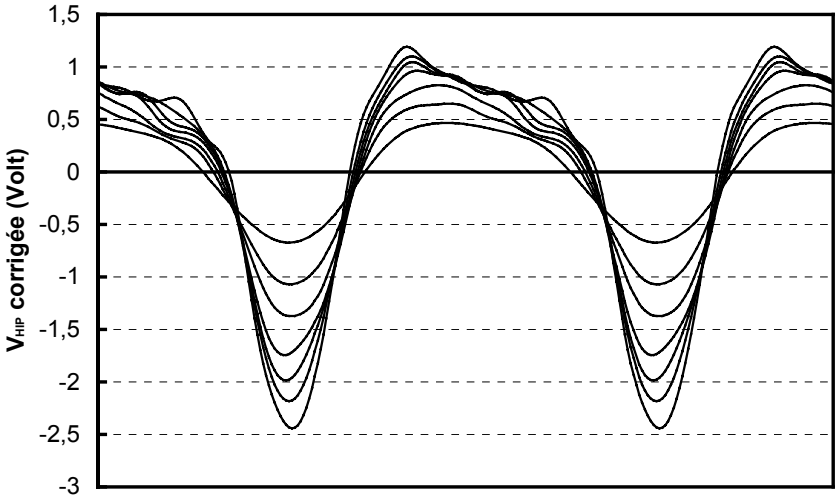


Fig. 35 - Forme temporelle de la tension mesurée (et corrigée) en entrée du transistor HBT du haut.

Mesures en sortie du HBT du haut par la sonde HIP

Finalement, la sonde HIP est placée en sortie du même transistor (celui du « haut »). Les formes d'ondes obtenues sont représentées figure 36. On s'aperçoit qu'à forte puissance disponible (autour de 14 dBm) la forme d'onde est quasi-carrée. Cela accrédite la conception optimisée en classe F. Là encore, la conjonction entre les résultats de mesures et de simulations est flagrante.

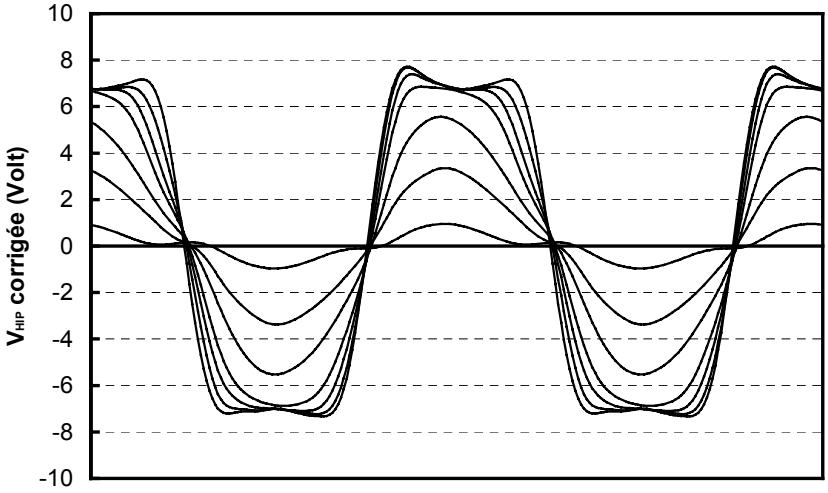


Fig. 36 - Tension de sortie du transistor HBT du haut mesurée par sonde HIP.

Conclusion

On a pu mettre en évidence via mesures dans des plans de référence intrinsèques la qualité d'une conception d'un amplificateur « à priori » optimisé en classe F.

Notons que les mesures ne fournissent pas le DC : ce qui donne des formes d'ondes centrées autour de zéro. Néanmoins, la mesure du DC est facilement réalisable de façon calibrée avec

les sondes HIP : il suffit pour cela de remplacer les « DC Blocks » présents entre les sondes HIP et le 'downconverter box' par des tés de polarisation. La valeur du continu peut être relevée en connectant sur le port BF un simple voltmètre. L'étalonnage peut se faire en injectant une tension DC connue dans une ligne d'étalonnage qui est mesurée par la sonde HIP afin d'en déduire les pertes du système.

En se qui concerne l'amplificateur, des mesures ont également été faites sur le transistor du bas. En ce qui concerne la puce MMIC dont les résultats ont été présentés, la symétrie est parfaite.
En revanche, certaines autres puces présentent une asymétrie et des formes moins « carrées » que celles visualisées ici. Cela est sûrement due à l'ancienneté du circuit (1996) qui est, par ailleurs, basé sur une technologie HBT vieillissant relativement mal.

Vérification de phénomènes dynamiques complexes : la stabilité non-linéaire de circuits MMIC

La vérification du comportement de dispositifs non-linéaires est l'application principale du système de mesure développé. Cette application peut être étendue à des domaines encore plus expérimentaux tel que l'analyse de stabilité non-linéaire. C'est le sujet de ce paragraphe.
Dans un premier temps, des généralités seront abordées concernant l'étude de stabilité non-linéaire dans les amplificateurs de puissance.
Ensuite, un amplificateur à 2 étages conçu autour d'une technologie HBT sera proposé comme dispositif sous test. Cet amplificateur a été l'objet de nombreuses études sur la stabilité non-linéaire paramétrique par plusieurs laboratoires car il présente une division de fréquence à fo/2 et une topologie relativement simple. Il constitue pour nous un véhicule de test idéal afin d'établir le lien entre l'instrumentation et l'analyse de stabilité.
Après une brève présentation de l'amplificateur et du banc de mesure, nous nous placerons dans un point de fonctionnement particulier afin de pouvoir déceler ce qui pourrait être un mode de stabilité impair.

Généralités sur les études de stabilité non-linéaire

Ce paragraphe s'attache à présenter d'une façon d'une façon simple (et donc non exhaustive) les problèmes et les intérêts liés à l'analyse de stabilité des circuits non-linéaire. Il a pour but de donner quelques explications élémentaires, parfois de façon simpliste compte tenu de la complexité du sujet, afin de mieux interpréter des phénomènes obtenus expérimentalement sur un amplificateur. Ces résultats expérimentaux seront présentés plus tard dans le rapport (cf. « Analyse comportementale de l'amplificateur »).

Définition d'un système non-linéaire dynamique [9]

Typiquement, un circuit non-linéaire micro-onde peut être assimilé, dans tous processus de modélisation et de simulation à un système dynamique non-linéaire.
Ce système dynamique non-linéaire est généralement représenté avec :

- une entrée notée u(t) : cela peut être un scalaire ou un vecteur colonne suivant le nombre d'entrées considérées ;
- une sortie notée y(t) : même remarque que pour l'entrée ;
- un vecteur X(t) constitué des variables d'états du système.

La représentation couramment adoptée est représentée figure 37.

Les variables d'états sont les variables caractérisant intrinsèquement le comportement du système. Elles doivent être définies telles que la sortie du système y(t) puisse être déterminée de façon unique quelque soit l'instant t en fonction de cette information (le vecteur X(t)) et de l'entrée u(t).

Par exemple, nous pouvons avoir un système non-linéaire tel que :

$$y(t) = f_{NL}\left(u(t), \frac{\partial u(t)}{\partial t}, \frac{\partial^2 u(t)}{\partial t^2}, \dots, \frac{\partial y(t)}{\partial t}, \frac{\partial^2 y(t)}{\partial t^2}, \dots, t \right)$$

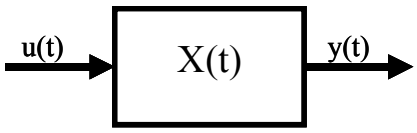
On peut alors exprimer la réponse du système y(t) tel que :

$$y(t) = f_{NL}[u(t), X(t), t] \text{ avec } X(t) = \begin{pmatrix} \frac{\partial u(t)}{\partial t} \\ \vdots \\ \frac{\partial y(t)}{\partial t} \\ \vdots \end{pmatrix} \text{ le vecteur des variables d'état.}$$

Pour les systèmes dynamiques, il est nécessaire de connaître les variations au cours du temps du vecteur d'état, d'où l'équation :

$$\frac{\partial X(t)}{\partial t} = g_{NL}(X(t), u(t), t)$$

Pour un système invariant dans le temps, les fonctions non-linéaires f et g sont indépendantes du temps. Ainsi on peut écrire : $y(t) = f_{NL}(u(t), X(t))$ et $\frac{\partial X(t)}{\partial t} = g_{NL}(u(t), X(t))$.



$$\begin{cases} y(t) = f_{NL}(u(t), X(t)) \\ \dot{X}(t) = g_{NL}(u(t), X(t)) \end{cases}$$

Fig. 37 - Processus de modélisation d'un système non-linéaire

Nous remarquerons qu'un système statique est décrit par une équation d'état telle que : $\dot{X}(t) = 0$ quelque soit t ; et qu'un système est linéaire (principe de superposition possible) si et seulement si les fonctions f et g sont toutes deux linéaires.

Espace des phases

L'espace des phases d'un système est la « signature » du comportement de ce système. Généralement cet espace est la représentation graphique de l'ensemble formé par toutes les valeurs possibles pouvant être prise par les variables d'états : X(t). Les variables d'états forment la base vectorielle de la représentation graphique. Pour des systèmes simples, dotés

de 2 ou 3 variables d'états, l'espace des phases peut être visualisée dans un plan ou un espace en 3 dimensions. Si le nombre de variables d'états est plus important, on travaille alors avec des projections de l'espace des phases telles que les sections de Poincaré [10].
Le traditionnel cycle de charge d'un transistor est en quelque sorte une représentation dans un espace des phases dont les 2 variables d'états sont le courant et la tension en sortie du transistor.
Des exemples d'espaces de phases sont représenté figure 38. Chacune de ces 3 illustrations correspond à un système doté de 3 variables d'états (représentation en 3 dimensions). Ces 3 illustrations sont extraits de [11].

Méthodes de simulation et solutions comportementales

Les méthodes de simulations permettant une analyse précise des phénomènes dynamiques non-linéaires sont des méthodes d'intégration temporelle (type SPICE) car elles restent compatible avec le domaine d'étude : le domaine temporel.
On peut ainsi constater qu'un système peut avoir plusieurs solutions comportementales qui peuvent coexister. Parmi ces solutions, une seule est physiquement possible, visualisable expérimentalement, vraie, *stable*[3]. Les autres solutions, si elles coexistent sont dites *instables*.

La figure 38 présente différente solutions qui peuvent exister (voire même coexister) dans un amplificateur de puissance RF (ou dans son modèle dynamique) dont le signal d'excitation est une simple porteuse CW.

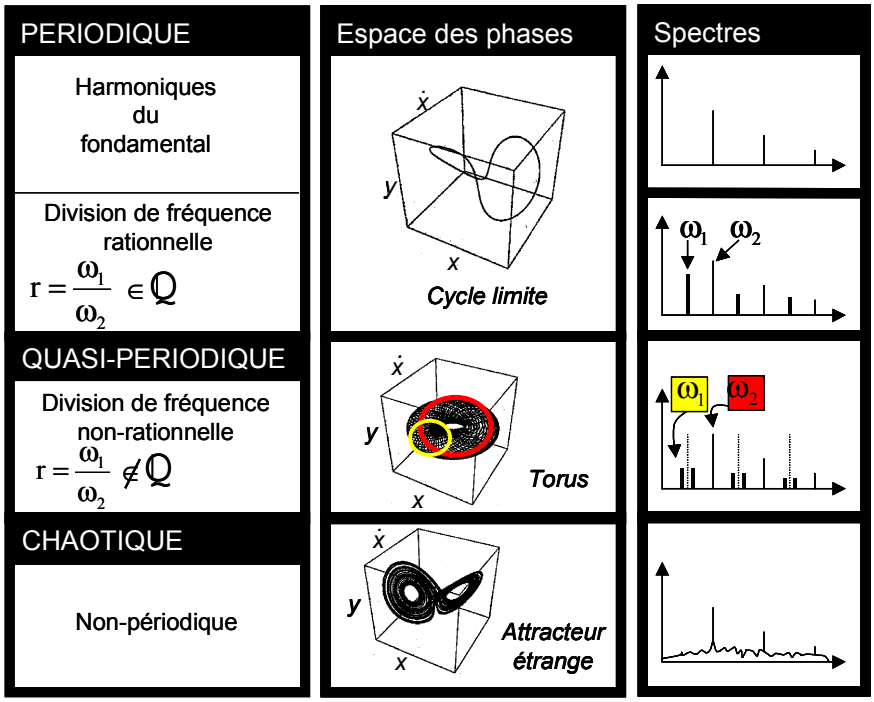


Fig. 38 - Résumé symbolique de différents comportements pouvant être pris par un système dynamique non-linéaire.

Parmis les solutions possibles, on trouve des réponses :

[3] Lors de mesures expérimentales d'un amplificateur présentant une oscillation, on aura tendance à dire qu'il est « instable ». C'est un abus de langage. En effet, la solution du système dynamique qui est mesurée expérimentalement est forcément stable. Ce qui est instable, c'est la solution souhaitée par le concepteur, celle dont les oscillations sont absentes.

- périodiques ;
- quasi-périodiques ;
- chaotiques.

Une solution en régime périodique est constitué du fondamental et de ses harmonique. C'est la solution triviale (donnée par l'équilibrage harmonique) et désirée par tous les concepteurs d'amplificateurs. Faut-il encore s'assurer qu'il s'agit là d'une solution stable.
Une autre solution périodique est la division de fréquence rationnelle. La période du signal de sortie augmente d'un même rapport que la division de fréquence. La représentation du vecteur d'état X(t) dans l'espace des phases reste alors une courbe bouclée : un cycle limite.

Une solution quasi-périodique peut apparaître dans les amplificateurs. Il s'agit de l'apparition d'une fréquence autonome complètement indépendante du signal d'excitation. A l'analyseur de spectre, on visualise alors 2 raies parasites : la fréquence autonome ainsi que son produit vis à vis de la porteuse CW d'excitation. La représentation dans l'espace des phases lorsque le nombre de variable d'état est de 3 conduit à un torus dont chacune des rotations dans le parcours du lieu (la largeur du « donut » et son épaisseur) est liée aux fréquence de la porteuse CW et de la fréquence autonome.

Enfin, on a représenté un comportement chaotique figure 38. Ce comportement pourrait faire penser à un phénomène stochastique : il n'est rien d'autre que du bruit et amène à un spectre continu. Le tracé des réponses temporelles dans le diagramme des phases permet de se rendre compte que ce phénomène « à priori » aléatoire est en fait déterministe : il est chaotique.

La variation d'un paramètre de fonctionnement du système (la puissance ou la fréquence dans le cas des amplificateur de puissance RF) peut amener au changement de la solution (et donc du comportement) du système. Il s'agit alors d'un système présentant des oscillations paramétriques. On parle alors de *bifurcation* dans le comportement du dispositif. Ces bifurcations peuvent conduire à de nombreux états comportementaux intermédiaires à ceux qui viennent d'être présentés.
On trouve dans la littérature des tracés de lieux comportementaux : on repère différentes zones sur un plan dont les axes sont la puissance et la fréquence du signal CW d'excitation. Chaque zone correspond à un type de comportement du dispositif.
Par ailleurs, il se peut que des phénomènes d'hystérésis interviennent dans ces bifurcation. Ainsi on passera d'un comportement A à un comportement B pour une puissance P1 lorsque le niveau de puissance augmente. Le passage du comportement B au A en diminuant la puissance (chemin inverse) se fera pour un niveau P2 différent de P1.

La plupart des analyses de tels phénomènes se fait par des simulations en temporelles. Or la conception et la simulation de dispositifs RF ne permet pas l'emploi de tels simulateurs compte tenu des coûts prohibitifs des temps de calculs et de la profondeur mémoire nécessaire. Les méthodes de simulations et de conceptions sont principalement fréquentielles (équilibrage harmonique) ou tempo-fréquentielle (transitoire d'enveloppe). De telles méthodes peuvent présenter des difficultés à converger et vont donner quasi-systématiquement la solution triviale (fondamental et harmonique) même si cette dernière n'est pas stable.
Des méthodes récentes [12] [13] permettent de déceler, dans un environnement de simulation par équilibrage harmonique la potentialité de solutions différentes de la solution triviale, mais ne permettent pas de prédire avec certitude l'unique solution stable.

Parmi ces travaux récents, des analyses de stabilité sur des amplificateurs présentant des division de fréquence rationnelles en mode pair ou impair ont été faite. L'identification et la compréhension de ces phénomènes de division de fréquence permettent de corriger de façon optimale la topologie d'un dispositif afin de rendre stable l'unique solution qui intéresse le concepteur d'amplificateurs : un spectre ne présentant que le fondamental associé à ses harmoniques en ne dégradant que faiblement les performances au fondamental.

Nouvelles solutions pour la conception d'amplificateurs de puissance [14]

Parmi les mécanismes d'oscillation pouvant exister dans les transistors de puissance lors d'une division de fréquence, on dénote entre autre :

- la stabilité de mode pair ;
- la stabilité de mode impair.

De plus, ces oscillations peuvent être paramétriques, les rendant d'autant plus difficile à prédire[4]. Des exemple vont illustrer ces types de stabilités qui ont lieu sur des topologies symétriques. Dans nos exemples, on supposera que la division de fréquence est produite par l'entrée du transistor (effet de la capacité non-linéaire).

Le mode pair

On est en présence d'un mode pair lorsque les oscillations présentes sur les transistors symétriques sont en phase. En conséquence, cette oscillation est également présente à la jonction des 2 ondes en extrémité du diviseur / combineur de puissance.
Une technique habituelle de stabilisation consiste à placer en entrée de chaque transistor un réseau RC. L'oscillation sous harmonique peut être annulée, mais les performances globales de l'amplificateur au fondamental peuvent également être diminuées. Cette technique reviens à augmenter la marge de stabilité sur le facteur K (si on raisonne en stabilité petit signal).

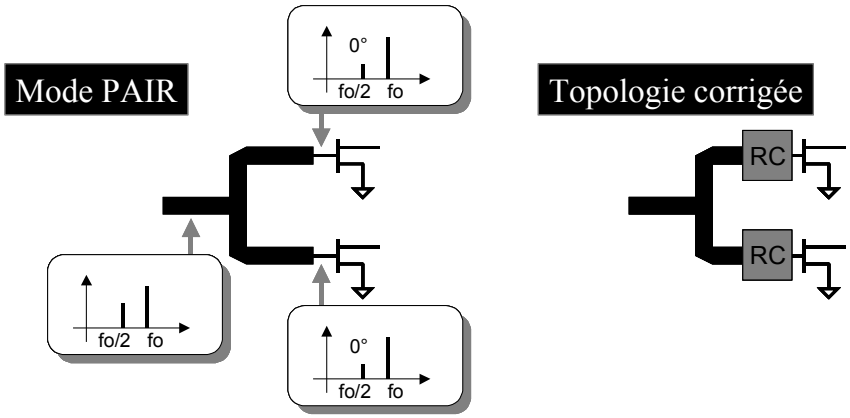


Fig. 39 - Correction d'une stabilité en mode pair

Le mode impair

[4] Les méthodes d'analyse de stabilité linéaire (facteur K) ne sont plus valables pour des oscillations paramétriques. Ces oscillations apparaissent et évoluent généralement en fonction de la puissance : les dispositifs non-linéaires ne sont plus en « petit signal ».

Le mode impair se caractérise par des oscillations qui sont en opposition de phase dans le plan des transistors. Si la structure est parfaitement symétrique, cette oscillation peut ne pas être présente en entrée du diviseur de puissance. Cela crée un plan de masse virtuel à la fréquence d'oscillation en ce point. Ce type d'oscillation est particulièrement dangereuse et destructrice lors de mesures car elle peut avoir lieu dans les plans des transistors sans qu'elle puisse être détectée dans les plans d'accès traditionnels du circuit.
Une méthode efficace pour stabiliser un mode impair consiste à placer des résistances entre les entrées et/ou sortie des transistors symétriques. Les performances du circuit au fondamental ne sont pas affectées par ce changement de configuration.

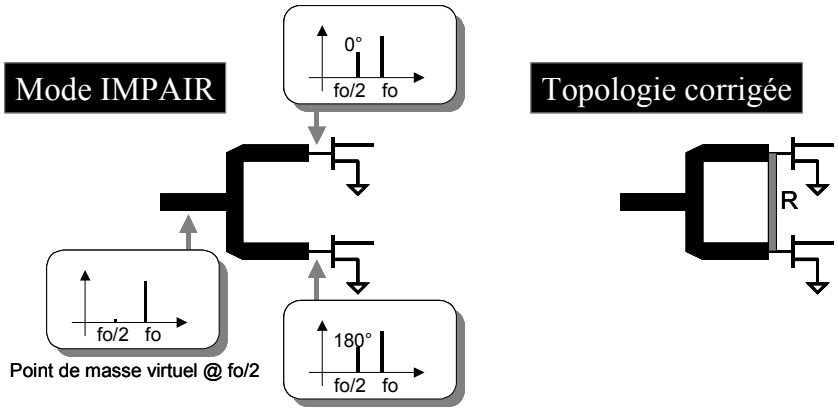


Fig. 40 - Correction d'une stabilité en mode impair

L'instrumentation pour l'analyse de stabilité

Comme nous l'avons vu dans les chapitres et rapports précédents, le LSNA est un système d'instrumentation qui fonctionne exactement comme l'équilibrage harmonique à 1 porteuse. Il tient compte du fondamental et de ses harmoniques. On peut également l'utiliser en présence de signaux modulés en bande étroite (8 MHz). Cet aspect n'a pas été présenté dans ces travaux, mais cela lui confère un aspect « transitoire d'enveloppe ».
Un lien extrêmement puissant existe entre l'instrumentation architecturée autour d'un LSNA et les méthodes de simulations couramment utilisée dans la conception des circuits RF.
L'association du LSNA aux sondes hautes impédance permet une liberté spatiale concernant les points de mesure et aboutit à un système complètement comparable à un environnement de simulation / modélisation.

Le fait de pouvoir placer nos sondes HIP en des endroits arbitraires à l'intérieur de circuits hybrides ou MMICs permet de mesurer des formes d'ondes aux accès mêmes des transistors. Il devient alors possible de détecter et même d'identifier (via la mesures des phases) les types d'oscillations présentes dans les plans des transistors. C'est un outils qui trouve alors un intérêt particulier sur des stabilités de mode impair qui, lorsqu'elles existent, peuvent être fortement atténuées voire inexistantes dans les plans de mesures traditionnels.

C'est pour mettre en évidence ce profond intérêt, qui a des conséquences sur la conception même des circuits, que nous avons faits des mesures sur des amplificateurs présentant des oscillations. L'un d'eux, fonctionnant en bande X, a été largement étudié à travers plusieurs publications [15] [16] et présente « a priori » une division de fréquence par 2 en mode impair. C'est cet amplificateur que nous allons présenter et mesurer.

Présentation de l'amplificateur bande X

L'amplificateur sur lequel ont été effectuées les mesures est le circuit « Frio ». Il fonctionne en bande X et est doté de deux étages. Il a été conçu autour d'une technologie HBT au cours de la thèse de Marie-Anne Perez [17]. Le layout de ce circuit est représenté figure 41.
Cet amplificateur présente une stabilité paramétrique en division de fréquence par 2 de mode impair qui a largement été étudiée sur un aspect simulation.
Ce phénomène a pu être vérifier expérimentalement à l'analyseur de spectre sur une puce ne fonctionnant plus à ce jour.
Une seconde puce de ce circuit mise en boîtier a été mesurée mais elle présente un comportement bien plus complexe qu'une simple division de fréquence paramétrique par 2. Dans la plupart des cas, on visualise une division de fréquence par 4. Une procédure d'optimisation du point de polarisation et de la fréquence d'excitation CW a due être faite manuellement afin de trouver un point de fonctionnement présentant quasiment une division de fréquence par 2. Ceci explique que le point de polarisation et la fréquence de travail ne sont pas les même que ce qui a été déjà publié au sujet de cet amplificateur.
De plus l'amplificateur présentait des oscillations BF dès la polarisation. En conséquence, des capacités ont dues être ajouté au plus prés du boîtier (soudures) au niveau des plots de polarisation.

<u>Fréquence de fonctionnement</u>
f_0 = 9.630 GHz

<u>Polarisation 1er étage :</u>
Vb1 ≅ 1.5 volts (valeur non exacte : contrôle suivant Ic1)
Vc1 = 6.5 volts
Ic1 = 56 mA

<u>Polarisation 2ème étage :</u>
Vb2 ≅ 1.5 volts (valeur non exacte : contrôle suivant Ic2)
Vc2 = 8 volts
Ic2 = 224 mA

L'usage de cette puce qui manifestement ne correspond pas au comportement prédit laisse supposer que nous ne pourrons pas comparer nos résultats de mesures à des simulations et que nous n'aurons pas forcément une stabilité de mode impair.

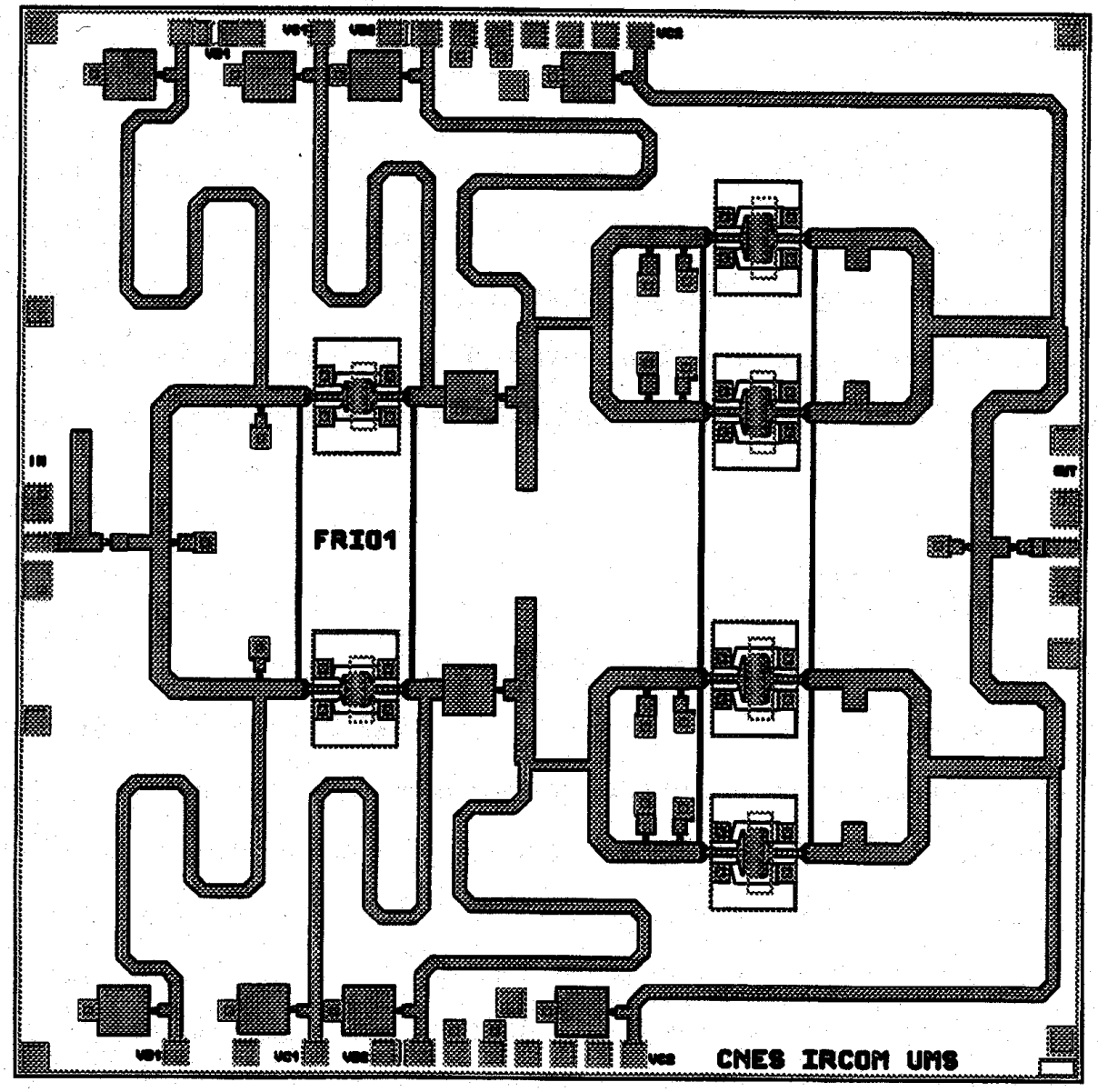


Fig. 41 - Layout de l'amplificateur « Frio »

Banc de mesure

Le banc de mesure utilisé pour la caractérisation du circuit « Frio » est schématisé figure 42. Nous verrons lors des mesures de mode de stabilité impair que l'usage de 2 sondes HIP s'avère indispensable ici.
Une source RF extérieure est utilisée (toujours pour résoudre des problèmes de puissance disponible). De plus, on utilise un wattmètre pour mesurer les puissances moyennes en entrée et sortie du dispositif sous test ainsi qu'un analyseur de spectre. Cette structure du banc nous permet de faire varier le comportement de l'amplificateur « Frio » afin de trouver une configuration de fonctionnement optimale pour nos investigations. Alors le LSNA pourra mesurer les formes d'ondes sans nécessiter aucune déconnexion.

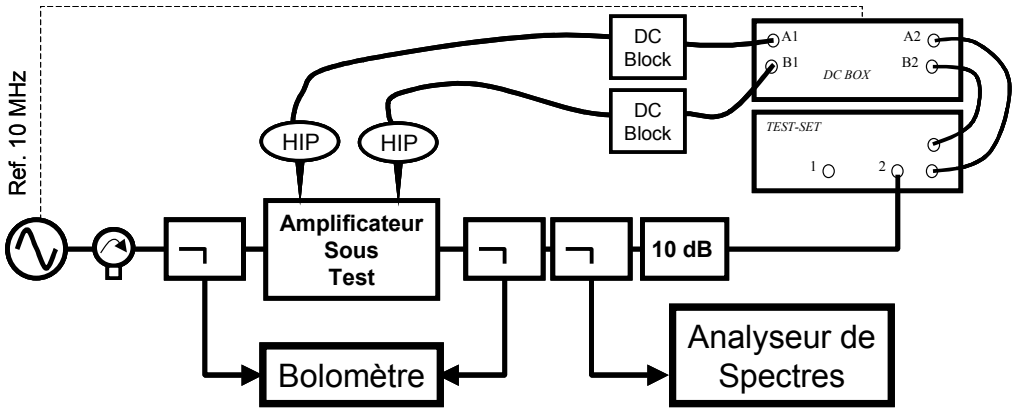


Fig. 42 - Banc de mesure pour l'amplificateur « Frio »

Analyse comportementale de l'amplificateur

On se propose de balayer la puissance disponible du signal CW d'excitation de l'amplificateur « Frio ». Les mesures sont faites, lors de ce balayage, simultanément par l'analyseur de spectre et le wattmètre.

Dans un premier temps on s'intéressera aux relevés de courants de polarisation moyens et de puissances moyennes aux accès de l'amplificateur. Ces courbes sont représentées en fonction de la puissance d'entrée (figure 43) et de la puissance de sortie (figure 44).
Un phénomène intéressant apparaît entre les points correspondants aux puissances d'entrées Pe = 8.80 dBm et Pe = 9.76 dBm : il y a une discontinuité des caractéristiques. De façon locale, la puissance de sortie a diminué alors que la puissance d'entrée a augmenté. Nous pourrons voir dans la représentation des spectres correspondant à chacun des points de puissances mesurés qu'il s'agit d'un changement de comportement de l'amplificateur.

La figure 45 est la mesure faite par l'analyseur de spectre pour le premier point de puissance (Pe = 4.86 dBm). La fenêtre fréquentielle imposée permet de voir le fondamental et ses sous harmoniques mais pas ses harmoniques. Comme une division par 4 pourra être observée (à partir de Pe = 9.76 dBm : soit la discontinuité observée précédemment), la grille de mesure du LSNA est choisie de façon telle que le fondamental du LSNA corresponde à la fréquence divisée par 4 présent au niveau de l'amplificateur. Cette grille est également représentée

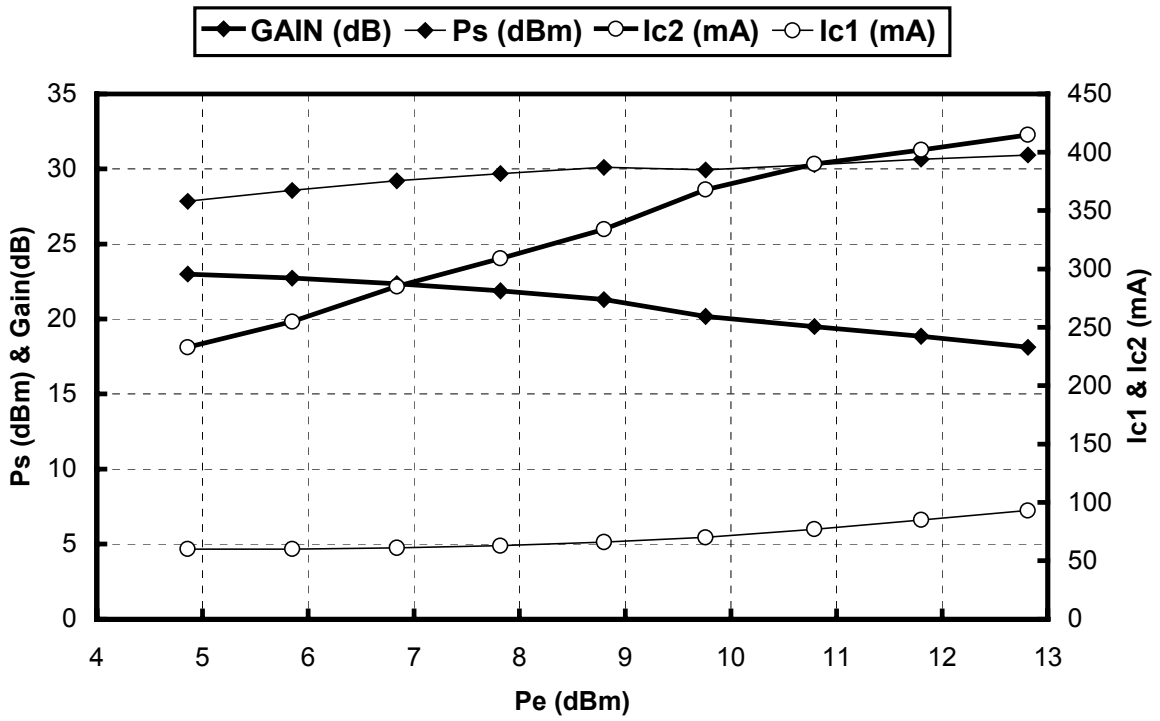


Fig. 43 - Caractéristiques statiques de l'amplificateur « Frio » en fonction de la puissance disponible injectée en entrée.

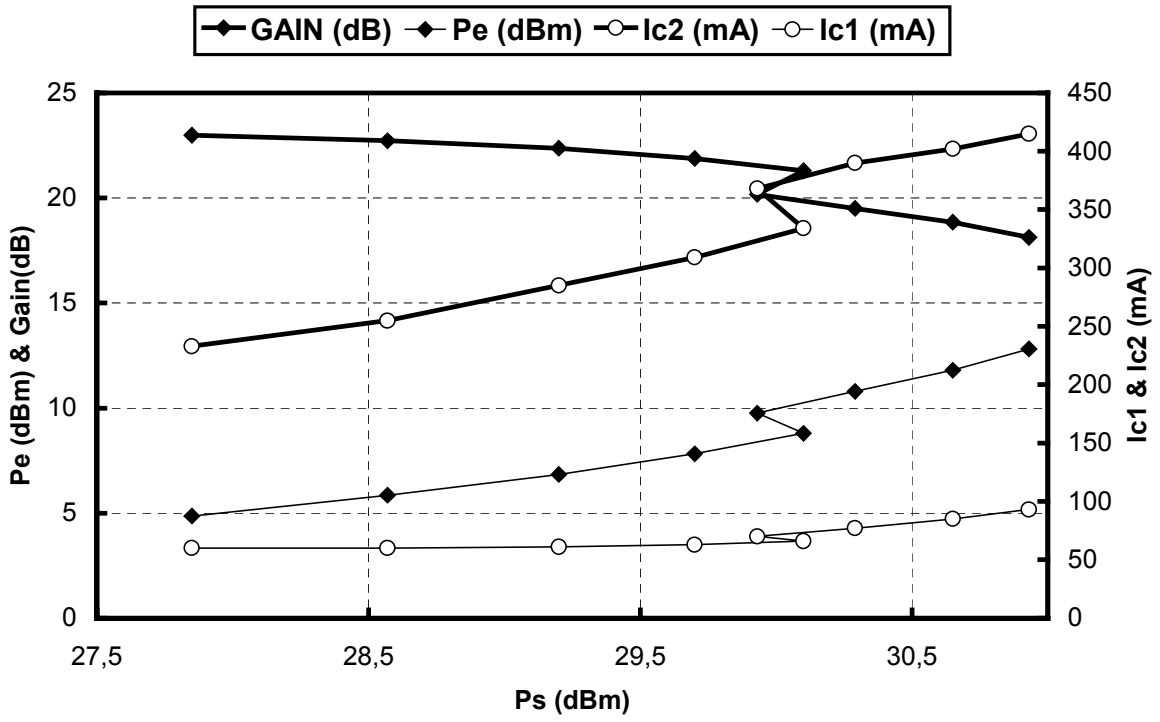


Fig. 44 - Caractéristiques statiques de l'amplificateur « Frio » en fonction de la puissance moyenne présente en sortie du circuit. Cette représentation permet d'apprécier une discontinuité dont l'origine est l'apparition d'une division de fréquence par 4.

figure 45. En conséquences, l'étalonnage du LSNA se fait à 2.4075 GHz avec au moins 4 harmoniques pour pouvoir mesurer le véritable fondamental. 8 harmoniques ont été mesurées dans nos expérimentations : la limite étant la fréquence d'analyse maximale de 20 GHz sur le système NMRC.

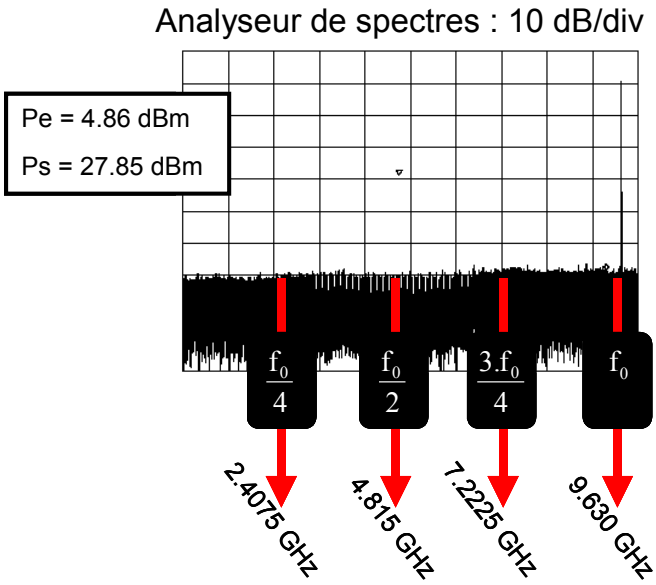


Fig. 45 - Spectre visualisé pour le premier point de puissance. Aucune oscillation parasite n'est alors visualisée. La grille d'analyse du LSNA est présentée en superposition du spectre. Elle peut permettre une analyse lors d'une division par 2 ou par 4.

Les spectres visualisés pour chacun des autres points de puissances sont représentés figure 46. On peut alors interpréter les différents comportements paramétriques que prend le circuit en faisant varier le niveau du signal CW d'excitation. Alors que le circuit fonctionne normalement (fondamental et harmoniques) pour Pe = 4.86 dBm, une oscillation apparaît à partir de Pe = 5.85 dBm. Voici les différents comportements en fonction du niveau de puissance disponible :

Pe = 5.85 dBm

Apparition d'une oscillation. Il s'agit d'une fréquence autonome du circuit et de son image (produit par le signal d'excitation CW). En augmentant légèrement la puissance du fondamental, les 2 raies d'oscillations (la fréquence autonome et son image) se resserre pour tendre « a priori » vers la division par 2. Cette bifurcation passe cependant par un état intermédiaire : le précurseur.

Pe = 6.84 dBm à 8.80 dBm

Visualisation du précurseur. Un précurseur correspond à une remontée spectrale du bruit de façon locale (autour de l'oscillation attendue). Ce phénomène est caractéristique d'une bifurcation. La bifurcation observée ici se fait entre un état d'oscillation libre (fréquence autonome) et un état d'oscillation par division de fréquence.

Pe = 9.76 dBm à 10.79 dBm

Passage du précurseur à la division par 4. Cela correspond à la discontinuité dans les caractéristiques statiques présentées figures 43 et 44. Typiquement la division par 2 apparaît avant une éventuelle division par 4 mais ce n'est pas le cas ici, même en réglant très précisément le niveau de puissance injectée.

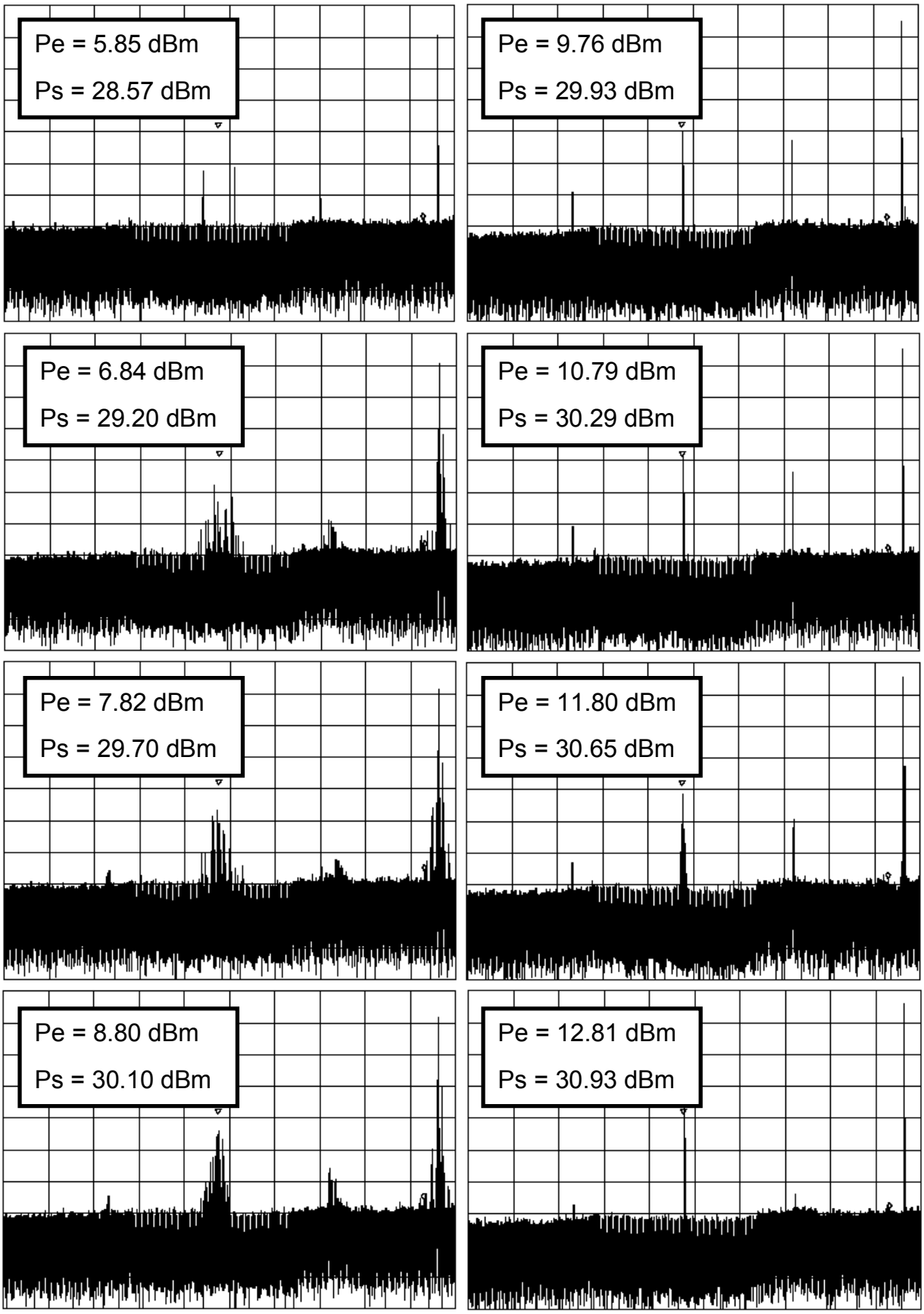


Fig. 46 - Spectres obtenus à l'analyseur pour chaque points de puissance considéré. Lorsque la puissance disponible d'entrée augmente, on visualise entre autre l'apparition de fréquence autonome, un précurseur, une division par 4, une division par 4 bruitée et une quasi division par 2.

Pe = 11.80 dBm

A la division par 4 s'ajoute des oscillations BF présentant un grand nombre de raies autour de chaque fréquence sous harmonique du signal d'excitation CW. Il est probable que le phénomène soit une intermittence [10]. Les intermittences sont caractéristique d'une bifurcation vers un état chaotique.

Pe = 12.81 dBm

Les oscillations BF sont atténuées. En augmentant le niveau de 11.80 dBm à 12.81 dBm, les sous harmoniques ¼ et ¾ diminuent de façon paramétrique. Le système se retrouve alors quasiment dans une configuration de division de fréquence par 2. C'est dans cette configuration que nous effectuerons des mesures avec le LSNA et les sondes HIPs.

Mesures effectuées avec le système LSNA+HIP

Des mesures sont faites lorsque l'amplificateur « Frio » est attaqué par un niveau CW d'environ 12.8 dBm avec une fréquence de 9.630 GHz.

Le LSNA et les sondes HIP sont calibrés pour un fondamental à 2.4075 GHz. Ainsi le signal d'excitation CW (f_0) peut être mesuré sur l'harmonique n°4, la division par 2 ($\frac{f_0}{2}$) sur l'harmonique n°2 et l'oscillation à $\frac{3.f_0}{4}$ sur l'harmonique n°3.

Les mesures au moyen des sondes HIP sont faites en sortie des transistors du dernier étages. La figure 47 permet de localiser les mesures qui seront présentées ultérieurement.

Compte tenu de la topologie « symétrique », une oscillation en mode impair est attendue. Ceci explique le choix des localisations des mesures.

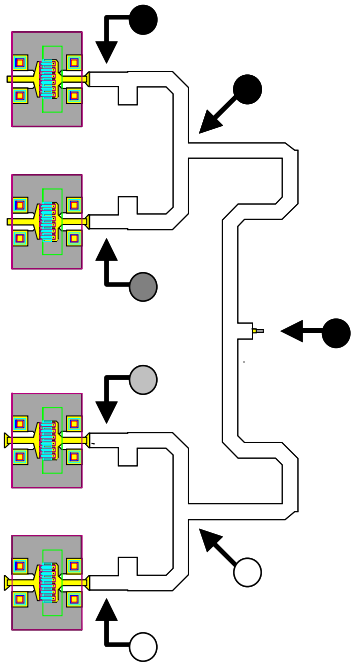


Fig. 47 - Indices de couleurs pour les mesures effectuées en sortie de transistors

Mesures en amplitude

Les représentations des spectres du module de la tension (en dBV) mesurés par sondes HIP sont données figures 48 (sortie des transistors), 49 (sortie du premier combineur) et 50 (sortie du second combineur). Les couleurs correspondent aux points de mesures illustrés figure 47 et les représentations spectrales sont fonction des harmoniques de mesure du LSNA (signal d'excitation en harmonique 4).

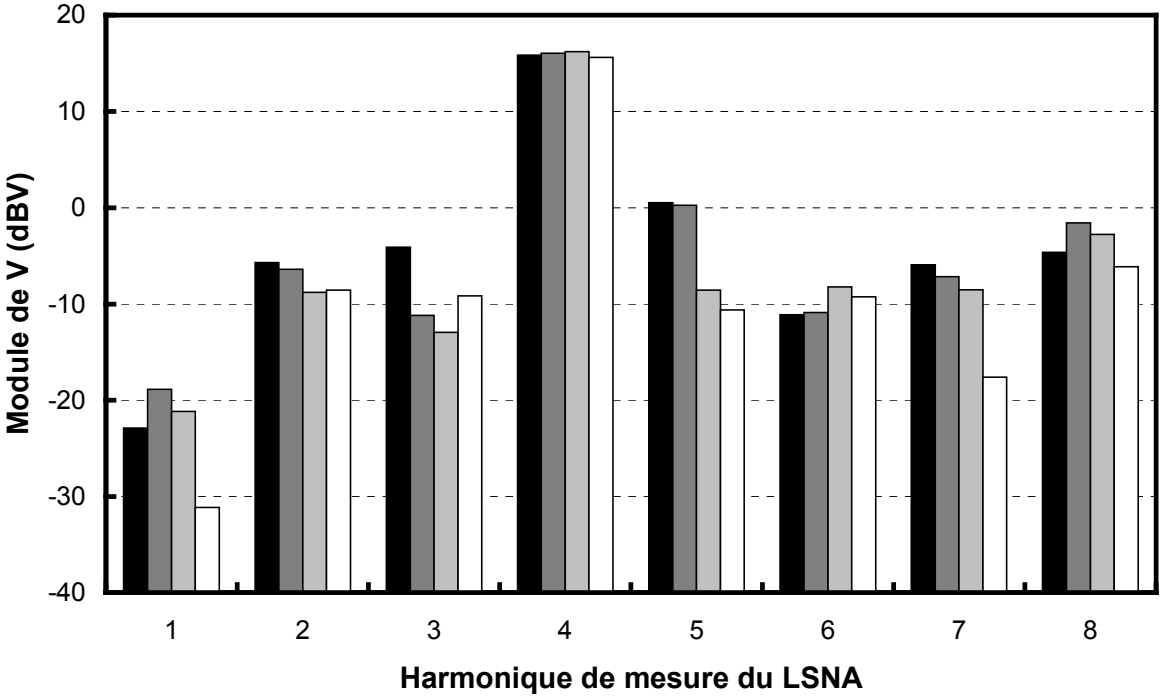


Fig. 48 - Module des tensions mesurées directement en sortie des 4 transistors répertoriés du haut (foncé) vers le bas (clair)

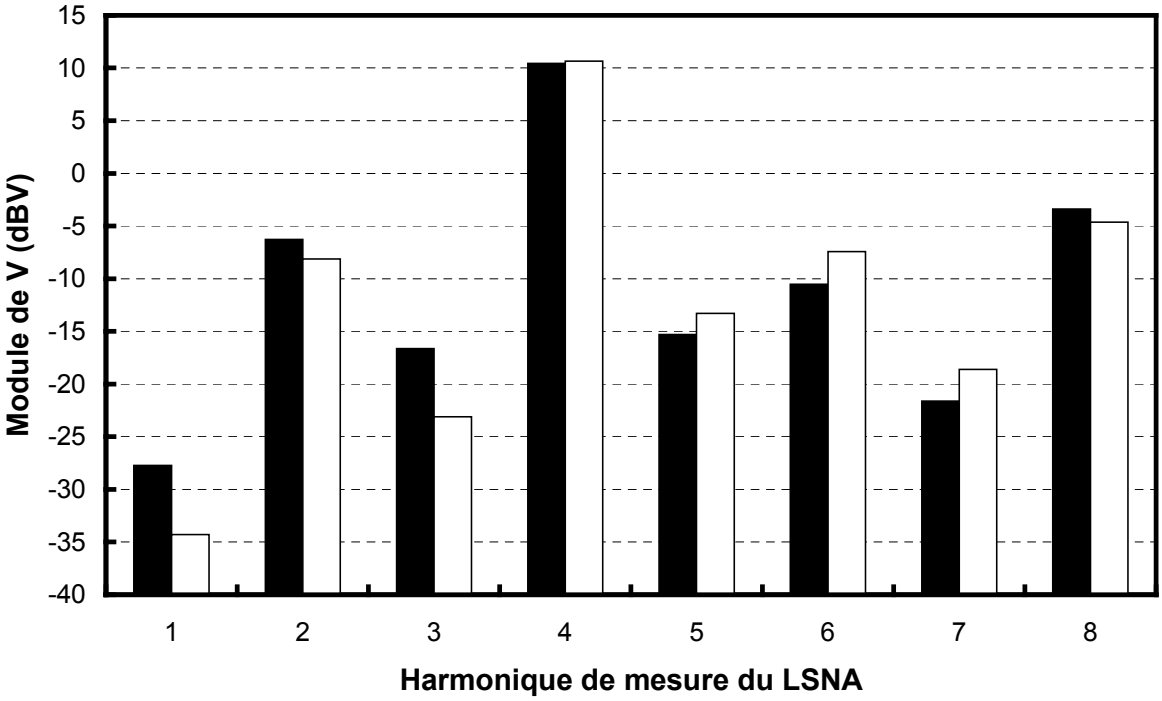


Fig. 49 - Module des tensions mesurées directement en sortie du premier combineur de puissance.

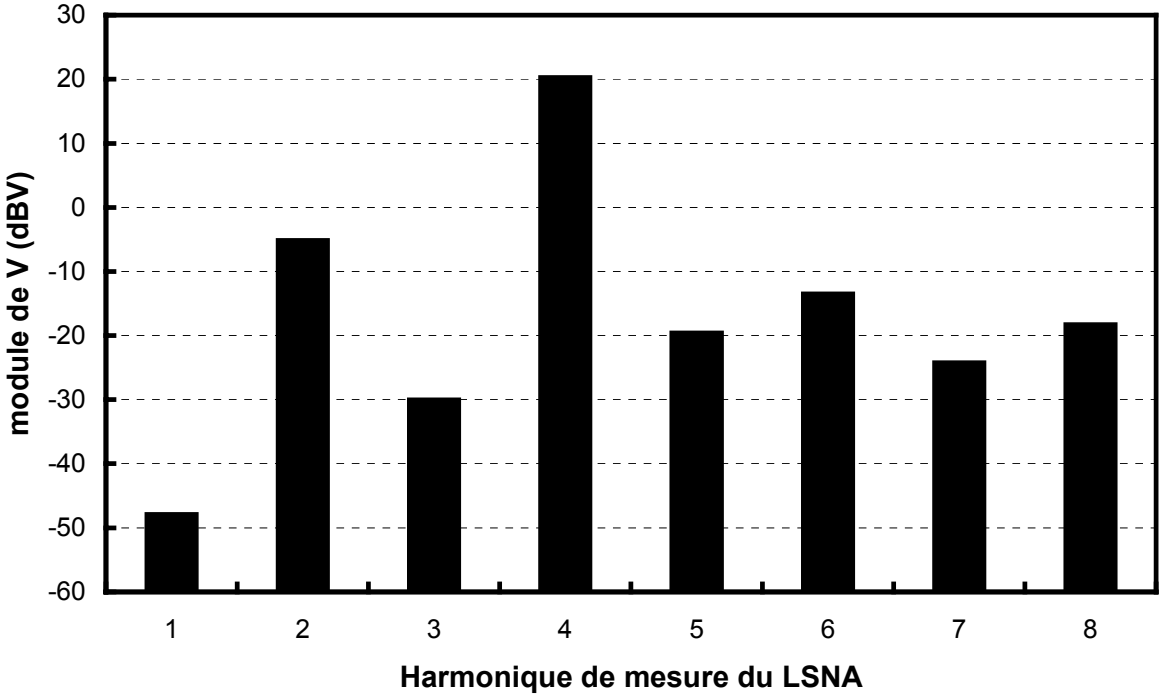


Fig. 50 - Module des tensions mesurées en sortie du second combineur de puissance

Les harmoniques 1 et 3 présentent un aspect fort remarquable : alors qu'elle sont quasiment absentes du signal de sortie (aussi bien sur la mesure à l'analyseur de spectre que sur les mesures temporelles), elles sont bien plus présentes au cœur du circuit.

L'écart entre les harmoniques 3 et 4 (entre l'oscillation à $\frac{3.f_0}{4}$ et la raie à f_0) est de l'ordre de 50 dBc en sortie du circuit ; de 30 dBc en sortie du premier combineur et de 25 dBc en sortie des transistors.

Ces écarts montrent l'importance des oscillations dans le circuits vis à vis de ce qui est traditionnellement mesurable en sortie de circuit.

Mesures en phase

Les phases également ont été mesurées. Typiquement l'origine des phases est imposée sur la mesure de tension faite sur le port 2 du LSNA au fondamental. Dans notre cas, la raie au fondamental du LSNA est l'oscillation à $\frac{f_0}{4}$: elle est par conséquent noyée dans le bruit de mesure. La synchronisation en phase doit donc se faire sur la fréquence présentant une amplitude maximale sur le port 2 : la 4ème harmonique (f_0). Mais en imposant la phase de f_0 sur le port 2 comme phase de référence, la valeur de la phase mesurée à $\frac{f_0}{2}$ sur le port 2 ou sur les HIPest modulo π ; la phase mesurée à $\frac{f_0}{4}$ ou $\frac{3.f_0}{4}$ sur les HIP (les mesures issue du port 2 sont trop bruitées, donc inexploitables) sont modulo $\frac{\pi}{2}$.

Les investigations avec une seule sonde HIP sont possibles pour mesurer la phase à $\frac{f_0}{2}$. Pour cela, il faut également mesurer la phase à $\frac{f_0}{2}$ sur le port 2 à chaque mesure de sonde HIP pour pouvoir comparer les mesures entres elles. On peut fixer par exemple une phase de référence telle que sur le port 2, la phase du signal à f_0 soit nulle et celle à $\frac{f_0}{2}$ soit positive (entre 0 et π).

Les mesures de phases sur l'oscillation en $\frac{f_0}{2}$ n'ont rien données. Les valeurs de phases mesurées n'ont pas de relation notable : il ne s'agit ni d'un mode de stabilité pair, ni d'un mode de stabilité impair.

Les investigations pour mesurer les phases à $\frac{3f_0}{4}$ sont plus délicates. En effet, pour une référence de phase imposée en f_0 sur le port 2, 4 valeurs sont possibles. Ces valeurs ne sont pas triables comme l'oscillation à $\frac{f_0}{2}$ car on ne mesure rien à $\frac{3f_0}{4}$ sur le port 2.
Il nous faut pourtant une référence de phase viable qui puisse être la même sur toutes les mesures faites avec les sondes HIP. La meilleur solution consiste à utiliser 2 sondes HIP. Une des sondes est placée en sortie directe d'un des transistor. Elle permet une bonne mesure de l'oscillation à $\frac{3f_0}{4}$: on fixe ainsi l'origine des phase sur cette mesure, via HIP, de la raie à $\frac{3f_0}{4}$. Les autres points de mesures sont investigués avec une seconde sonde HIP, et toute les phases mesurées à $\frac{3f_0}{4}$ sont référencées par rapport à la première phase mesurée à $\frac{3f_0}{4}$ avec la première sonde.

Deux oppositions de phases suivant la symétrie du circuit ont pu être ainsi être mises à jour à $\frac{3f_0}{4}$. On pourrais alors ainsi dire qu'une portion du circuit est en mode de stabilité impair.

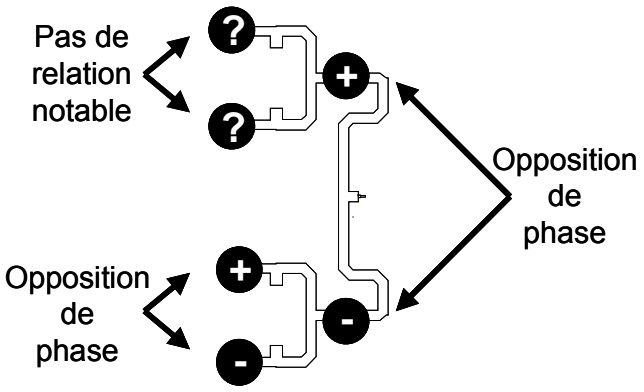


Fig. 51 - Comparatif des phases mesurées à 3.fo/4

Les oscillations à $\frac{3f_0}{4}$ ne sont pas pour autant annulées sur les plan de masses virtuels compte tenu des amplitudes des tensions qui ne sont pas symétriques. Néanmoins, c'est bien ce principe d'oscillation en mode impair qui est à l'origine du fait que l'oscillation à $\frac{3f_0}{4}$ se détecte à peine en comparaison à celle présente dans le plan du transistor.

Les amplitudes des mesures d'oscillations à $\frac{f_0}{2}$ via sondes HIP ne permettent pas une mesure correcte des phases.

Conclusion

Nous avons pu mettre en évidente, de façon expérimentale, une stabilité de mode impair dans un circuit MMIC. Nous regrettons cependant que la puce mesurée ne présente pas les mêmes caractéristiques que celles obtenues dans des investigations précédentes comme une pure division par 2.
Monter d'autres puces du même circuit en boîtier aurait été appréciable mais n'a pas été possible compte tenu du temps impartit pour l'utilisation du LSNA.

De nombreux autres circuits ont été mesurés. Parmi ceux-ci, un amplificateur Alcatel présentant une division de fréquence paramétrique par 2. Un mode de stabilité pair ou impair a été recherché sans succès car la topologie du combineur rend les interprétations des mesures assez difficile.

CONCLUSION

Ce rapport a présenté l'état d'avancement des travaux liés aux mesures calibrées de formes d'ondes temporelles par sondes hautes impédance. Il a permis de présenter les investigations majeures qui ont été faite dans 3 domaines :

- La vérification et la cohérence de la procédure d'étalonnage inspirée de celle proposée par le NIST : la fonction de transfert $\widetilde{K}(f)$;
- La création d'un nouvel étalonnage plus rigoureux ;
- L'intérêt de ce nouveau système de mesure dans des applications de fiabilité et de stabilité.

En ce qui concerne la cohérence des étalonnages des poursuites peuvent avoir lieu, notamment pour la détermination expérimentale des constantes de propagations et des impédances caractéristiques de lignes quelconques. Ces axes constituent une évolution naturelle de la nouvelle technique qui a été mise en place.

Toutes les nouvelles techniques abordées depuis maintenant 3 rapports, on été intégralement implantées dans l'environnement 'Mathematica' du LSNA à travers un nouveau module : le module « HIP` ». L'utilisation de ce module a été très clairement illustré et est extrêmement simple pour un utilisateur habitué à la structure conventionnelle du LSNA.

Par ailleurs une modification du code de l'environnement 'Mathematica' du LSNA permet dorénavant d'utiliser un pied de test pour toutes les mesures classique réalisables sous pointes avec le LSNA.

Ajoutons qu'un outils d'analyse de maillages de tensions a été développé sous Scilab.

Tous ces ajouts sont fournis en annexe ce rapports à travers leurs listing. Ce rapport propose donc une méthode de mesure et d'analyse nouvelle « clef en main » et pourra s'avérer d'une grande utilité pour quelqu'un souhaitant prolonger cette activité : chose fortement probable compte tenu des potentialités énormes que peut offrir une système de mesure calibré de formes temporelles sur les nœuds internes de circuits MMICs.

BIBLIOGRAPHIE

[1] P. Kabos, H.C. Reader, U. Arz, and D.F. Williams, "Calibrated Waveform Measurement with High Impedance Probes", IEEE Trans. on MTT, Février 2003.

[2] U. Arz et Al., "Wideband frequency domain characterization of high impedance probes", 58th ARFTG Conference Digest, San Diego, CA, November 2001

[3] J. Stenarson, K. Yhland, C. Wingqvist, "An In-Circuit Noncontacting measurement Method for S-Parameters and Power in Planar Circuits", .IEEE Trans. On MTT, vol.49, no 12, Dec 2001, pp.2567-2572.

[4] J. Verspecht, "Calibration of a measurement system for high frequency nonlinear devices", PhD dissertation, Vije Universiteit Brussel, September 1995.

[5] D.C. DeGroot, J.A. Jargon, R.B. Marks, "Multiline Revealed", 60th ARFTG Conference Digest, pp. 131-156, Washington DC, December 5-6, 2002

[6] T. Reveyrand, "Mesures de formes d'ondes temporelles multi-harmoniques avec le LSNA. Mise en place expérimentale du calibrage des sondes HIP", Rapport d'avancement post-doctoral n°2, Février 2003

[7] T. Reveyrand, "Etude préliminaire pour la mise en place d'un système de mesure de formes d'ondes temporelles multi-harmonique au moyen de sondes HIP", Rapport d'avancement post-doctoral n°1, Novembre 2002

[8] A. Mallet, "Optimisation des conditions de fonctionnement du Transistor Bipolaire à Hétérojonction pour l'amplification de puissance à haut rendement : applications aux communications microondes entre mobiles", Thèse de Doctorat de l'Université de Limoges, n°53-96, 28 novembre 1996

[9] C. G. Cassandras, "Discrete Event Systems : Modeling and Performace Analysis", Ed. Aksen Associates, Inc. 1993

[10] A. Suárez, R. Quéré, "Stability Analysis of Nonlinear Microwave Circuits", Artech House Publishers, ISBN 1-58053-303-5, 2003

[11] Y. Yamamoto, "Detection of Chaos and Fractals from Experimental Time Series", in : *Modern Techniques in Neuroscience Research*, (Eds. U. Windhorst & H. Johansson), Springer-Verlag, Chapter 24, pp. 669-687, 1998

[12] J. Portilla et Al., "Analysis of Nonlinear RF and Microwave Circuits Using Harmonic Balance and System Identification Methods", Journal of RF and Microwave, pp. 448-459, december 2002

[13] A. Anakabe et Al., “Analysis and Elimination of Parametric Oscillations in Monolithic Power Amplifiers”, IEEE MTT-S Digest, pp.2181-2184, 2002

[14] P. de Hek, “Design and results of state-of-the-art microwave monolithic integrated high-power amplifiers”, Procedings of SAFE 2000, pp. 58-65, 2000

[15] A. Anakabe et Al., “Analysis of Odd-Mode Parametric Oscillations in HBT Multi-Stage Power Amplifiers”, à paraître, European Microwave Week, GAAS 2003 Conference, Septembre 2003

[16] A. Anakabe et Al., “Efficient nonlinear stability analysis of microwave circuits usig commercially available tools”, EuMc, pp. 1017-1020, 2002

[17] M. A. Pérez, “Modèle électrothermique distribué de transistor bipolaire à hétérojonction : application à la conception non linéaire d’amplificateurs de puissance optimisés en température”, Thèse de Doctorat de l’Université de Limoges, n°61-98, 9 décembre 1998

ANNEXES

Modifications de la procédure d'étalonnage LRRM sous l'environnement 'Mathematica'

L'intégralité du code source du module 'LRRM.m' en 'Mathematica' est présenté ici. Les zones ajoutées et / ou modifiées ont été surlignées. L'utilisateur lambda d'un système LSNA pourra alors étalonner l'appareil avec un kit de calibrage semblable à celui présent avec le pied de test Wiltron.
Une fois le code du module **'LRRM.m'**modifié on sélectionne le type d'étalonnage grâce à une nouvelle variable binaire dans le fichier ressource **'LRRM.mrs'** : LRRM`UTF. Si elle est vraie, alors le module considérera un étalonnage sur l'UTF Wiltron ; sinon c'est un étalonnage « on wafer » classique.

FICHIER RESSOURCE « LRRM.mrs »

```
(* Module LRRM` Version : LRRM`Private`VersionString *)
(* LineDelay: Specifies the delay of the line element (seconds). *)
LRRM`LineDelay=1.0*10^-12;
(* MatchResistance: This value defines the value of the match resistance. If one wants to
measure the value automatically one can specify the module which is used for applying and
measuring DC bias. (e.g. LRRM`MatchResistance="DCSource4142`") *)
(* LRRM`MatchResistance="DCSource4142`"; *)
LRRM`MatchResistance=50.0;
(* A rajouter : Match 2 suivant le fwrd mode *)
(* AuxiliaryCalPort: This parameter specifies at which RF port the auxiliary calibration takes
place.  When the default auxiliary cal connector is used the value should be 1. For specific
set-ups with an external test-set  (cf. Alex Cognata) it is possible to perform the auxiliary
cal at port 2. *)
LRRM`AuxiliaryCalPort=1;
LRRM`MeasureDCResistance=False;
LRRM`UTF=True ;
```

FICHIER MODULE « LRRM.m »

```
(* Revision Control Information *)
(* $Id: LRRM.m,v 2.3.1.9 2002-06-09 17:22:29+02 marcvb Exp $ *)
(* $LRRM$ *)
(* $Header: /NMDGArchive/NNMS/MAT/source/LRRM.m 2.9 2002-06-09 17:15:43+02 marcvb Exp $ *)
(* Modifications : MARS 2003 Tibo / CNES *)
(*                + 2 fonctions GO["OS"] et GO["SO"] *)
(*                + CalElements conditionel -depend de UTF ds les resources *)
(*                + CalculateCalCoefficients => Un test en plus pour achever le travail des
                   nouvelles fonctions GO *)

(*
                ____________________________________

Agilent Confidential Copyright (C) 2000 Agilent Technologies, Inc.

DISCLAIMER OF WARRANTY

 The following software: $Source: /NMDGArchive/NNMS/MAT/source/LRRM.m $
 is experimental and is provided under a research agreement, "AS-IS" by Agilent
```

```
 Technologies, Inc. ("Agilent"). Agilent shall have no obligation to maintain or
 support this software. AGILENT MAKES NO EXPRESS OR IMPLIED
 WARRANTY OF ANY KIND REGARDING THIS SOFTWARE.
 AGILENT SHALL NOT BE LIABLE FOR ANY DIRECT, INDIRECT,
 SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES,
 WHETHER BASED ON CONTRACT, TORT OR ANY OTHER LEGAL
 THEORY, IN CONNECTION WITH OR ARISING OUT OF THE
 FURNISHING, PERFORMANCE OR USE OF THIS SOFTWARE..


                  RESTRICTED RIGHTS LEGEND

 Use, duplication or disclosure by the U.S. Government is subject to restrictions as
 set forth in subparagraph (c)(1)(ii) of the Rights in Technical Data and Computer
 Software clause in DFARS 252.227-7013.

                     Agilent Technologies, Inc.
                       395 Page Mill Road
                          Palo Alto, CA
                        94303-0870, USA

 Rights for non-DOD U.S. Government Departments and Agencies are as set forth
 in FAR 52.227-19(c)(1,2).


                  ______________________________

*)

(* You need to specify the names of all resource variables and a string describing the
   resource *)

LRRM`Private`resourcelist={
        {"MeasureDCResistance","MeasureDCResistance specifies whether the parasitic DC
resistance should be measured or not during the measurement of the SHORT calibration
elements."},

     {"LineDelay","Specifies the delay of the line element (seconds)."},

     {"MatchResistance","This value defines the value of the match resistance. If one wants to
      measure the value automatically one can specify the module which is used for applying
      and measuring DC bias. (e.g. LRRM`MatchResistance=\"DCSource4142`\")"},

     {"AuxiliaryCalPort","This parameter specifies at which RF port the auxiliary calibration
      takes place. When the default auxiliary cal connector is used the value should be 1. For
      specific set-ups with an external test-set (cf. Alex Cognata) it is possible to perform
      the auxiliary cal at port 2."},

      {"UTF","This parameter specifies if a Wiltron Universal Test Fixture is used (TRUE) or
not. It enables calibrations with SHORT on a port when an OPEN is on the other one."}

};

(* If there are undefined resources, an error is thrown to return from loading the package *)
(* Please specify the module name by replacing Template` *)
If[Module`Private`UndefinedResourcesQ["LRRM`"][[1]],Throw[False]];

(* You need to specify the name of the package *)
BeginPackage["LRRM`"];

(* You need to declare all the functions which need to become public *)
SRDWarm::usage="SRDWarm returns whether the SRDhas been warmed up or not. This
information is used during the phase calibration.";

LoadInductance::invalid="Invalid specification for load inductance. Allowed are
a list with two numbers (to automatically estimate the load inductance starting
with the specified interval), or a single number. Calibration calculations are
continued with a load inductance assumed to be zero.";

LRRMCalCoefficients::usage="LRRMCalCoefficients returns, for each user setup and
frequency, the corresponding LRRM calibration coefficients.";

GenCalCoefficients::usage="GenCalCoefficients returns, for each user setup and
frequency, the calibration coefficients corresponding to eq. (5.5-8) in
\"Calibration of a Measurement System for High Frequency Nonlinear Devices\",
PhD. thesis of Jan Verspecht, VUB (University of Brussels), September 1995.
(downloadable PDF at http://users.skynet/jan.verspecht, through NMDG-Bibliography hyperlink)";

Tmatrices::usage="Tmatrices returns, for each user setup and
frequency, the T-matrix coefficients corresponding to eq. (5.5-2) in
\"Calibration of a Measurement System for High Frequency Nonlinear Devices\",
PhD. thesis of Jan Verspecht, VUB (University of Brussels), September 1995.
(downloadable PDF at http://users.skynet/jan.verspecht, through NMDG-Bibliography hyperlink)"

KCoefficients::usage="KCoefficients returns,for each user setup and
frequency, the K calibration coefficients corresponding to eq. (5.5-1) in
```

```
\"Calibration of a Measurement System for High Frequency Nonlinear Devices\",
PhD. thesis of Jan Verspecht, VUB (University of Brussels), September 1995.
(downloadable PDF at http://users.skynet/jan.verspecht, through NMDG-Bibliography hyperlink)"

ReflectMatchEstimates::usage="ReflectMatchEstimates returns, for each user setup and
frequency, the corresponding estimates for the reflection coefficient of the
SHORT and OPEN cal element, and the estimated impedance of both matches.";

MatchResistances::usage="CalData[MatchResistances] returns the measured DC resistances of the
Match cal

element.";
ShortResistances::usage="CalData[ShortResistances] returns the measured DC resistances of the
Short cal element.";

(* Allows automatic MatchResistance calibration at DC *)
(* If one wants to automatically determine the value of the match resistance
the name of the bias module to be used (as a string) is specified in stead
of the value *)

Begin["`Private`"];

(* The VersionString is automatically update by the RCS revision control *)
(* Do not touch this *)
VersionString = "$Id: LRRM.m,v 2.3.1.9 2002-06-09 17:22:29+02 marcvb Exp $";

(* Making sure that the utilities to work with modules are known *)
(* Do not touch when you do need Module utilities like OpenModule in this module *)
error=Needs["Module`"];
If[(error === $Failed),Throw[False]];

(* Implementation of the module *)

error=OpenModule["CalibrationMgr`","RawAcquisition`","CalModule`","RefGen`","General`",
                      "Source`","Powermeter`","PowerLeveling`","CalKit`",
                      "Spectrum`"];

(* When an error occured, an exception must be thrown to leave the routine *)
(* When multiple packages are opened, error contains a list of error messages. *)
(* Each error message is a list of 3 elements *)
If[Depth[error]>2,
  If[MemberQ[Transpose[error][[1]],False],Throw[False]],
  If[error[[1]]==False,Throw[False]]
];

(* Allows automatic MatchResistance calibration at DC *)
(* If one wants to automatically determine the value of the match resistance
the name of the bias module to be used (as a string) is specified in stead
of the value *)
If[StringQ[MatchResistance],error=OpenModule[MatchResistance]];

(* When an error occured, an exception must be thrown to leave the routine *)
(* When multiple packages are opened, error contains a list of error messages. *)
(* Each error message is a list of 3 elements *)
If[Depth[error]>2,
  If[MemberQ[Transpose[error][[1]],False],Throw[False]],
  If[error[[1]]==False,Throw[False]]
];


(* Measuring Tools *)

SRDWarm=False;

If[UTF==True,CalElements=
{"1| SHORT-OPEN |2","1| OPEN-SHORT |2","MATCH","LINE","Laux","Oaux","Saux","POWER","HPR"},
CalElements={"SHORT","OPEN","MATCH","LINE","Laux","Oaux","Saux","POWER","HPR"}];

Go[any1_,any2__]:=Map[Go[#]&,Join[{any1,any2}]];

(* MODIFICATION marcvb 17 December 2001 *)
(* Checking if the requested Aux cal port is supported by the hardware *)
   If[Xor[(AuxiliaryCalPort ==

1),(Length[StringPosition[GetSupportedModes,"FORWARD_AUX1"]]>0)],OKMessageString["AuxiliaryCal
Port is

set to port 1 while not supported by hardware"];Throw[False]];
   If[Xor[(AuxiliaryCalPort ==

2),(Length[StringPosition[GetSupportedModes,"FORWARD_AUX2"]]>0)],OKMessageString["AuxiliaryCal
Port is

set to port 2 while not supported by hardware"];Throw[False]];
   If[(AuxiliaryCalPort > 2),OKMessageString["LRRM`AuxiliaryCalPort is set larger than 2 while
not
```

```
supported by hardware"];Throw[False]];

Go["LINE"]:=
     Module[{},
              If[StringQ[MatchResistance],
                 origforce = ForceMode;
                 BiasOn;
                 SetForceMode[Voltage,Current];
                 ForceBias[0.5,0.1];
                 Pause[1];
                 DCline = MeasureBias;
                 LineResistance = (DCline[[2,1]]-DCline[[1,1]])/DCline[[2,2]];
                 PrintMessage["Measured line resistance: "<>ToString[LineResistance]<>" Ohm."];
                 BiasOff;
                 SetForceMode @@ origforce,
                 LineResistance = 0
                ];
              SetSourcePower[-20];
              SetSourceOutput[On];
              CalData["LINE"]=Map[thrucal[#]&,CalibrationMgr`Private`calSetups];
              SetSourceOutput[Off];
            ];

Go["1| SHORT-OPEN |2"]:=
Module[{forwS1,forwO2},
     SetSourcePower[-20];
              SetSourceOutput[On];
              ForwardPort1Mode;
              forwS1=Map[relcal[#,1]&,CalibrationMgr`Private`calSetups];

              ForwardPort2Mode;
              forwO2=Map[relcal[#,2]&,CalibrationMgr`Private`calSetups];
     CalData["1| SHORT-OPEN |2"]={forwS1,forwO2};
              SetSourceOutput[Off];
];

Go["1| OPEN-SHORT |2"]:=
Module[{forwO1,forwS2},
     SetSourcePower[-20];
              SetSourceOutput[On];
              ForwardPort1Mode;
              forwO1=Map[relcal[#,1]&,CalibrationMgr`Private`calSetups];

              ForwardPort2Mode;
              forwS2=Map[relcal[#,2]&,CalibrationMgr`Private`calSetups];
     CalData["1| OPEN-SHORT |2"]={forwO1,forwS2};
              SetSourceOutput[Off];
];


Go["OPEN"]:=
          Module[{forw1,forw2},
              SetSourcePower[-20];
              SetSourceOutput[On];
              (* MODIFICATION marcvb 16 Dec 2001 *)
              (* Replacing the CalBox by TestSet: Forward1Mode -> ForwardPort1Mode *)
              ForwardPort1Mode;
              forw1=Map[relcal[#,1]&,CalibrationMgr`Private`calSetups];
              (* MODIFICATION marcvb 16 Dec 2001 *)
              (* Replacing the CalBox by TestSet: Forward2Mode -> ForwardPort2Mode *)
              ForwardPort2Mode;
              forw2=Map[relcal[#,2]&,CalibrationMgr`Private`calSetups];
              CalData["OPEN"]=Map[Transpose,{forw1,forw2}//Transpose];
              SetSourceOutput[Off];
            ];

Go["SHORT"]:=
          Module[{forw1,forw2},
              If[StringQ[MatchResistance],
                 origforce = ForceMode;
                 BiasOn;
                 SetForceMode[Current,Current];
                 ForceBias[0.1,0.1];
                 Pause[0.5];
                 DCshort = MeasureBias;
                 CalData[ShortResistances] =

{DCshort[[1,1]]/DCshort[[1,2]],DCshort[[2,1]]/DCshort[[2,2]]};
                 CalData["DCResistances"]=CalData[ShortResistances];
                 PrintMessage["Measured short resistance:
"<>ToString[CalData[ShortResistances]]<>"

Ohm"];
                 BiasOff;
                 SetForceMode @@ origforce,
                 CalData[ShortResistances] = {0., 0.};
                 CalData["DCResistances"]=CalData[ShortResistances]
```

```
                ];
               SetSourcePower[-20];
               SetSourceOutput[On];
               (* MODIFICATION marcvb 16 Dec 2001 *)
               (* Replacing the CalBox by TestSet: Forward1Mode -> ForwardPort1Mode *)
               ForwardPort1Mode;
               forw1=Map[relcal[#,1]&,CalibrationMgr`Private`calSetups];
               (* MODIFICATION marcvb 16 Dec 2001 *)
               (* Replacing the CalBox by TestSet: Forward2Mode -> ForwardPort2Mode *)
               ForwardPort2Mode;
               forw2=Map[relcal[#,2]&,CalibrationMgr`Private`calSetups];
               CalData["SHORT"]=Map[Transpose,{forw1,forw2}//Transpose];
               SetSourceOutput[Off];
             ];

Go["MATCH"]:=
           Module[{forw1,forw2},
               If[StringQ[MatchResistance],
                  origforce = ForceMode;
                  BiasOn;
                  SetForceMode[Voltage,Voltage];
                  ForceBias[0.5,0.5];
                  Pause[1];
                  DCmatch = MeasureBias;
                  CalData[MatchResistances] =

{DCmatch[[1,1]]/DCmatch[[1,2]],DCmatch[[2,1]]/DCmatch[[2,2]]};
                  PrintMessage["Measured match resistance:
"<>ToString[CalData[MatchResistances]]<>"

Ohm"];
                  BiasOff;
                  SetForceMode @@ origforce,
                  CalData[MatchResistances] = {50., 50.}
                ];
               SetSourcePower[-20];
               SetSourceOutput[On];
               (* MODIFICATION marcvb 16 Dec 2001 *)
               (* Replacing the CalBox by TestSet: Forward1Mode -> ForwardPort1Mode *)
               ForwardPort1Mode;
               forw1=Map[relcal[#,1]&,CalibrationMgr`Private`calSetups];
               (* MODIFICATION marcvb 16 Dec 2001 *)
               (* Replacing the CalBox by TestSet: Forward2Mode -> ForwardPort2Mode *)
               ForwardPort2Mode;
               forw2=Map[relcal[#,2]&,CalibrationMgr`Private`calSetups];
               CalData["MATCH"]=Map[Transpose,{forw1,forw2}//Transpose];
               SetSourceOutput[Off];
             ];


Go[any_]:=
     Module[{},
               SetSourcePower[-20];
               SetSourceOutput[On];
               (* MODIFICATION marcvb 16 December 2001 *)
               (* Replacing the CalBox with the TestSet: Forward2Mode -> ForwardAux1Mode *)
               (*                                        Forward1Mode -> ForwardAux2Mode *)
               Switch[AuxiliaryCalPort,1,ForwardAux1Mode,2,ForwardAux2Mode];
               CalData[any]=Map[relcal[#,AuxiliaryCalPort]&,CalibrationMgr`Private`calSetups];
               SetSourceOutput[Off];
             ] /; MemberQ[{"Laux","Oaux","Saux"},any];

Go["POWER"]:=
     Module[{},
               SetSourcePower[-20];
               SetSourceOutput[On];
               (* MODIFICATION marcvb 16 December 2001 *)
               (* Replacing the CalBox with the TestSet: Forward2Mode -> ForwardAux1Mode *)
               (*                                        Forward1Mode -> ForwardAux2Mode *)
               Switch[AuxiliaryCalPort,1,ForwardAux1Mode,2,ForwardAux2Mode];
               CalData["POWER"]=Map[powercal[#]&,CalibrationMgr`Private`calSetups];
               SetSourceOutput[Off];
           ];

(* MODIFICATION marcvb 16 December 2001 *)
(* Replace CalBox by TestSet and seperate RefGen *)
Go["HPR"]:=
     Module[{},
               SetSourcePower[-20];
               SetSourceOutput[On];
               If[PowerSensorAcknowledge=!=False,PowerSensorAcknowledge=True];
               (*
               If[RefGenAuxPowerConnector && Not[$Notebooks] && PowerSensorAcknowledge,
                  InputString[" Make sure powersensor is connected to the Power Leveling of the
the

Harmonic Phase Reference! (press return)"];
```

```
                refgencont
               ];
             If[RefGenAuxPowerConnector && Not[$Notebooks] && Not[PowerSensorAcknowledge],
                refgencont
               ];
             If[RefGenAuxPowerConnector && $Notebooks && PowerSensorAcknowledge,
                powerOKbutton
               ];
             If[RefGenAuxPowerConnector && $Notebooks && Not[PowerSensorAcknowledge],
                refgencont
               ];
             If[Not[RefGenAuxPowerConnector],refgencont];
             *)
             If[Not[RefGenAuxPowerConnector],refgencont,
                If[Not[PowerSensorAcknowledge],refgencont,OKMessageString[" Make sure
powersensor is

connected to the Power Leveling of the Harmonic Phase Reference!"];refgencont]
             ];
            ];

refgencont:=(
             (* MODIFICATION marcvb 16 December 2001 *)
             (* Replacing the CalBox with the TestSet: RefGenMode -> RefGenAux1Mode *)
             (*                                        RefGenMode -> RefGenAux2Mode *)
             Switch[AuxiliaryCalPort,1,RefGenAux1Mode,2,RefGenAux2Mode];
             If[RefGenAuxPowerConnector && $Notebooks,NotebookClose[nbpow]];
             CalData["HPR"]=Map[phasecal[#]&,CalibrationMgr`Private`calSetups];
             PrintMessage["HPR finished."];
             SetSourceOutput[Off];
             SRDWarm=False;
            );

powerOKbutton:=Module[{continue},
          nbpow = NotebookCreate[
           System`FrontEndVersion -> "X 3.0",
           System`ScreenRectangle -> {{0, 1280}, {0, 1024}},
           System`Editable -> False,
           System`WindowToolbars -> {},
           System`PageWidth -> 325,
           System`WindowSize -> {250, 40},
           System`WindowMargins -> {{Automatic, 375}, {Automatic, 75}},
           System`WindowFrame -> "Palette",
           System`WindowElements -> {},
           System`WindowFrameElements -> "CloseBox",
           System`WindowClickSelect -> False,
           System`WindowTitle -> "Power Sensor Connected?",
           System`ScrollingOptions -> {"PagewiseScrolling" -> False},
           System`ShowCellBracket -> False,
           System`CellMargins -> {{0, 0}, {System`Inherited, 0}},
           System`Active -> True,
           System`CellOpen -> True,
           System`ShowCellLabel -> False,
           System`ShowCellTags -> False,
           System`ImageMargins -> {{0, System`Inherited}, {System`Inherited, 0}},
           System`Magnification -> 1,
           System`ButtonBoxOptions -> {ButtonStyle -> "Evaluate"}];

           NotebookApply[nbpow,Cell[
                                  BoxData[
                                          ButtonBox["OK",
                    ButtonEvaluator->Automatic,
                    ButtonFunction:>(
                      (
                       refgencont;
                      )
                                 &)
                                                  ]
                                          ],"Input"
                                     ]
                             ];
                     ];

relcal[setup_,port_]:=
   Module[{calinstrumentstate,f0},
      f0=setup[[1]];
      calinstrumentstate=
          {setup[[2]],setup[[3]],setup[[4]],setup[[5,1]],setup[[5,2]]};
      SetInstrumentState[calinstrumentstate];
      Table[
          SetSourceFreq[i*f0];
          SetSourcePower[RelCalPowerFunction[i*f0][[port]]];
          Transpose[
                      (* Occasionally an ADC overflow may occur, due to ADC offset, coupler
ripple,...

which
```

```
                         make it impossible to accurately calculate maximum synthesizer power
setting
                         during this part of the calibration. Solution is to detect the
overload
                         and lower the power on the synthesizer until the problem is solved.
*)
                       data=SettledSpectralData;
                       While[LastRawAcquisitionIFStatus!=0,
                             PrintMessage["Overload will be solved automatically by reducing
source

power."];
                             SetSourcePower[SourcePower-1];data=SettledSpectralData];
                       data
                            ][[i]][[
               Switch[port,1,{cha1,chb1},2,{cha2,chb2}]
                                 ]]
          ,{i,NofHarmonics}]
         ];

(* Function added to avoid overload during "ranging" calibrations *)
(* Used only during "thrucal" function                            *)
(* Jan Verspecht, 2-8-1999                                        *)

(* thrumaxcalpower := Min[dBm[ADCInputRanges]+CouplingLoss
                      +AttenuatorSettings+ConversionLoss]; *)

(* thrumaxcalpower was a quick fix for the overload problem during thru cals *)
(* it is obsolete because of the automated power adjustment when overload is detected *)

thrumaxcalpower := Infinity;

thrucal[setup_]:=
   Module[{calst,f0,forw12measurement,port1states,port2states},
      f0=setup[[1]];
      calst={setup[[2]],setup[[3]],setup[[4]],setup[[5,1]],setup[[5,2]]};
      forw12measurement[instrstate_]:=
        Module[{forw1,forw2},
           SetInstrumentState[instrstate];
           (* MODIFICATION marcvb 16 Dec 2001 *)
           (* Replacing the CalBox by TestSet: Forward1Mode -> ForwardPort1Mode *)
           ForwardPort1Mode;
           forw1=
              Table[
                    SetSourceFreq[i*f0];
                    SetSourcePower[Min[
                                       {
                       RelCalPowerFunction[i*f0][[1]],thrumaxcalpower
                                    }
                                      ]
                                 ];
                    Transpose[
                      (* Occasionally an ADC overflow may occur, due to ADC offset, coupler
ripple,...

which
                         make it impossible to accurately calculate maximum synthesizer power
setting
                         during this part of the calibration. Solution is to detect the
overload
                         and lower the power on the synthesizer until the problem is solved.
*)
                       data=SettledSpectralData;
                       While[LastRawAcquisitionIFStatus!=0,
                             PrintMessage["Overload will be solved automatically by reducing
source

power."];
                             SetSourcePower[SourcePower-1];data=SettledSpectralData];
                       data
                            ][[i]][[{cha1,chb1,cha2,chb2}]]
                            ,{i,NofHarmonics}];
           (* MODIFICATION marcvb 16 Dec 2001 *)
           (* Replacing the CalBox by TestSet: Forward2Mode -> ForwardPort2Mode *)
           ForwardPort2Mode;
           forw2=
              Table[
                    SetSourceFreq[i*f0];
                    SetSourcePower[Min[
                                       {
                       RelCalPowerFunction[i*f0][[2]],thrumaxcalpower
                                       }
                                      ]
                                 ];
                    Transpose[
```

```
                        (* Occasionally an ADC overflow may occur, due to ADC offset, coupler
ripple,...

which
                           make it impossible to accurately calculate maximum synthesizer power
setting
                           during this part of the calibration. Solution is to detect the
overload
                           and lower the power on the synthesizer until the problem is solved.
*)
                          data=SettledSpectralData;
                          While[LastRawAcquisitionIFStatus!=0,
                                PrintMessage["Overload will be solved automatically by reducing
source

power."];
                                SetSourcePower[SourcePower-1];data=SettledSpectralData];
                          data
                               ][[i]][[{cha1,chb1,cha2,chb2}]]
                      ,{i,NofHarmonics}];
             {forw1,forw2} //Transpose
               ];
       port1states=Map[
               {calst[[1]],calst[[2]],calst[[3]],
                {#[[1,1]],#[[1,2]],calst[[4,3]],calst[[4,4]]},
                {#[[2,1]],#[[2,2]],calst[[5,3]],calst[[5,4]]}}&
                 ,setup[[6,1]]
                      ];
       port2states=Map[
               {calst[[1]],calst[[2]],calst[[3]],
                {calst[[4,1]],calst[[4,2]],#[[1,1]],#[[1,2]]},
                {calst[[5,1]],calst[[5,2]],#[[2,1]],#[[2,2]]}}&
                 ,setup[[6,2]]
                      ];
      {forw12measurement[calst],
       If[port1states=={},
            Table[Null,{NofHarmonics}]
            (* adapted 27-7-1999 to allow absence of peak-voltage values *),
            (forw12measurement /@ port1states)//Transpose[#,{2,1,3,4}]&
         ],
       If[port2states=={},
            Table[Null,{NofHarmonics}]
            (* adapted 27-7-1999 to allow absence of peak-voltage values *),
            (forw12measurement /@ port2states)//Transpose[#,{2,1,3,4}]&
         ]
       } //Transpose
         ];

powercal[setup_]:=
   Module[{calinstrumentstate,f0},
      f0=setup[[1]];
      calinstrumentstate=
          {setup[[2]],setup[[3]],setup[[4]],setup[[5,1]],setup[[5,2]]};
      SetInstrumentState[calinstrumentstate];
      Table[
          SetSourceFreq[i*f0];
          SetSourcePower[
              Min[RelCalPowerFunction[i*f0][[1]],MaxPowerPowerCal[i*f0]]
                        ];
          Pause[DefaultSettlingTime];
          {MeasuredPower[i*f0],Transpose[
            SpectralData][[i]][[
                {{cha1,chb1},{cha2,chb2}}[[AuxiliaryCalPort]]
                                   ]]}
          ,{i,NofHarmonics}]
         ];

phasecal[setup_]:=
    Module[{calst,f0,fref},
       calst={setup[[2]],setup[[3]],setup[[4]],setup[[5,1]],setup[[5,2]]};
       SetInstrumentState[calst];
       f0=setup[[1]];
       fref=RefGenFrequency[f0];
       If[RefGenAuxPowerConnector,
          LevelSourcePower[RefGenCalPower[fref],fref],
          SetSourcePower[RefGenCalPower[fref]];
  SetSourceFreq[fref]
         ];
        SetSourceOutput[On];
        If[Not[SRDWarm],
           PrintMessage[" Warming up SRD. This takes "<>
                    ToString[SRDWarmingUpTime]<>" seconds."];
           Pause[SRDWarmingUpTime];
           SRDWarm=True;
           PrintMessage[" SRD warm. Continuing calibration."];
          ];
```

```
        refgendata=Table[PhaseNormalizedSpectralData[Switch[AuxiliaryCalPort,1,cha1,2,cha2],
               PhaseReference->RefGenAveragingPhaseReference][[
                 {{cha1,chb1},{cha2,chb2}}[[AuxiliaryCalPort]]
                                                          ]],
                {NofRefGenAverages}];
        refgendata=RefGenFineAlignPhase[refgendata,1];
        Sum[refgendata[[i]],{i,NofRefGenAverages}]/NofRefGenAverages //Transpose
          ];

(* Calculation Tools *)

(* Transformation matrices: waves to voltage-current *)

M4={{1,1,0,0},{1/50,-1/50,0,0},{0,0,1,1},{0,0,1/50,-1/50}} //N;

CalculateCalCoefficients:=
  If[MissingCalElements=!={},Message[CalculateCalCoefficients::calmissing],

  If[Not[StringQ[MatchResistance]],
    CalData[MatchResistances]={MatchResistance,MatchResistance};
    CalData[ShortResistances]={0,0}
    ];


  If[UTF==True,
 CalData["SHORT"]=Map[Transpose,{CalData["1| SHORT-OPEN |2"][[1]],
                  CalData["1| OPEN-SHORT |2"][[2]]}//Transpose];
CalData["OPEN"]=Map[Transpose,{CalData["1| OPEN-SHORT |2"][[1]],
                  CalData["1| SHORT-OPEN |2"][[2]]}//Transpose];
];


  Module[{LRRMcoeffandreflect},
     PrintMessage["Calculating LRRM coefficients.\nPlease be patient."];
     LRRMcoeffandreflect=
       Table[
         Table[
         PrintMessage["Setup: "<>ToString[i]<>"   FreqIndex: "<>ToString[j]];
         OneFreqLRRM[
            CalData["SHORT"][[i,j]],
            CalData["OPEN"][[i,j]],
            CalData["MATCH"][[i,j]],
            CalData["LINE"][[i,j,1]],
            N[Freq[i,j]],-1,1
                    ]
         ,{j,NofFreqs[i]}]
         ,{i,NofUserSetups}];
      LRRMCalCoefficients=Table[
              LRRMcoeffandreflect[[i,j]][[1]]
              ,{i,NofUserSetups},{j,NofFreqs[i]}];
      ReflectMatchEstimates=Table[
              LRRMcoeffandreflect[[i,j]][[2]]
              ,{i,NofUserSetups},{j,NofFreqs[i]}];
      GenCalCoefficients=Table[
              OneFreqGenCalCoefficients[i,j]
              ,{i,NofUserSetups},{j,NofFreqs[i]}];
     PrintMessage["Calculating K coefficients."];
      {KCoefficients,Tmatrices}=MapThread[
          CalculateK[#1,#2]&
          ,{LRRMCalCoefficients,GenCalCoefficients}]//Transpose;
      CalibrationMgr`Private`calCoefficients=Table[
              OneFreqCalCoefficients[i,j]
              ,{i,NofUserSetups},{j,NofFreqs[i]}]
          ]
    ];

(*
   More robust algorithm to find the phase of K.
   It is based on the a priori knowledge that the diagonal terms of the
   T-matrix description between AUX-quantities and DUT-quantities correspond to
   a delay line (linear phase with attenuation).
   Jan Verspecht, Santa Rosa, May 1st 2000
*)

CalculateK[LRRM_,Gen_]:=
  Module[{Kestsq,Kest,Test1,Testsq,TestsqN,time,maxpos,multiplier,
           multipliermatrix,Test2,test11n,Test3,nofh,range},
     range=Switch[AuxiliaryCalPort,1,{1,2},2,{3,4}];
     nofh = Length[LRRM];
     Kestsq = MapThread[
      Det[#1.Inverse[#2[[range, range]]]] &, {Gen,LRRM}];
     Kest = Sqrt[Kestsq];
     Test1 = MapThread[(#1.Inverse[#2[[range, range]]*#3])&,{Gen,LRRM,Kest}];
     Testsq=Transpose[Test1,{3,1,2}][[1,1]]^2;
     TestsqN=Testsq/Abs[Testsq];
     time = TimeData[TestsqN, DC -> False, MinNofSamples -> 16384, NofPeriods -> 1];
     maxpos = Position[Abs[time], Max[Abs[time]]][[1,1]];
```

```
     multiplier = Table[Exp[-I*Pi*i*maxpos/16384] // N, {i, nofh}];
     multipliermatrix = Map[{{#, 0}, {0, 1/#}} &, multiplier];
     Test2 = MapThread[(#1.Inverse[#2[[range, range]]*#3].Conjugate[#4])&,
          {Gen, LRRM, Kest, multipliermatrix}];
     test11n = Transpose[Test2,{3,1,2}][[1,1]];
     Kest=Kest*Sign[Re[test11n/Abs[test11n]]];
     Test3 = MapThread[(#1.Inverse[#2[[range, range]]*#3].Conjugate[#4])&,
          {Gen, LRRM, Kest, multipliermatrix}];
     {Kest,Test3}
        ];

OneFreqGenCalCoefficients[setup_,freq_]:=Module[
  {alfa,beta,gamma,delta,matrix,L,O,S,G,powerpeak,powervector,calmatrix,
   fund,harmindex,rawdata},
 fund=RefGenFrequency[CalibrationMgr`Private`userSetups[[setup,1]]];
 harmindex=Round[CalibrationMgr`Private`userSetups[[setup,1]]/fund];
 powerpeak=dBmToPeak[CalData["POWER"][[setup,freq,1]]];
 powervector=CalData["POWER"][[setup,freq,2]];
 L=reflection["Laux"][Freq[setup,freq]];
 O=reflection["Oaux"][Freq[setup,freq]];
 S=reflection["Saux"][Freq[setup,freq]];
 G=RefGenReflection[Freq[setup,freq]];
 rawdata=Map[
       CalData[#][[setup,freq]]&
       ,{"Laux","Oaux","Saux","HPR"}];
 matrix={
 {rawdata[[1,1]],rawdata[[1,2]],-L*rawdata[[1,1]],-L*rawdata[[1,2]]},
 {rawdata[[2,1]],rawdata[[2,2]],-O*rawdata[[2,1]],-O*rawdata[[2,2]]},
 {rawdata[[3,1]],rawdata[[3,2]],-S*rawdata[[3,1]],-S*rawdata[[3,2]]},
 {rawdata[[4,1]],rawdata[[4,2]],-G*rawdata[[4,1]],-G*rawdata[[4,2]]}
        };
 {alfa,beta,gamma,delta}=LinearSolve[matrix,
            {0,0,0,RefGenSpectralValue[fund][[harmindex*freq]]}];
 calmatrix={{alfa,beta},{gamma,delta}};
 calmatrix=
         calmatrix*powerpeak/Abs[(calmatrix.powervector)[[2]]]
                                                          ];

OneFreqCalCoefficients[setup_,freq_]:=
  Module[
  {port1ranging,port2ranging,alfa1,beta1,gamma1,delta1,
   alfa2,beta2,gamma2,delta2,delay},
  delay=Exp[-I*2*Pi*Freq[setup,freq]*LineDelay]//N;
  Off[Set::setraw];
  {{alfa1,beta1,0,0},
   {gamma1,delta1,0,0},
   {0,0,alfa2,beta2},
   {0,0,gamma2,delta2}}=
       KCoefficients[[setup,freq]]*LRRMCalCoefficients[[setup,freq]];
  On[Set::setraw];
  calculateranging1[data_]:=
    Module[{alfa,beta,gamma,delta,matrix},
        matrix={{data[[1,1]],data[[1,2]]},{data[[2,1]],data[[2,2]]}};
        {alfa,beta}=LinearSolve[matrix,
              {gamma2*data[[1,3]]+delta2*data[[1,4]],
               gamma2*data[[2,3]]+delta2*data[[2,4]]}
                                ]/delay;
        {gamma,delta}=LinearSolve[matrix,
              {alfa2*data[[1,3]]+beta2*data[[1,4]],
               alfa2*data[[2,3]]+beta2*data[[2,4]]}
                                  ]*delay;
        Inverse[{{alfa1,beta1},{gamma1,delta1}}].{{alfa,beta},{gamma,delta}}
          ];

  calculateranging2[data_]:=
    Module[{alfa,beta,gamma,delta,matrix},
        matrix={{data[[1,3]],data[[1,4]]},{data[[2,3]],data[[2,4]]}};
        {alfa,beta}=LinearSolve[matrix,
              {gamma1*data[[1,1]]+delta1*data[[1,2]],
               gamma1*data[[2,1]]+delta1*data[[2,2]]}
                                ]/delay;
        {gamma,delta}=LinearSolve[matrix,
              {alfa1*data[[1,1]]+beta1*data[[1,2]],
               alfa1*data[[2,1]]+beta1*data[[2,2]]}
                                  ]*delay;
        Inverse[{{alfa2,beta2},{gamma2,delta2}}].{{alfa,beta},{gamma,delta}}
          ];
 port1ranging=Map[calculateranging1[#]&,CalData["LINE"][[setup,freq]][[2]]];
 port2ranging=Map[calculateranging2[#]&,CalData["LINE"][[setup,freq]][[3]]];
 (* Adapted for voltage current representation in stead of waves *)

   {M4.(KCoefficients[[setup,freq]]*LRRMCalCoefficients[[setup,freq]]),
   port1ranging,port2ranging}
   ];

(* Map calibration elements with CalKit *)
```

```
      reflection["Laux"]=Reflection["Lfemale"];
      reflection["Oaux"]=Reflection["Ofemale"];
      reflection["Saux"]=Reflection["Sfemale"];

(* Code needed for doing calculations on LRRM *)

(* Faster procedure based upon linear interpolation *)

skmax=4;

OneFreqLRRM[reflectm1_,reflectm2_,match_,line_,freq_,reflectestimate1_,
                    reflectestimate2_]:=
   Module[{sk1,sk2,sksol},
    sk1=Abs[FastOneFreqLRRMMatchLC[reflectm1,reflectm2,match,line,freq,
          reflectestimate1,reflectestimate2,0,0,0*10^-12][[2,1]]]-1;
    sk2=Abs[FastOneFreqLRRMMatchLC[reflectm1,reflectm2,match,line,freq,
          reflectestimate1,reflectestimate2,0,0,skmax*10^-12][[2,1]]]-1;
    sksol = skmax*sk1/(sk1-sk2);
    FastOneFreqLRRMMatchLC[reflectm1,reflectm2,match,line,freq,
                           reflectestimate1,reflectestimate2,0,0,sksol*10^-12]
         ];

(* Speeding up through the usage of compiled functions *)

zfunction = Compile[{l, f, m, c},I*2*Pi*l*f + 1/(1/m + I*2*Pi*c*f)];

gammafunction = Compile[{{z, _Complex}}, (z - 50)/(z + 50)];

dfunction=Compile[{{freq,_Real},{linedelay,_Real},{skin,_Real}},
                  Exp[I*2*Pi*freq*linedelay+skin*freq]
                 ];

P0function = Compile[{{d,_Complex},{gammaload,_Complex},{a,_Complex,2},
                      {b,_Complex,2}},
     {(a[[1, 5]]*a[[2, 4]] - a[[1, 4]]*a[[2, 5]])/(a[[2, 5]]*b[[1, 4]] -
       a[[2, 4]]*b[[1, 5]]), (gammaload*(a[[1, 5]]*a[[2, 4]]*b[[1, 3]] -
        a[[1, 4]]*a[[2, 5]]*b[[1, 3]] + a[[1, 3]]*(a[[2, 5]]*b[[1, 4]] -
          a[[2, 4]]*b[[1, 5]]))))/(a[[1, 3]]*(a[[2, 5]]*b[[1, 4]] -
          a[[2, 4]]*b[[1, 5]]))),
     (d*gammaload*(a[[1, 5]]*a[[2, 4]]*b[[1, 3]] -
        a[[1, 4]]*a[[2, 5]]*b[[1, 3]] + a[[1, 3]]*(a[[2, 5]]*b[[1, 4]] -
        a[[2, 4]]*b[[1, 5]]))*(a[[1, 5]]*b[[2, 4]] -
        a[[1, 4]]*b[[2, 5]]))/(a[[1, 3]]*(a[[2, 5]]*b[[1, 4]] -
        a[[2, 4]]*b[[1, 5]])*(a[[2, 5]]*b[[2, 4]] - a[[2, 4]]*b[[2, 5]])),
     (d*gammaload*(a[[1, 5]]*a[[2, 4]] - a[[1, 4]]*a[[2, 5]])*
       (-(a[[1, 5]]*a[[2, 4]]*b[[1, 3]]) + a[[1, 4]]*a[[2, 5]]*b[[1, 3]] +
        a[[1, 3]]*(-(a[[2, 5]]*b[[1, 4]]) + a[[2, 4]]*b[[1, 5]]))))/
      (a[[1, 3]]*(a[[2, 5]]*b[[1, 4]] - a[[2, 4]]*b[[1, 5]])*
      (a[[2, 5]]*b[[2, 4]] - a[[2, 4]]*b[[2, 5]])), (a[[1, 5]]*b[[1, 4]] -
       a[[1, 4]]*b[[1, 5]])/(d*a[[2, 5]]*b[[1, 4]] - d*a[[2, 4]]*b[[1, 5]])}];

P1function = Compile[{{d,_Complex},{a,_Complex,2},{b,_Complex,2}},
      {0, -(b[[1, 3]]/a[[1, 3]]), (d*(-(a[[1, 5]]*b[[1, 3]]*b[[2, 4]]) +
        a[[1, 4]]*b[[1, 3]]*b[[2, 5]] + a[[1, 3]]*(b[[1, 5]]*b[[2, 4]] -
          b[[1, 4]]*b[[2, 5]]))))/(a[[1, 3]]*(a[[2, 5]]*b[[2, 4]] -
          a[[2, 4]]*b[[2, 5]])),
     (d*(-(a[[1, 5]]*a[[2, 4]]*b[[1, 3]]) + a[[1, 4]]*a[[2, 5]]*b[[1, 3]] +
        a[[1, 3]]*(-(a[[2, 5]]*b[[1, 4]]) + a[[2, 4]]*b[[1, 5]]))))/
      (a[[1, 3]]*(-(a[[2, 5]]*b[[2, 4]]) + a[[2, 4]]*b[[2, 5]])), 0}];

P2function =Compile[{{d,_Complex},{gammaload,_Complex},{a,_Complex,2},
                     {b,_Complex,2}},
      {(d*(a[[2, 5]]*b[[2, 4]] - a[[2, 4]]*b[[2, 5]]))/(a[[2, 5]]*b[[1, 4]] -
       a[[2, 4]]*b[[1, 5]]), (d*gammaload*b[[1, 3]]*(a[[2, 5]]*b[[2, 4]] -
        a[[2, 4]]*b[[2, 5]]))/(a[[1, 3]]*(a[[2, 5]]*b[[1, 4]] -
        a[[2, 4]]*b[[1, 5]])),
     (d^2*gammaload*b[[1, 3]]*(a[[1, 5]]*b[[2, 4]] - a[[1, 4]]*b[[2, 5]]))/
      (a[[1, 3]]*(a[[2, 5]]*b[[1, 4]] - a[[2, 4]]*b[[1, 5]])),
     (d^2*gammaload*(a[[1, 5]]*a[[2, 4]] - a[[1, 4]]*a[[2, 5]])*b[[1, 3]])/
      (a[[1, 3]]*(-(a[[2, 5]]*b[[1, 4]]) + a[[2, 4]]*b[[1, 5]])),
     (b[[1, 5]]*b[[2, 4]] - b[[1, 4]]*b[[2, 5]])/
     (a[[2, 5]]*b[[1, 4]] - a[[2, 4]]*b[[1, 5]])}];

coefflistsfunction= Compile[{{a,_Complex,2},{b,_Complex,2},
                             {p0,_Complex,1},{p1,_Complex,1},{p2,_Complex,1}},
    {{{-(a[[1, 1]]*(b[[2, 1]]*p0[[2]]*p0[[4]] + a[[2, 1]]*
            (p0[[2]]*p0[[3]] - p0[[5]]))) + a[[2, 1]]*b[[1, 1]]*p0[[1]]*p0[[5]],
           -(a[[1, 1]]*(b[[2, 1]]*(-1 + p0[[4]]*p2[[2]] + p0[[2]]*p2[[4]]) +
          a[[2, 1]]*(p0[[3]]*p2[[2]] + p0[[2]]*p2[[3]] - p2[[5]]))) +
        b[[1, 1]]*(b[[2, 1]]*p0[[1]] + a[[2, 1]]*(p0[[5]]*p2[[1]] +
            p0[[1]]*p2[[5]])), -(a[[1, 1]]*p2[[2]]*(a[[2, 1]]*p2[[3]] +
          b[[2, 1]]*p2[[4]])) + b[[1, 1]]*p2[[1]]*(b[[2, 1]] +
          a[[2, 1]]*p2[[5]])},
      {-(b[[2, 1]]*(b[[1, 1]]*p0[[4]] + a[[1, 1]]*(p0[[4]]*p1[[2]] +
             p0[[2]]*p1[[4]]))) + a[[2, 1]]*
         (a[[1, 1]]*(-(p0[[3]]*p1[[2]]) - p0[[2]]*p1[[3]] + p1[[5]]) +
          b[[1, 1]]*(-p0[[3]] + p0[[5]]*p1[[1]] + p0[[1]]*p1[[5]])),
```

```
        -(a[[1, 1]]*(a[[2, 1]]*(p1[[3]]*p2[[2]] + p1[[2]]*p2[[3]]) +
            b[[2, 1]]*(p1[[4]]*p2[[2]] + p1[[2]]*p2[[4]]))) +
         b[[1, 1]]*(b[[2, 1]]*(p1[[1]] - p2[[4]]) + a[[2, 1]]*
            (p1[[5]]*p2[[1]] - p2[[3]] + p1[[1]]*p2[[5]])), 0},
       {-(b[[2, 1]]*(b[[1, 1]] + a[[1, 1]]*p1[[2]])*p1[[4]]) -
         a[[2, 1]]*(a[[1, 1]]*p1[[2]]*p1[[3]] + b[[1, 1]]*
            (p1[[3]] - p1[[1]]*p1[[5]])), 0, 0}},
      {{-(a[[1, 2]]*(b[[2, 2]]*p0[[2]]*p0[[4]] + a[[2, 2]]*
          (p0[[2]]*p0[[3]] - p0[[5]])) + a[[2, 2]]*b[[1, 2]]*p0[[1]]*p0[[5]],
        -(a[[1, 2]]*(b[[2, 2]]*(-1 + p0[[4]]*p2[[2]] + p0[[2]]*p2[[4]]) +
            a[[2, 2]]*(p0[[3]]*p2[[2]] + p0[[2]]*p2[[3]] - p2[[5]]))) +
         b[[1, 2]]*(b[[2, 2]]*p0[[1]] + a[[2, 2]]*(p0[[5]]*p2[[1]] +
             p0[[1]]*p2[[5]])), -(a[[1, 2]]*p2[[2]]*(a[[2, 2]]*p2[[3]] +
            b[[2, 2]]*p2[[4]])) + b[[1, 2]]*p2[[1]]*(b[[2, 2]] +
           a[[2, 2]]*p2[[5]])},
       {-(b[[2, 2]]*(b[[1, 2]]*p0[[4]] + a[[1, 2]]*(p0[[4]]*p1[[2]] +
              p0[[2]]*p1[[4]]))) + a[[2, 2]]*
          (a[[1, 2]]*(-(p0[[3]]*p1[[2]]) - p0[[2]]*p1[[3]] + p1[[5]]) +
           b[[1, 2]]*(-p0[[3]] + p0[[5]]*p1[[1]] + p0[[1]]*p1[[5]])),
        -(a[[1, 2]]*(a[[2, 2]]*(p1[[3]]*p2[[2]] + p1[[2]]*p2[[3]]) +
            b[[2, 2]]*(p1[[4]]*p2[[2]] + p1[[2]]*p2[[4]]))) +
         b[[1, 2]]*(b[[2, 2]]*(p1[[1]] - p2[[4]]) + a[[2, 2]]*
            (p1[[5]]*p2[[1]] - p2[[3]] + p1[[1]]*p2[[5]])), 0},
       {-(b[[2, 2]]*(b[[1, 2]] + a[[1, 2]]*p1[[2]])*p1[[4]]) -
         a[[2, 2]]*(a[[1, 2]]*p1[[2]]*p1[[3]] + b[[1, 2]]*
            (p1[[3]] - p1[[1]]*p1[[5]])), 0, 0}}};

newcoefflistfunction = Compile[ {{a,_Complex,2},{b,_Complex,2},
                          {p0,_Complex,1},{p1,_Complex,1},{p2,_Complex,1}},
    {{-((-(a[[1, 2]]*(b[[2, 2]]*p0[[2]]*p0[[4]] + a[[2, 2]]*
              (p0[[2]]*p0[[3]] - p0[[5]]))) + a[[2, 2]]*b[[1, 2]]*p0[[1]]*
            p0[[5]])*(-(b[[2, 1]]*(b[[1, 1]] + a[[1, 1]]*p1[[2]])*
             p1[[4]]) - a[[2, 1]]*(a[[1, 1]]*p1[[2]]*p1[[3]] +
             b[[1, 1]]*(p1[[3]] - p1[[1]]*p1[[5]]))))) +
         (-(a[[1, 1]]*(b[[2, 1]]*p0[[2]]*p0[[4]] + a[[2, 1]]*
               (p0[[2]]*p0[[3]] - p0[[5]]))) +
          a[[2, 1]]*b[[1, 1]]*p0[[1]]*p0[[5]])*
          (-(b[[2, 2]]*(b[[1, 2]] + a[[1, 2]]*p1[[2]])*p1[[4]]) -
           a[[2, 2]]*(a[[1, 2]]*p1[[2]]*p1[[3]] + b[[1, 2]]*
              (p1[[3]] - p1[[1]]*p1[[5]]))),
        (-(b[[2, 2]]*(b[[1, 2]] + a[[1, 2]]*p1[[2]])*p1[[4]]) -
           a[[2, 2]]*(a[[1, 2]]*p1[[2]]*p1[[3]] + b[[1, 2]]*
              (p1[[3]] - p1[[1]]*p1[[5]])))*
          (-(a[[1, 1]]*(b[[2, 1]]*(-1 + p0[[4]]*p2[[2]] + p0[[2]]*p2[[4]]) +
              a[[2, 1]]*(p0[[3]]*p2[[2]] + p0[[2]]*p2[[3]] - p2[[5]]))) +
           b[[1, 1]]*(b[[2, 1]]*p0[[1]] + a[[2, 1]]*(p0[[5]]*p2[[1]] +
               p0[[1]]*p2[[5]]))) -
         (-(b[[2, 1]]*(b[[1, 1]] + a[[1, 1]]*p1[[2]])*p1[[4]]) -
           a[[2, 1]]*(a[[1, 1]]*p1[[2]]*p1[[3]] + b[[1, 1]]*
              (p1[[3]] - p1[[1]]*p1[[5]])))*
          (-(a[[1, 2]]*(b[[2, 2]]*(-1 + p0[[4]]*p2[[2]] + p0[[2]]*p2[[4]]) +
              a[[2, 2]]*(p0[[3]]*p2[[2]] + p0[[2]]*p2[[3]] - p2[[5]]))) +
           b[[1, 2]]*(b[[2, 2]]*p0[[1]] + a[[2, 2]]*(p0[[5]]*p2[[1]] +
               p0[[1]]*p2[[5]]))),
        (-(b[[2, 2]]*(b[[1, 2]] + a[[1, 2]]*p1[[2]])*p1[[4]]) -
           a[[2, 2]]*(a[[1, 2]]*p1[[2]]*p1[[3]] + b[[1, 2]]*
              (p1[[3]] - p1[[1]]*p1[[5]])))*
          (-(a[[1, 1]]*p2[[2]]*(a[[2, 1]]*p2[[3]] + b[[2, 1]]*p2[[4]])) +
           b[[1, 1]]*p2[[1]]*(b[[2, 1]] + a[[2, 1]]*p2[[5]])) -
         (-(b[[2, 1]]*(b[[1, 1]] + a[[1, 1]]*p1[[2]])*p1[[4]]) -
           a[[2, 1]]*(a[[1, 1]]*p1[[2]]*p1[[3]] + b[[1, 1]]*
              (p1[[3]] - p1[[1]]*p1[[5]])))*
          (-(a[[1, 2]]*p2[[2]]*(a[[2, 2]]*p2[[3]] + b[[2, 2]]*p2[[4]])) +
           b[[1, 2]]*p2[[1]]*(b[[2, 2]] + a[[2, 2]]*p2[[5]]))},
       {-((-(b[[2, 2]]*(b[[1, 2]]*p0[[4]] + a[[1, 2]]*(p0[[4]]*p1[[2]] +
                p0[[2]]*p1[[4]]))) + a[[2, 2]]*(a[[1, 2]]*(-(p0[[3]]*p1[[2]]) -
           p0[[2]]*p1[[3]] + p1[[5]]) + b[[1, 2]]*(-p0[[3]] + p0[[5]]*p1[[1]] +
                 p0[[1]]*p1[[5]]))*(-(b[[2, 1]]*(b[[1, 1]] +
                a[[1, 1]]*p1[[2]])*p1[[4]]) - a[[2, 1]]*
              (a[[1, 1]]*p1[[2]]*p1[[3]] + b[[1, 1]]*(p1[[3]] -
                 p1[[1]]*p1[[5]])))) +
         (-(b[[2, 1]]*(b[[1, 1]]*p0[[4]] + a[[1, 1]]*(p0[[4]]*p1[[2]] +
                p0[[2]]*p1[[4]]))) + a[[2, 1]]*
             (a[[1, 1]]*(-(p0[[3]]*p1[[2]]) - p0[[2]]*p1[[3]] + p1[[5]]) +
              b[[1, 1]]*(-p0[[3]] + p0[[5]]*p1[[1]] + p0[[1]]*p1[[5]])))*
          (-(b[[2, 2]]*(b[[1, 2]] + a[[1, 2]]*p1[[2]])*p1[[4]]) -
           a[[2, 2]]*(a[[1, 2]]*p1[[2]]*p1[[3]] + b[[1, 2]]*
              (p1[[3]] - p1[[1]]*p1[[5]]))),
        (-(b[[2, 2]]*(b[[1, 2]] + a[[1, 2]]*p1[[2]])*p1[[4]]) -
           a[[2, 2]]*(a[[1, 2]]*p1[[2]]*p1[[3]] + b[[1, 2]]*
              (p1[[3]] - p1[[1]]*p1[[5]])))*
          (-(a[[1, 1]]*(a[[2, 1]]*(p1[[3]]*p2[[2]] + p1[[2]]*p2[[3]]) +
              b[[2, 1]]*(p1[[4]]*p2[[2]] + p1[[2]]*p2[[4]]))) +
           b[[1, 1]]*(b[[2, 1]]*(p1[[1]] - p2[[4]]) + a[[2, 1]]*
              (p1[[5]]*p2[[1]] - p2[[3]] + p1[[1]]*p2[[5]]))) -
         (-(b[[2, 1]]*(b[[1, 1]] + a[[1, 1]]*p1[[2]])*p1[[4]]) -
           a[[2, 1]]*(a[[1, 1]]*p1[[2]]*p1[[3]] + b[[1, 1]]*
```

```
            (p1[[3]] - p1[[1]]*p1[[5]])))*
        (-(a[[1, 2]]*(a[[2, 2]]*(p1[[3]]*p2[[2]] + p1[[2]]*p2[[3]]) +
            b[[2, 2]]*(p1[[4]]*p2[[2]] + p1[[2]]*p2[[4]]))) +
         b[[1, 2]]*(b[[2, 2]]*(p1[[1]] - p2[[4]]) + a[[2, 2]]*
            (p1[[5]]*p2[[1]] - p2[[3]] + p1[[1]]*p2[[5]]))), 0}, {0, 0, 0}}];

gammasolutionsfunction = Compile[{{delta1,_Complex},{delta2,_Complex},
                                  {a,_Complex,2},{b,_Complex,2},
        {p0,_Complex,1},{p1,_Complex,1},{p2,_Complex,1}},
      {(a[[1, 1]] + b[[1, 1]]*(p0[[1]] + delta1*p1[[1]] + delta2*p2[[1]]))/
  (delta1*b[[1, 1]] + a[[1, 1]]*(p0[[2]] + delta1*p1[[2]] + delta2*p2[[2]])),
 (a[[1, 2]] + b[[1, 2]]*(p0[[1]] + delta1*p1[[1]] + delta2*p2[[1]]))/
  (delta1*b[[1, 2]] + a[[1, 2]]*(p0[[2]] + delta1*p1[[2]] + delta2*p2[[2]]))}
                                 ];

FastOneFreqLRRMMatchLC[reflectm1_,reflectm2_,match_,line_,freq_,
                       reflectestimate1_,reflectestimate2_,l_,c_,skin_]:=
 Module[{gammaload,d,M,P0,P1,P2,matches,
         gamma1,beta1,delta1,alpha2,beta2,gamma2,delta2,calmatrix,
         awaves,bwaves,coefflists,newcoefflist,rule,numerator,delta2rules,
         delta1rules,deltasolutions,
         gammasolutions,gammares,solutionindex},

   (* Zc resource is overwritten *)
   (* Zc will disappear as a resource *)
   matchResistances=CalData[MatchResistances]-CalData[ShortResistances];
   (* The quadratic equations become singular for Z=50 (delta1 is not present in the equation
for Z=50
      and thus cannot be eliminated). Following solutions needs to be implemented more
carefully ! *)
   matchResistances=Map[If[# == 50 || # == 50., 50.1, #]&, matchResistances];
   gammaload=gammafunction[zfunction[l,freq,matchResistances[[1]],c]];
   d=dfunction[freq,LineDelay,skin];

   (* Defining awaves and bwaves *)
   M =  Transpose[
                 {
                   Flatten[reflectm1],
                   Flatten[reflectm2],
                   Flatten[{match[[1]],match[[2]]}],
                   line[[1]],
                   line[[2]]
                  }
                 ];
    awaves = M[[{1,3}]];
    bwaves = M[[{2,4}]];

    {P0,P1,P2}={P0function[d,gammaload,awaves,bwaves],
                P1function[d,awaves,bwaves],
                P2function[d,gammaload,awaves,bwaves]};

 (* Define linear relation between {beta1, ..., gamma2} and {delta1, delta2} *)
    {beta1, gamma1, alpha2, beta2, gamma2} = P0 + P1*delta1 + P2*delta2;

    (* Algorithm to solve for the 2 coupled quadratic equations *)

    coefflists=coefflistsfunction[awaves,bwaves,P0,P1,P2];
    newcoefflist = newcoefflistfunction[awaves,bwaves,P0,P1,P2];
    (* 27-03-2002 Jan and Ewout  *)
    (* Chop introduced to circumvent problem with Solve if 0.*variable occurs in equation *)
    (* Mathematica does not simplify 0.*variable automatically to 0 !! *)
    rule=Solve[Chop[FromCoeffList[newcoefflist,{delta1,delta2}] == 0,10^-18],{delta1}][[1]];
    numerator =  Numerator[Simplify[FromCoeffList[coefflists[[2]],
                                    {delta1,delta2}] /. rule
                                     ]
                           ];
    delta2rules=Solve[numerator==0,{delta2}];
    delta1rules=(rule /. delta2rules);
    deltarules = Map[Flatten,Transpose[{delta1rules,delta2rules}],3];
    deltasolutions = Transpose[{delta1,delta2} /. deltarules];
    gammasolutions = MapThread[gammasolutionsfunction[#1,#2,awaves,bwaves,P0,
                               P1,P2]&,deltasolutions];

(*  Following line adapted for more robust choice of roots           *)
(*  Jan Verspecht, 21 February 2002                                  *)
(*  gammares = Map[(Plus @@  Abs[# - {-1, 1}]^2) &, gammasolutions]  *);
    gammares = Map[(Re[#[[2]]]-Re[#[[1]]])&,gammasolutions];
    solutionindex = Select[{1,2,3,4},(gammares[[#]]==Max[gammares])&];
    calmatrix =
     ({
       {1,beta1,0,0},
       {gamma1,delta1,0,0},
       {0,0,alpha2,beta2},
       {0,0,gamma2,delta2}
      }
          /. deltarules[[solutionindex]])[[1]];
    matches = With[{dut=calmatrix.Transpose[M][[3]]},
```

```
                {ToImpedance[dut[[2]]/dut[[1]]],
                 ToImpedance[dut[[4]]/dut[[3]]]};
    {calmatrix,
    {gammasolutions[[solutionindex,1]][[1]],
    gammasolutions[[solutionindex,2]][[1]],matches[[1]],matches[[2]]}
    }
     ];

ToImpedance=Compile[{{gamma,_Complex}},(1+gamma)/(1-gamma)*50];

FromCoeffList[list_,any_]:=Apply[Plus,Flatten[Table[Table[list[[i,j]]*
                      any[[1]]^(i-1)*any[[2]]^(j-1),{j,1,4-i}],{i,3}]]];


(* The following function is present for historical reasons in order to trace back
   how the FastOneFreqLRRMMatchLC was developed *)
(*
SlowOneFreqLRRMMatchLC[reflectm1_,reflectm2_,match_,line_,freq_,reflectestimate1_,reflectestim
ate2_,l_,c

_,skin_]:=
 Module[{gammaload,d,z,M,V1,V2,V3,A,B,P0,P1,P2,lu,
        (* gamma1,beta1,delta1,alpha2,beta2,gamma2,delta2,*)
        reflectsolutions,reflect1,reflect2},

   (* Zc resource is overwritten *)
   (* Zc will disappear as a resource *)
   matchResistance=
     (CalData[MatchResistances][[1]]-CalData[ShortResistances][[1]]+
      CalData[MatchResistances][[2]]-CalData[ShortResistances][[2]])/2;
   Zc=50;
   z=I*2*Pi*l*freq+1/(1/matchResistance+I*2*Pi*c*freq) //N;
   gammaload=(z-Zc)/(z+Zc) //N;
   d=Exp[I*2*Pi*freq*LineDelay+Sqrt[skin*freq]] //N;

   (* Defining the measured A and B matrices *)
   A=Array[0&,{2,5}];
   B=Array[0&,{2,5}];
   {{A[[1,1]],B[[1,1]]},{A[[2,1]],B[[2,1]]}}=reflectm1;
   {{A[[1,2]],B[[1,2]]},{A[[2,2]],B[[2,2]]}}=reflectm2;
   {A[[1,3]],B[[1,3]]}=match[[1]];
   {A[[1,4]],B[[1,4]],A[[2,4]],B[[2,4]]}=line[[1]];
   {A[[1,5]],B[[1,5]],A[[2,5]],B[[2,5]]}=line[[2]];

   (* Defining the necessary matrices and vectors *)
   V1={-gammaload*A[[1,3]],  -A[[1,4]],           0,  -A[[1,5]],           0};
   V2={           B[[1,3]],          0, -d*B[[1,4]],          0, -d*B[[1,5]]};
   V3={                  0, d*B[[2,4]],           0, d*B[[2,5]],           0};
   M={
      {gammaload*B[[1,3]],  -A[[1,3]],         0,         0,           0},
      {          B[[1,4]],          0,         0,         0, -d*A[[2,4]]},
      {                 0, d*A[[1,4]], -A[[2,4]], -B[[2,4]],           0},
      {          B[[1,5]],          0,         0,         0, -d*A[[2,5]]},
      {                 0, d*A[[1,5]], -A[[2,5]], -B[[2,5]],           0}
     };
    lu = LUFactor[M];
    {P0,P1,P2}=Map[LUSolve[lu,#]&,{V1,V2,V3}];

    (* Define linear relation between {beta1, ..., gamma2} and {delta1, delta2} *)
    {beta1, gamma1, alpha2, beta2, gamma2} = P0 + P1*delta1 + P2*delta2;

    (* Algorithm to solve for the 2 coupled quadratic equations *)

    coefflists={
         CoefficientList[(A[[1,1]]+beta1*B[[1,1]])*(gamma2*A[[2,1]]+delta2*B[[2,1]]) -
         (alpha2*A[[2,1]]+beta2*B[[2,1]])*(gamma1*A[[1,1]]+delta1*B[[1,1]]),{delta1,delta2}],
         CoefficientList[(A[[1,2]]+beta1*B[[1,2]])*(gamma2*A[[2,2]]+delta2*B[[2,2]]) -
         (alpha2*A[[2,2]]+beta2*B[[2,2]])*(gamma1*A[[1,2]]+delta1*B[[1,2]]),{delta1,delta2}]
                };
    newcoefflist = coefflists[[1]]*coefflists[[2,3,1]]-coefflists[[2]]*coefflists[[1,3,1]];
    denominator = (delta2*newcoefflist[[2,2]]+newcoefflist[[2,1]]);
    rule=Solve[FromCoeffList[newcoefflist,{delta1,delta2}] == 0,{delta1}][[1]];
    delta2rules=Solve[Simplify[(FromCoeffList[coefflists[[2]],{delta1,delta2}] /.

rule)*denominator^2]==0,
                    {delta2}];
    delta1rules=(rule /. delta2rules);
    deltarules = Map[Flatten,Transpose[{delta1rules,delta2rules}],3];
    gammasolutions = {(A[[1,1]]+beta1*B[[1,1]])/(gamma1*A[[1,1]]+delta1*B[[1,1]]),
                      (A[[1,2]]+beta1*B[[1,2]])/(gamma1*A[[1,2]]+delta1*B[[1,2]])} /.
deltarules;
    gammares = Map[(Plus @@  Abs[# - {-1, 1}]^2) &, gammasolutions];
    solutionindex = Select[{1,2,3,4},(gammares[[#]]==Min[gammares])&];
    {
     ({{1,beta1,0,0},{gamma1,delta1,0,0},{0,0,alpha2,beta2},{0,0,gamma2,delta2}
      } /. deltarules[[solutionindex]])[[1]],
    gammasolutions[[solutionindex,1]][[1]],
```

```
    gammasolutions[[solutionindex,2]][[1]]
    }
     ];
*)

End[];

EndPackage[];
```

Programme 'Scilab' : Visualisation et traitement de maillages de lignes mesurés par sonde HIP

Pour installer le programme d'analyse sous Scilab, on doit placer dans un répertoire de travail 3 fichiers :

- un raccourssi de Scilab, afin que le répertoire courant soit le répertoire par défault ;
- un « .scilab » qui est la configuration spécifique de Scilab lorsqu'il est lancé du répertoire de travail : un menu HIP est alors disponible dans Scilab ;
- le fichier source : « HIP.sci ».

Fig. 52 - Fichiers à placer dans le répertoire de travail

FICHIER CONFIGURATION « .scilab »

Il faut placer dans ce fichier texte dont le listing est ci-dessous le chemin du repertoire de travail.

```
HIP='exec(''E:\POST_DOC_TIBO\Tibo_Scilab\LSNA\HIP_modifmtrl.sci'');'
addmenu('HIP Process',list(0,'HIP'));
```

FICHIER SOURCE « HIP.sci »

```
// ##################################################################
// ###      Principales variables utilisées dans le programme      ###
// ##################################################################
//
// _M : Matrice (f,z) de mesure de la tension corrigée [VOLT/RADIAN]
// _V1 : Vecteur (f) de la tension mesurée dans le plan d'etalonnage du LSNA port 1 (optionel) [VOLT]
// _V2 : Idem - port 2 (optionel) [VOLT]
// _K : Vecteur (f) de la correction appliquée sur les Raw-Data (optionel) [VOLT/RADIAN]
//
// _nbf : Nombre de fréquences
// _nbz : Nombre de positions
//
// _fundamental : Pas fréquentiel (et premier point) de la table de fréquences pour les mesures [Hz]
// _zstep : Pas spatial des mesures [m]
//
// NB : Unité de gamma utilisée [u/m] => alpha [Neper/m] - beta [Radian/m]
//
// _delay_estimate : vecteur (f) - retard estimé de la ligne mesurée dans toute la bande [seconde/m]
// _gamma_estimate : vecteur (f) - estimation de gamma(f)=alpha(f)+j.beta(f) d'après les paramètres de la ligne mesurée [u/m]
//
// _delay_measure : vecteur (f) - retard issue du gamma (f) des mesures en 3 points equidistants [seconde/m]
// _gamma_measure : vecteur (f) - gamma (f) ccalculé à partir de 3 mesures equidistantes [u/m]
//
// _VP : Matrice Onde de tension incidente
// _VM : Matrice Onde de tension réfléchie
//
global _M; global _V1; global _V2; global _K;
global _fundamental; global _zstep;
global _nbf; global _nbz;
global _gamma_estimate; global _delay_estimate;
global _gamma_measure; global _delay_measure;
global _VP; global _VM;


// ##################################################################
// ###             Fonction Mathématiques Elementaires            ###
// ##################################################################
%c=3*10^8;
deff('[x]=phase(y)','if y==0 then x=0, else x=atan(imag(y),real(y)); end;');
deff('[x]=dB(y)','x=20*log10(abs(y))');


// ##################################################################
```

```
// ###                  Paramètres supplémentaires                ###
// ##################################################################
for i=1:32, bluecolormap(i,1)=0.5+(i/32)*0.5;bluecolormap(i,2)=0.3+(i/32)*0.7; bluecolormap(i,3)=1; end;
clear i;


// ##################################################################
// ### Lecture des mesure stockée avec "Export" dans 'Mathematica' ###
// ##################################################################
function Load_File()  // (_M,_V1,_V2,_K,_fundamental,_zstep,_nbf,_nbz)
global _M; global _V1; global _V2; global _K; global _fundamental; global _zstep; global _nbf; global _nbz;
l1=list('Unité du module',2,['Volt','dBV']);
l2=list('Unité de la phase',2,['Radian','Degré']);
l3=list('Contenu des fichiers',2,['V1, V2, K et Mesures','Mesures seulement']);
rep=x_choices('Spécifications des fichiers',list(l1,l2,l3))
my_title(1)='Matrice du module (Volt)';
my_title(2)='Matrice de la phase (Radian)';
if (rep(1)==2) then module_title(1)='Matrice du module (dBV)'; end;
if (rep(2)==2) then my_title(2)='Matrice de la phase (Degré)'; end;
f1=xgetfile('*.txt','T:\Département DTS-AE-TTL-HY/Tibo\Scilab_HIP',title=my_title(1));
f2=xgetfile('*.txt','T:\Département DTS-AE-TTL-HY/Tibo\Scilab_HIP',title=my_title(2));
M=excel2sci(f1,";");
P=excel2sci(f2,";");
M=evstr(M);
P=evstr(P);
if (rep(1)==2) then M=10.^(M./20); end;
if (rep(2)==2) then P=P./180*%pi; end;
_M=M.*exp(%i.*P);
[_nbf,_nbz]=size(_M);
_K=linspace(1,_nbf,_nbf);
_V1=linspace(1,_nbf,_nbf);
_V2=linspace(1,_nbf,_nbf);
if (rep(3)==1) then
_K=_M(:,3);
_V1=_M(:,1);
_V2=_M(:,2);
_M=_M(:,4:_nbz);
[_nbf,_nbz]=size(_M);
end;
l1=["Fondamental (GHz)";"Delta Z (mm)"];
[ok,_fundamental,_zstep]=getvalue("Entrez les paramètres des mesures",l1,list("vec",1,"vec",1),["1.0";"0.3175"]);
_fundamental=_fundamental*10^9;
_zstep=_zstep*10^(-3);
//clear l1; clear l2; clear l3; clear rep; clear my_title; clear f1; clear f2; clear M; clear P;
endfunction;


// ##################################################################
// ###          Estimation de gamma(f)=alpha(f)+j.beta(f)          ###
// ##################################################################

// #### Estimation du NIST - Rapport n°2 - p.13 ####
function Calcul_gamma_1()
global _gamma_estimate;
global _delay_estimate;
l1=["E'";"E"'"];
[ok,e1,e2]=getvalue("Entrez les paramètres de la ligne mesurée @ 1 GHz",l1,list("vec",1,"vec",1),["9.96";"6.72"]);
f=linspace(1,_nbf,_nbf);
delay=(1/(100*%c))*real(sqrt(e1+%i*e2))*10^12;// Delay à 1 GHz par cm de ligne
x_message(['Retard de la ligne @ 1 GHz';'-----------------------';'     '+string(delay)+' ps/cm']);
_gamma_estimate(1)=(%i*2*%pi*_fundamental/%c)*sqrt(e1+%i*e2*(10^9/_fundamental));// en [unité par m]
for i=2:_nbf, _gamma_estimate(i)=real(_gamma_estimate(1))+%i*imag(_gamma_estimate(1))*(i);end;
for i=1:_nbf, _delay_estimate(i)=imag(_gamma_estimate(i))/(2*%pi*i*_fundamental); end;
//clear ok; clear e1; clear e2; clear f; clear delay;
endfunction;

// #### Estimation simple : retard pur ####
function Calcul_gamma_2()
global _gamma_estimate;
global _delay_estimate;
l1=["Retard (ps/cm)"];
[ok,delay]=getvalue("Entrez les paramètres de la ligne mesurée",l1,list("vec",1),["105.24"]);
_delay_estimate=delay*10^(-10);// en [seconde / metre]
f=linspace(1,_nbf,_nbf);
alpha_estimate=linspace(0,0,_nbf)';
beta_estimate=f';
beta_estimate=2*%pi*beta_estimate*_delay_estimate*_fundamental;
_gamma_estimate=alpha_estimate+%i*beta_estimate;
f=linspace(1,1,_nbf);
_delay_estimate=_delay_estimate*f'
//clear beta_estimate; clear alpha_estimate; clear f; clear ok; clear l1; clear _delay;
endfunction;

// #### Estimation de gamma (unité/m) ####
function Estimate_gamma()
n=x_choose(['Permitivité effective complexe';'Retard de la ligne'],...
['Estimation de la constante de propagation';'Choisissez une méthode']);
if n==1 then
Calcul_gamma_1();
else
Calcul_gamma_2();
end
//clear n;
endfunction;


// ##################################################################
// ###       Calcul du Gamma mesuré via 3 points de mesures        ###
// ##################################################################
// Formule : cf. rapport n°2, p.16
// Probleme n°1 : choix de la racine issue du Sqrt().
//                on doit considérer 2 solutions potentielles
// Probleme n°2 : determination de Im{gamma} à 2k%pi près
```

```
// Solution : Algorithme du MTRL (cf. rapport n°2, p.13)
//
function Measure_gamma(my_gamma)
global _gamma_measure; global _delay_measure;
labels=["Position de V1";"Position de V2"];
[ok,z1,z2]=getvalue(...
"Entrez les positions de V1 et V2 pour la déduction de la constante de propagation à partir des mesures",labels,...
list("vec",1,"vec",1),["1";"2"]);
z3=z2+(z2-z1); // Attention, il faut z3<_nbz

V1=_M(:,z1); V2=_M(:,z2); V3=_M(:,z3);
L=(z2-z1)*_zstep;

// ##########################################
// Déduction NON CORRIGEE pour visualiser les
// probleme 1 et 2 de la determination de gamma
// Inutile dans le programme final

EGL1=V1+V3+sqrt((V1+V3)^2-4*V2^2)./(2*V2);
EGL2=V1+V3-sqrt((V1+V3)^2-4*V2^2)./(2*V2);

alpha_1=log(abs(EGL1))/L;// Alpha [Neper/m]
alpha_2=log(abs(EGL2))/L;

beta_1=phase(EGL1)/L;// Beta [Radian/m]
beta_2=phase(EGL2)/L;

tau_1=linspace(1,_nbf,_nbf)';
tau_1=2*%pi*tau_1*_fundamental;
tau_1=beta_1./tau_1;

tau_2=linspace(1,_nbf,_nbf)';
tau_2=2*%pi*tau_2*_fundamental;
tau_2=beta_2./tau_2;
// ##########################################

// Déduction de gamma corrigé : algorithme du MTRL
// Attention : erreur ds la formule de DC DeGroot

// #### 1- Unwarpping

// Modif du code :
// ... -imag(-log(EGL...
// EGL1 et EGL2
a=(EGL1+(EGL2^(-1)))/2;
b=(EGL2+(EGL1^(-1)))/2;
EGL1=a;
EGL2=b;

my_gamma=my_gamma*L// Estimation de gamma en unité/(distance mesurée)
GL1=log(EGL1)+%i*2*%pi*round((imag(my_gamma)-imag(-log(EGL1)))/(2*%pi) );
GL2=log(EGL2)+%i*2*%pi*round((imag(my_gamma)-imag(-log(EGL2)))/(2*%pi) );

// #### 2- Best Solution
for i=1:_nbf,
if ( abs(GL1(i)-my_gamma(i))/abs(my_gamma(i)) )<( abs(GL2(i)-my_gamma(i))/abs(my_gamma(i)) ) then
  GL_corr(i)=GL1(i);
else GL_corr(i)=GL2(i);
end;
end;
// ##########################################
alpha_corr=real(GL_corr);
beta_corr=imag(GL_corr);

tau_corr=linspace(1,_nbf,_nbf)';
tau_corr=2*%pi*tau_corr*_fundamental;
tau_corr=beta_corr./tau_corr;

_gamma_measure=GL_corr/L;
_delay_measure=tau_corr/L;
endfunction;


// ##################################################################
// ###                Fonctions d'affichage des courbes            ###
// ##################################################################
function Display3D_2(matrice,fenetre,titre,couleur,affichage)
[nb_f,nb_z]=size(matrice);
z=linspace(1,nb_z,nb_z);
f=linspace(1,nb_f,nb_f);
xset("window",fenetre);
xset("wpos",affichage(1),affichage(2));
xset("wdim",affichage(3),affichage(4));
xbasc()
xname(titre);
xset("colormap",couleur);
// Phase (warpped)
xsetech([0,0,1,0.5]);
plot3d1(f,z,phase(matrice(f,z)),theta=44,alpha=75,leg='Fréquence@Position@Phase',flag=[2,2,4],ebox=[1,20,1,25,-%pi,%pi]);
// Module
xsetech([0,0.5,1,0.5]);
plot3d1(f,z,abs(matrice(f,z)),theta=44,alpha=89,leg='Fréquence@Position@Module');
clear matrice; clear fenetre; clear titre; clear couleur; clear z; clear f; clear nb_f; clear nb_z;
endfunction;

function Display3D_4(m1,m2,fenetre,titre,couleur,affichage)
[nb_f,nb_z]=size(m1);
z=linspace(1,nb_z,nb_z);
f=linspace(1,nb_f,nb_f);


xset("window",fenetre);
xset("wpos",affichage(1),affichage(2));
```

```
xset("wdim",affichage(3),affichage(4));

xbasc()
xname(titre);
xset("colormap",couleur);
// m1 - Phase (warpped)
xsetech([0.01,0,0.49,0.5]);
plot3d1(f,z,phase(m1(f,z)),theta=44,alpha=75,leg='Fréquence@Position@Phase',flag=[2,2,4],ebox=[1,20,1,_nbz,-%pi,%pi]);
// m1 - Module
xsetech([0.01,0.5,0.49,0.5]);
plot3d1(f,z,abs(m1(f,z)),theta=44,alpha=89,leg='Fréquence@Position@Module',flag=[2,1,4],ebox=[1,20,1,_nbz,0,1]);
// m2 - Phase (warpped)
xsetech([0.5,0,0.49,0.5]);
plot3d1(f,z,phase(m2(f,z)),theta=44,alpha=75,leg='Fréquence@Position@Phase',flag=[2,2,4],ebox=[1,20,1,_nbz,-%pi,%pi]);
// Module
xsetech([0.5,0.5,0.49,0.5]);
plot3d1(f,z,abs(m2(f,z)),theta=44,alpha=89,leg='Fréquence@Position@Module',flag=[2,1,4],ebox=[1,20,1,_nbz,0,1]);
clear m1; clear m2; clear fenetre; clear titre; clear couleur; clear z; clear f; clear nb_f; clear nb_z;
endfunction;

function Display2D(data,fenetre,titre,affichage)
f=linspace(1,_nbf,_nbf)';
xset("window",fenetre);
xset("wpos",affichage(1),affichage(2));
xset("wdim",affichage(3),affichage(4));
xbasc();
xname(titre);
plot2d(f,data,,"021");
endfunction;


// ##################################################################
// ###                      FONCTIONS DIVERSES                      ###
// ##################################################################

// Calcul les ondes a et b avec 2 points successifs dans les mesures
function Calculate_VP_VM(my_gamma)
// clearglobal _VP,_VM;
global _VP; global _VM;
_VP=[];
_VM=[];
gammaL=my_gamma*_zstep;
for z=1:(_nbz-1),for f=1:_nbf,
  _VP(f,z)=(_M(f,z)*exp(gammaL(f))-_M(f,z+1))/(2*sinh(gammaL(f)));
  _VM(f,z)=(_M(f,z)*exp(-gammaL(f))-_M(f,z+1))/(-2*sinh(gammaL(f)));
end;end;
clear gammaL; clear z; clear f;
endfunction;

function Calculate_VP0_VM0
global _VP; global _VM;
zV1=1;

_VP=[];
_VM=[];
menu=string(linspace(1,_nbz,_nbz));
l1=list('Position de V1',1,menu);
zV1=x_choices('Emplacement des sondes...',list(l1));

// n=x_choose(...
// ['Approximation simple (retard pur)';'Approximation du NIST (permitivité complexe)';'Valeur mesurée par 3 points de
mesure'],...
// ['Valeur de la constante de propagation à utiliser';'Choisissez la valeur de gamma(f)']);
// if n==1 then Calcul_gamma_2(); my_gamma=_gamma_estimate; end;
// if n==2 then Calcul_gamma_1(); my_gamma=_gamma_estimate; end;
// if n==3 then
// Estimate_gamma();
// Measure_gamma(_gamma_estimate);
// my_gamma=_gamma_measure;
// end;
my_gamma=_gamma_estimate;
gammaL=my_gamma*_zstep;


// for z=1:(zV1-1), for f=1:_nbf,
//  D=exp(my_gamma(f).*_zstep);
//   _VP(f,z)=(_M(f,zV1).*(D^(z-zV1))-_M(f,z))/(2*sinh(gammaL(f)*(z-zV1)));
//   _VM(f,z)=(_M(f,zV1).*(D^(zV1-z))-_M(f,z))/(-2*sinh(gammaL(f)*(z-zV1)));
//end; end;
for z=1:_nbz, for f=1:_nbf,
if (z<>zV1) then,
  _VP(f,z)=(_M(f,zV1).*exp(gammaL(f)*(z-zV1))-_M(f,z))/(2*sinh(gammaL(f)*(z-zV1)));
  _VM(f,z)=(_M(f,zV1).*exp(gammaL(f)*(zV1-z))-_M(f,z))/(-2*sinh(gammaL(f)*(z-zV1)));
end;
end; end;
_VP(:,zV1)=1*10^(-40);_VM(:,zV1)=1*10^(-40);
endfunction;

function Clip(seuil)
global _VP _VM;
for i=1:_nbf, for j=1:_nbz,
if (abs(_VP(i,j))>seuil) then, _VP(i,j)=seuil*exp(%i*phase(_VP(i,j))); end;
if (abs(_VM(i,j))>seuil) then, _VM(i,j)=seuil*exp(%i*phase(_VM(i,j))); end;
end; end;
endfunction;


function Menu1

Load_File();
Display3D_2(_M,1,"Voltage Measurements",hotcolormap(32),[0,0,350,380]);

Estimate_gamma();
```

```
Measure_gamma(_gamma_estimate);
Display2D([(imag(_gamma_estimate/100)),(imag(_gamma_measure/100))],2,"Beta (Radian/cm)",[340,450,330,200]);
Display2D([(real(_gamma_estimate/100)),(real(_gamma_measure/100))],3,"Alpha (Neper/cm)", [0,450,330,200]);
Display2D([((_delay_estimate/10^(-10))),((_delay_measure/10^(-10)))],4,"Delay (ps/cm)",[680,450,330,200]);

Calculate_VP_VM(_gamma_estimate);
Display3D_4(_VP,_VM,5,"Voltage Travelling Waves",bluecolormap,[360,0,650,380]);
endfunction;


function Menu2
Estimate_gamma();
endfunction;

function Menu3
Measure_gamma(_gamma_estimate);
Display2D([(imag(_gamma_estimate/100)),(imag(_gamma_measure/100))],2,"Beta (Radian/cm)",[340,450,330,200]);
Display2D([(real(_gamma_estimate/100)),(real(_gamma_measure/100))],3,"Alpha (Neper/cm)", [0,450,330,200]);
Display2D([((_delay_estimate/10^(-10))),((_delay_measure/10^(-10)))],4,"Delay (ps/cm)",[680,450,330,200]);
endfunction;

function Menu4
Calculate_VP_VM(_gamma_estimate);
Display3D_4(_VP,_VM,5,"Voltage Travelling Waves",bluecolormap,[360,0,650,380]);
endfunction


function Menu5
Calculate_VP0_VM0();
Display3D_4(_VP,_VM,5,"Voltage Travelling Waves",bluecolormap,[360,0,650,380]);
endfunction


// ##################################################################
// ###                      PROGRAMME PRINCIPAL                   ###
// ##################################################################

// Mesures Corrigées : fenetre 1
Load_File();
Display3D_2(_M,1,"Voltage Measurements",hotcolormap(32),[0,0,350,380]);

Estimate_gamma();

Measure_gamma(_gamma_estimate);
Display2D([(imag(_gamma_estimate/100)),(imag(_gamma_measure/100))],2,"Beta (Radian/cm)",[340,450,330,200]);
Display2D([(real(_gamma_estimate/100)),(real(_gamma_measure/100))],3,"Alpha (Neper/cm)", [0,450,330,200]);
Display2D([((_delay_estimate/10^(-10))),((_delay_measure/10^(-10)))],4,"Delay (ps/cm)",[680,450,330,200]);

Calculate_VP_VM(_gamma_estimate);
Display3D_4(_VP,_VM,5,"Voltage Travelling Waves",bluecolormap,[360,0,650,380]);

// Rajout des menus pour l'utilisateur

HIP=['Menu1();','Menu2();','Menu3();','Menu4()','Menu5()'];

delmenu('HIP Process');
addmenu('HIP Process',...
['Load Measurements File','Estimate Gamma','Calculate Gamma from measurements','Voltage Travelling Waves','Voltage Travelling
Coefficients'],list(0,'HIP'));
```

Module « HIP` » sous ‘Mathematica’

Voici le code du module **HIP`** qui a été utilisé dans les exemples de ce rapport. Cela correspond à un fichier texte nommé « HIP.m » qui doit être placé dans le même repertoire que les autres modules du programme gérant le LSNA.
L’accès à ces fonctions se fait par un traditionel **OpenModule[« HIP` »]** sous ‘Mathematica’.

```
(* Module HIP developpement en cours *)


BeginPackage["HIP`"];

<< Statistics`DescriptiveStatistics`


(*
V1freqsweep::usage"V1freqsweep : renvoie V1 en fonction de la fréquence (1 à 20 GHz)";
V2freqsweep::usage"V2freqsweep : renvoie V2 en fonction de la fréquence (1 à 20 GHz)";
R1freqsweep::usage"R1freqsweep : renvoie R1 en fonction de la fréquence (1 à 20 GHz)";


MeasureRAWDATA::usage="MeasureRAWDATA[fréquence] : Mesure brute de R1 synchronisé en phase";

*)

(* Methode SIMPLE de CALIBRAGE de la sonde HIP *)

MultimeasureAll::usage="MultimeasureAll[fréquence,nombre de mesures]
Renvoie une matrice dont les colones sont {R1,R2,R3,R4} et dont chaque ligne correspond à une
mesure.
Exemple : SetSourceFreq[15*Fundamental]; NNMSProcessing`PhaseReferenceFrequency = 15; mat =
MultimeasureAll[15, 100];  u = Table[Arg[mat[[i, 1]]], {i, 1, 100, 1}];    v =
Table[Arg[mat[[i, 2]]], {i, 1, 100, 1}];  ListPlot[u, PlotJoined -> True, PlotRange -> All];
ListPlot[v, PlotJoined -> True, PlotRange -> All];";

Multimeasure::usage="Multimeasure[fréquence,nombre de mesures,quantité]";

DefineK::usage="DefineK[fréquence] : Détermine la fonction de transfert de la voie HIP lorsque
la pointe est dans le plan de référence";
DefineK1::usage="DefineK[fréquence] : Détermine la fonction de transfert de la voie 1 DCB +
HIP lorsque la pointe est dans le plan de référence";
DefineK2::usage="DefineK[fréquence] : Détermine la fonction de transfert de la voie 2 DCB +
HIP lorsque la pointe est dans le plan de référence";


MeasureHIP::usage="MeasureHIP[vecteurK , nombre de mesure] : Mesure de V HIP corrigé";

Freqsweep0::usage="Freqsweep0[harmonique max,bb mesures,quantité,puissance] Freqsweep à
puissance RF constante.";
Freqsweep::usage="Freqsweep[harmonique max,nombre de mesures,quantité] Freqsweep à puissance
RF variable - suivant le fichier de cal de puissance du système";


(* ETALONNAGE AVANCE *)

MakeSubMatrix1::usage="MakeSubMatrix1[K1,K2] : Genere la sous matrice d'étalonnage
{{K1,0},{0,K2}}, à partir des 2 vecteurs K1 et K2, afin de pouvoir creer le fichier de CAL.";

DefineGammaL1::usage="DefineGammaL1[fondamental,nbf,delay,L] : Genere un vecteur Gamma.L =
j.Beta.L = j.2*PI*f*Tau. Fondamental [Hz], Delay[ps/cm], L[mm]";
MakeSubMatrix2::usage="MakeSubMatrix2[K1,K2,gammaL,Zc] : Genere la sous matrice d'étalonnage
(2eme méthode), à partir de K1,K2,GammaL et Zc (4 vecteurs de même dim : nb d'harmoniques).";


MakeSubMatrix3::usage="MakeSubMatrix3[Matrice,CALgammaL,CALZc,MEASgammaL, MEASZc] :
Genere une sous matrice destinée à un environnement de mesure {MEASgammaL, MEASZc} à partir
d'une sous matrice d'étalonnage correspondant à l'environnement {CALgammaL, CALZc}

Il faut donc avoir préalablement identifié la sous matrice d'etalonnage, et généré les
VECTEURS
```

```
(1 point par frequence) CALgammaL, MEASgammaL, CALZc, MEASZc.

On peut dans un premier temps générer des Zc constant facilement  (fonction Table[] de
mathématica)
L'approximation la plus simple pour les vecteur GammaL est de connaitre le delay des lignes en
[ps/cm] ainsi
que la distance entre 2 sondes HIP (en mm). Les vecteurs pouront etre générés par la fonction
DefineGammaL[].";



Freqsweep2::usage="Freqsweep2[fmax_] : fonction utilisée par Cal2HIP. Renvoie une matrice V2
I2 R1 R2 mesurés simultanément.";

Cal2HIP::usage="Cal2HIP[nbf] : Etalonnage automatique à 4 termes d'erreur (Méthode 3). Cette
fonction renvoie une sous matrice utilisable
ultérieurement avec les fonctions GenerateHybridCalFile[] pour creer les fichier de calibrage
hybride du systeme total ou MakeSubMatrix3[]
pour utiliser l'etalonnage dans un autre environnement (hypothese de fonctions de tansfert en
voltage travelling waves)";

ExtractVNACalFile::usage="SubMatrix=ExtractVNACalFile[filename] : Renvoit la sous matrice
R1/R2 issue d'un étalonnage de type VNA.";



GenerateHybridVNACalFile::usage="
GenerateHybridVNACalFile[filename1,SubMatrix,filename2]:=Genere un fichier de calibrage
hybride à partir d'une sous-matrice R1/R2 et d'un fichier de calibrage standard.";

GenerateHybridCalFile::usage="GenerateHybridCalFile[filename1_,CalHIP_,filename2_] : Genere le
fichier de calibration total du LSNA à partir d'un fichier source (CAL LRRM). Les coefficients
du port 2 sont conservés. Les coefficients du port 1 sont remplacés par une sous matrice de
Calibration.
La sous matrice de calibration peut etre obtenue à partir de fonctions de transfert de la
sonde HIP (methode 1 et 2 : c.f. fonctions MakeSubMatrix1 et 2) ou par étalonnage étendu
(méthode 3).";

LRRMDebugCalFile::usage="LRRMDebugCalFile[filename1,filename2,freq,filename3]: Repare le bug
du LSNA dans l'etalonnage en puissance.
Pour cela faire 2 etalonnages LRRM. Les tester avec un Freqsweep0[]. Un 3eme fichier de cal
correspont à la cal n°1 pour 1<=f<=freq et à la cal n°2 pour f>freq.";



Begin["`Private`"]

error=Needs["Module`"];
If[(error === $Failed),Throw[False]];

(* Implementation of the module *)

error=OpenModule["General`","Spectrum`","Citifile`","NNMS`","Source`"];


(* When an error occured, an exception must be thrown to leave the routine *)
(* When multiple packages are opened, error contains a list of error messages. *)
(* Each error message is a list of 3 elements *)

If[Depth[error]>2,
  If[MemberQ[Transpose[error][[1]],False],Throw[False]],
  If[error[[1]]==False,Throw[False]]
];


(* === Les premieres fonctions basiques - INUTILES === *)
(*
V2freqsweep := Table[
      SetSourceFreq[(i*Fundamental)];
      NNMSProcessing`PhaseReferenceFrequency = i;   (* ==> Ligne 100 *)
      Pause;
      Measure;
      (v2[[i + 1]]), {i, 1, 20, 1}
      ];

V1freqsweep := Table[
```

```
      SetSourceFreq[(i*Fundamental)];
      NNMSProcessing`PhaseReferenceFrequency = i;
      Pause;
      Measure;
      (v1[[i + 1]]), {i, 1, 20, 1}
      ];

R1freqsweep := Table[
      SetSourceFreq[(i*Fundamental)];
      NNMSProcessing`PhaseReferenceFrequency = i;
      Pause;
      Measure;
      (r1[[i]]*myRefPhasor[[i + 1]]), {i, 1, 20, 1}
      ];

MeasureRAWDATA[fmax_] := Table[
      SetSourceFreq[(i*Fundamental)];
      NNMSProcessing`PhaseReferenceFrequency = i;
      Pause;
      Measure;
      (NNMS`r1[[i]]*NNMSProcessing`myRefPhasor[[i + 1]]), {i, 1, fmax, 1}];

*)

(* === Fonctions finales : nombre de mesure parametrable === *)


MultimeasureAll[freq_, n_] := Table[Pause;
      Measure;
      {NNMSProcessing`v1[[freq + 1]], NNMSProcessing`v2[[freq + 1]],
        NNMS`r1[[freq]]*NNMSProcessing`myRefPhasor[[freq + 1]],
        NNMS`r2[[freq]]*NNMSProcessing`myRefPhasor[[freq +
1]],NNMS`r3[[freq]]*NNMSProcessing`myRefPhasor[[freq + 1]],
        NNMS`r4[[freq]]*NNMSProcessing`myRefPhasor[[freq + 1]]}, {i, 1, n, 1}
];


Multimeasure[freq_,n_,mode_]:=
      Table[
      Pause;
      Measure;
      (Switch[mode,
"v1", NNMSProcessing`v1[[freq + 1]],
"v2", NNMSProcessing`v2[[freq + 1]],
"i1", NNMSProcessing`i1[[freq + 1]],
"i2", NNMSProcessing`i2[[freq + 1]],
"a1", NNMSProcessing`a1[[freq + 1]],
"a2", NNMSProcessing`a2[[freq + 1]],
"b1", NNMSProcessing`b1[[freq + 1]],
"b2", NNMSProcessing`b2[[freq + 1]],
"r1", NNMS`r1[[freq]]*NNMSProcessing`myRefPhasor[[freq +1]],
"r2", NNMS`r2[[freq]]*NNMSProcessing`myRefPhasor[[freq + 1]],
"r3", NNMS`r3[[freq]]*NNMSProcessing`myRefPhasor[[freq + 1]],
"r4", NNMS`r4[[freq]]*NNMSProcessing`myRefPhasor[[freq + 1]]
]
       ), {i, 1, n, 1}
      ];

Freqsweep[fmax_,nmax_,data_]:=Table[
      SetSourceFreq[(i*Fundamental)];
      SetSourcePower[
        Min[CalibrationMgr`RelCalPowerFunction[i*Fundamental][[1]],
          Source`MaxPowerPowerCal[i*Fundamental]]
        ];
      NNMSProcessing`PhaseReferenceFrequency=i;
      listemesures=Multimeasure[i,nmax,data];
      Extract[listemesures,Ordering[Abs[listemesures],-1]],{i,1,fmax}
      ];

Freqsweep0[fmax_,nmax_,data_,power_]:=Table[
      SetSourceFreq[(i*Fundamental)];
      SetSourcePower[power];
      NNMSProcessing`PhaseReferenceFrequency=i;
      listemesures=Multimeasure[i,nmax,data];
      Extract[listemesures,Ordering[Abs[listemesures],-1]],{i,1,fmax}
      ];
```

```
DefineK[fmax_,nb_] := Freqsweep[fmax,nb,"v2"]/Freqsweep[fmax,nb,"r1"];

MeasureHIP[vk_, nbm_] := Freqsweep[Length[vk], nbm, "r1"]*vk;

(* ======================== FONCTIONS POUR LA MATRICE DE CAL ================ *)


DefineK1[fmax_] := Table[
SetSourceFreq[(i*Fundamental)];
      SetSourcePower[
        Min[CalibrationMgr`RelCalPowerFunction[i*Fundamental][[1]],
          Source`MaxPowerPowerCal[i*Fundamental]]
       ];
      NNMSProcessing`PhaseReferenceFrequency=i;
Pause;
      Measure;
(NNMSProcessing`v2[[i + 1]]/(NNMS`r1[[i]]*NNMSProcessing`myRefPhasor[[i +1]])),
{i,fmax}];


DefineK2[fmax_] := Table[
SetSourceFreq[(i*Fundamental)];
      SetSourcePower[
        Min[CalibrationMgr`RelCalPowerFunction[i*Fundamental][[1]],
          Source`MaxPowerPowerCal[i*Fundamental]]
       ];
      NNMSProcessing`PhaseReferenceFrequency=i; (* => L 200 *)
Pause;
      Measure;
(NNMSProcessing`v2[[i + 1]]/(NNMS`r2[[i]]*NNMSProcessing`myRefPhasor[[i +1]])),
{i,fmax}];


MakeSubMatrix1[K1_,K2_]:=
Table[{{K1[[f]],0},{0,K2[[f]]}},{f,Length[K1]}
];

DefineGammaL1[fondamental_,nbf_,delay_,L_]:=
(*
Units: fondamental en Hz
nbf sans unité
delay en ps/cm
L en mm
Sortie : j*BetaL en [Rad]

gammaL = j*Beta*L = j*2*PI*f*tau
delay [ps/cm]
delay*10^(2)*10^(-12) = delay*10^(-10) = delay [s/m]
L*10^(-3) = L [m]
tau=delay*L
*)

Table[ (2*I*delay*L*(10^(-13))*(\[Pi])*fondamental*i),{i,nbf}];

MakeSubMatrix2[K1_,K2_,gammaL_,Zc_]:= Table[
{{K1[[f]],0},{((K1[[f]]*Coth[gammaL[[f]] ])/Zc[[f]]),(-K2[[f]]/(Zc[[f]]*Sinh[gammaL[[f]]
]))}},{f,Length[K1]}
];

(* ============================================ *)
MakeSubMatrix3[Mat_,CALgammaL_, CALZc_, MEASgammaL_, MEASZc]:=Module[{m1,m2,mat},

(*
 Genere une sous matrice destinée à un environnement de mesure {MEASgammaL, MEASZc} à partir
d'une sous matrice d'étalonnage correspondant à l'environnement {CALgammaL, CALZc}

Il faut donc avoir préalablement identifier la sous matrice d'etalonnage, et généré les
VECTEURS
(1 point par frequence) CALgammaL, MEASgammaL, CALZc, MEASZc.

On peut dans un premier temps générer des Zc constant facilement  (fonction Table[] de
mathématica)
L'approximation la plus simple pour les vecteur GammaL est de connaitre le delay des lignes en
[ps/cm] ainsi
```

que la distance entre 2 sondes HIP (en mm). Les vecteurs pouront etre générés par la fonction DefineGammaL[].

Renvoie la sous matrice :

```
{
   { {Alpha(fo), Beta(fo)}, {Gamma(fo) , Delta(fo)} },
  { {Alpha(2fo), Beta(2fo)}, {Gamma(2fo) , Delta(2fo)} },
      ...etc...
}
*)
SubMatrix=Table[

coefalpha=Csch[MEASgammaL[[f]]]*Sinh[CALgammaL[[f]]]*(Cosh[CALgammaL[[f]]-
MEASgammaL[[f]]]*Mat[[f,1,1]]-CALZc[[f]]*Mat[[f,2,1]]*Sinh[CALgammaL[[f]]-MEASgammaL[[f]]]);
coefbeta=Csch[MEASgammaL[[f]]]*Mat[[f,1,2]]*Sinh[CALgammaL[[f]]];

coefgamma=(1/MEASZc[[f]])*(Csch[MEASgammaL[[f]]]*Sinh[CALgammaL[[f]]]*(CALZc[[f]]*Cosh[CALgamm
aL[[f]]-MEASgammaL[[f]]]*Mat[[f,2,1]]-Mat[[f,1,1]]*Sinh[CALgammaL[[f]]-MEASgammaL[[f]]]));
coefdelta=(CALZc[[f]]/MEASZc[[f]])*Csch[MEASgammaL[[f]]]*Mat[[f,2,2]]*Sinh[CALgammaL[[f]]];
{{coefalpha,coefbeta},{coefgamma,coefdelta}},{f,fmax}]
];


(* ============================================ *)

(* ============================================================ *)
(* ==== NE MARCHE QU'AVEC 1 SEUL USER SETUP DANS LE FICHIER ==== *)
(* ============================================================ *)



GenerateHybridCalFile[filename1_,CalHIP_,filename2_]:=
Module[{stream},
(* 1- Lecture du fichier 1 => CalFile Originel *)

        stream=OpenRead[filename1];
        US=Read[stream,Expression];(* Prend le User Setup tel quel *)
CS=Read[stream,Expression];(* Idem pour le Cal Setup *)
fondamental=CS[[1,1]];(* Fondamental *)
nbf=Length[CS[[1,4]]];(* Nombre d'harmoniques *)
CC=Read[stream,Expression];(* Les Cal Coefficients *)
RR=Read[stream,Expression];(* Resistance DC *)
Close[stream];

PrintMessage["## FICHIER SOURCE ##"];
PrintMessage["fo = ",(fondamental/10^9)," GHz"];
PrintMessage["nb f = ",nbf];
PrintMessage[" "];

(* 2- Placement de la sous-matrice CalHIP dans les Cal Coeff totaux *)

nbf=Length[CC[[1]]];(* Nombre d'harmoniques *)

PrintMessage["## SOUS MATRICE ##"];
PrintMessage["nb f = ",nbf];
PrintMessage[" "];

(* Test à faire *)(* L 300 *)
Table[
CC[[1,f,1,1,1]]=CalHIP[[f,1,1]]; (* e11 *)
CC[[1,f,1,1,2]]=CalHIP[[f,1,2]]; (* e12 *)
CC[[1,f,1,2,1]]=CalHIP[[f,2,1]]; (* e21 *)
CC[[1,f,1,2,2]]=CalHIP[[f,2,2]], (* e22 *)
{f,nbf}
];


(* 3- Sauvegarde des données dans le fichier 2 *)

        stream=OpenWrite[filename2];
       Write[stream,US];
        Write[stream,CS];
        Write[stream,CC];
        Write[stream,RR];
        Close[stream];
```

```
];


LRRMDebugCalFile[filename1_,filename2_,freq_,filename3_]:=
Module[{stream,FL,vide,CC,CR},

        stream=OpenRead[filename1];
        FL=Read[stream,Expression];(* Liste des frequences *)
vide=Read[stream,Expression];(* Idem pour le Cal Setup *)
CC1=Read[stream,Expression];(* Les Cal Coefficients *)
RR=Read[stream,Expression];(* Resistance DC *)
Close[stream];

        stream=OpenRead[filename2];
        FL=Read[stream,Expression];(* Liste des frequences *)
vide=Read[stream,Expression];(* Idem pour le Cal Setup *)
CC2=Read[stream,Expression];(* Les Cal Coefficients *)
RR=Read[stream,Expression];(* Resistance DC *)
Close[stream];

Table[
CC2[[1,f]]=CC1[[1,f]],
{f,freq}
];

        stream=OpenWrite[filename3];
       Write[stream,FL];
        Write[stream,vide];
        Write[stream,CC2];
        Write[stream,RR];
        Close[stream];
];



GenerateHybridVNACalFile[filename1_,SubMatrix_,filename2_]:=
Module[{stream,FL,vide,CC,CR},
        stream=OpenRead[filename1];
        FL=Read[stream,Expression];(* Liste des frequences *)
vide=Read[stream,Expression];(* Idem pour le Cal Setup *)
CC=Read[stream,Expression];(* Les Cal Coefficients *)
RR=Read[stream,Expression];(* Resistance DC *)
Close[stream];
Table[
CC[[f,1,1]]=SubMatrxix[[f,1,1]];
CC[[f,1,2]]=SubMatrix[[f,1,2]];
CC[[f,2,1]]=SubMatrix[[f,2,1]];
CC[[f,2,2]]=SubMatrix[[f,2,2]],
{f,Length[FL]}
];
        stream=OpenWrite[filename2];
       Write[stream,FL];
        Write[stream,vide];
        Write[stream,CC];
        Write[stream,RR];
        Close[stream];
];



ExtractVNACalFile[filename_]:=
Module[{stream,FL,vide,CC,CR},
        stream=OpenRead[filename];
        FL=Read[stream,Expression];(* Liste des frequences *)
vide=Read[stream,Expression];(* Idem pour le Cal Setup *)
CC=Read[stream,Expression];(* Les Cal Coefficients *)
RR=Read[stream,Expression];(* Resistance DC *)
Close[stream];
Table[
{{CC[[f,1,1]],CC[[f,1,2]]},{CC[[f,2,1]],CC[[f,2,2]]}},{f,Length[FL]}
]
];
```

```
(* ==== FONCTIONS POUR L'ETALONNAGE DES HIPs à 4 TERMES D'ERREURS *)
(* ===== GENERE LA SOUS MATRICE {{{alpa,beta},{gamma,delta}}} ==== *)



Freqsweep2[fmax_]:=Table[

(* Renvoie la Matrice :*)
(*      V2(fo)I2(fo) R1(fo)R2(fo)*)
(*V2(2fo)I2(2fo) R1(2fo)R2(2fo)*)
(*       ... etc...*)

      SetSourceFreq[(i*Fundamental)];
      SetSourcePower[
       Min[CalibrationMgr`RelCalPowerFunction[i*Fundamental][[1]],
          Source`MaxPowerPowerCal[i*Fundamental]]
        ];
      NNMSProcessing`PhaseReferenceFrequency=i;
Pause;
Measure;
(* Contient pour chaque f les données suivantes : { V2(=V1) ; -I2(=I1) ; R1 ; R2 }      *)

{
  NNMSProcessing`v2[[i + 1]],
  NNMSProcessing`i2[[i + 1]]*(-1),
  NNMS`r1[[i]]*NNMSProcessing`myRefPhasor[[i+1]],
  NNMS`r2[[i]]*NNMSProcessing`myRefPhasor[[i+1]]
},{i,1,fmax}
      ];


Cal2HIP[fmax_]:=Module[{m1,m2,mat},

(* Renvoie la sous matrice :*)

(*{ { {Alpha(fo), Beta(fo)}, {Gamma(fo) , Delta(fo)} },*)
(*  { {Alpha(2fo), Beta(2fo)}, {Gamma(2fo) , Delta(2fo)} },*)
(*      ...etc...}*)

InputString["Placez le standard n°1 sur le port RF IN 2"];
m1=Freqsweep2[fmax];
(* m1={{v1a1,i1a1,r1a1,r2a1},{v1a2,i1a2,r1a2,r2a2}}; *)

InputString["Placez le standard n°2 sur le port RF IN 2"];
m2=Freqsweep2[fmax];
(* m2={{v1b1,i1b1,r1b1,r2b1},{v1b2,i1b2,r1b2,r2b2}}; *)

SubMatrix=Table[
coefalpha=( (m2[[f,4]]*m1[[f,1]])-(m1[[f,4]]*m2[[f,1]]) ) / ( (m1[[f,3]]*m2[[f,4]])-
(m2[[f,3]]*m1[[f,4]]) );
coefbeta=( (m2[[f,3]]*m1[[f,1]])-(m1[[f,3]]*m2[[f,1]]) ) / ( (m2[[f,3]]*m1[[f,4]])-
(m1[[f,3]]*m2[[f,4]]) );
coefgamma=( (m2[[f,4]]*m1[[f,2]])-(m1[[f,4]]*m2[[f,2]]) ) / ( (m1[[f,3]]*m2[[f,4]])-
(m2[[f,3]]*m1[[f,4]]) );
coefdelta=( (m2[[f,3]]*m1[[f,2]])-(m1[[f,3]]*m2[[f,2]]) ) / ( (m2[[f,3]]*m1[[f,4]])-
(m1[[f,3]]*m2[[f,4]]) );
{{coefalpha,coefbeta},{coefgamma,coefdelta}},{f,fmax}]
];


(*

Les fonctions qui suivent sont des expérimentations.
Elles n'ont pas été achevées et ne sont pas disponibles par l'utilisateur
lambda lors de l'ouverture du module HIP`

Identification Methode 3 => Extraction des GAMMA et Zc pour la Methode 2
                            Ne fonctionne pas.

*)

(*
ExtractLineParameters[SubMatrix_,K1_,K2_]:=
Module[{stream,FL,vide,CC,CR},
```

```
        (* Solution à vérifier *)

GLZc=Table[
X=(-K2[[f]]/SubMatrix[[f,2,2]])*(SubMatrix[[f,2,1]]/K1[[f]]);
GammaL=ArcCosh[X];
Zc=(-K2[[f]]/SubMatrix[[f,2,2]])*(1/(Sqrt[X-1]*Sqrt[X+1]));
{GammaL,Zc},
{f,Length[K1]}
];

Transpose[GLZc]
(* Renvoie les vecteurs {GammaL(f)},{Zc(f)} *)
];
*)

ExtractLineParameters[SubMatrix_,K1_,K2_]:=
Module[{stream,FL,vide,CC,CR},
        (* Solution à vérifier Impose Zc=50 *)

GLZc=Table[
X=(-K2[[f]]/SubMatrix[[f,2,2]])*(SubMatrix[[f,2,1]]/K1[[f]]);
GammaL=ArcCosh[X];
Zc=(-K2[[f]]/SubMatrix[[f,2,2]])*(1/(Sqrt[X-1]*Sqrt[X+1]));
{GammaL,Zc},
{f,Length[K1]}
];

Transpose[GLZc]
(* Renvoie les vecteurs {GammaL(f)},{Zc(f)} *)
];



End[];

EndPackage[];
```

Album de photos-souvenirs

Voici quelques photos du système de mesure. Elles ont été prises lors de la campagne de mesures sur l'amplificateur d'Alain Mallet optimisé en classe F. Les mesures sont faites ici simultanément sur l'entrée et la sortie d'un même transistor et la sortie du circuit MMIC.

Fig. 53 - Banc de mesure 'Cascade'. A gauche, la source extérieure Anritsu utilisée et le PC controllant la table sous pointe. A droite, le LSNA.

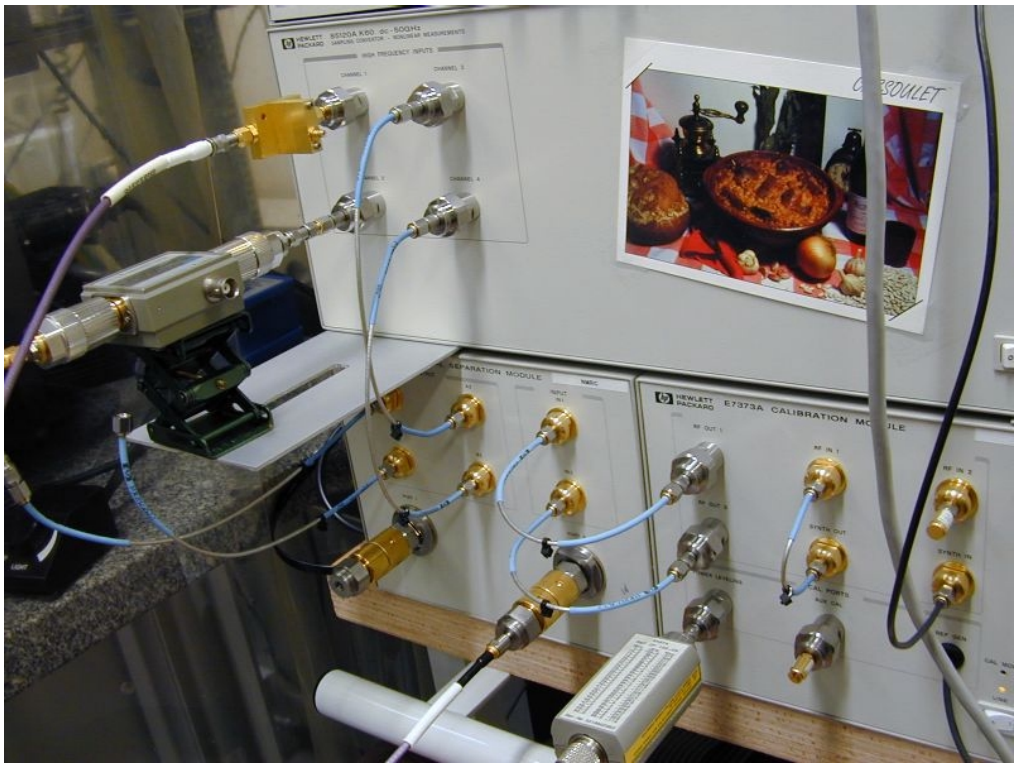


Fig. 54 - Zoom sur le LSNA. Le port 1 est déconnecté. Les entrées 1 et 2 du 'Downconverter Box' sont reliées aux sondes HIP via des tés de polarisation jouant le rôle de 'DC Block'.

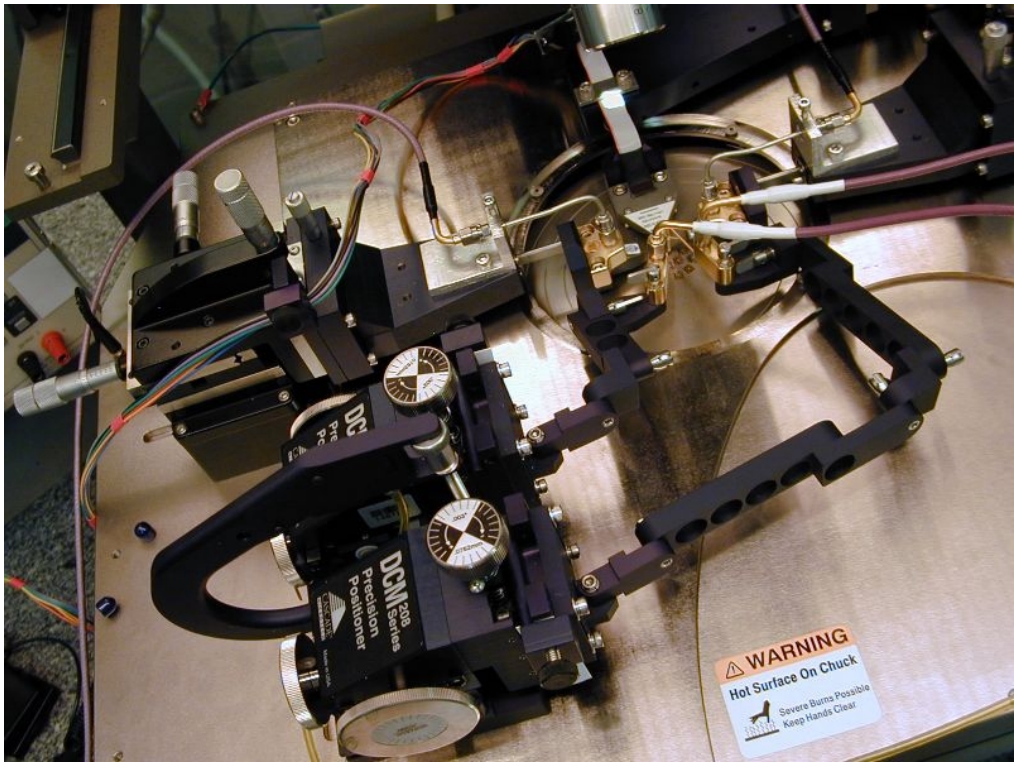


Fig. 55 - Le positionneur des sondes HIP placé sur la table sous pointes.

Fig. 56 - Zoom sur les sondes HIP : ces dernières sont posées sur le circuit, tout comme les sondes GSG et la pointe de polarisation.

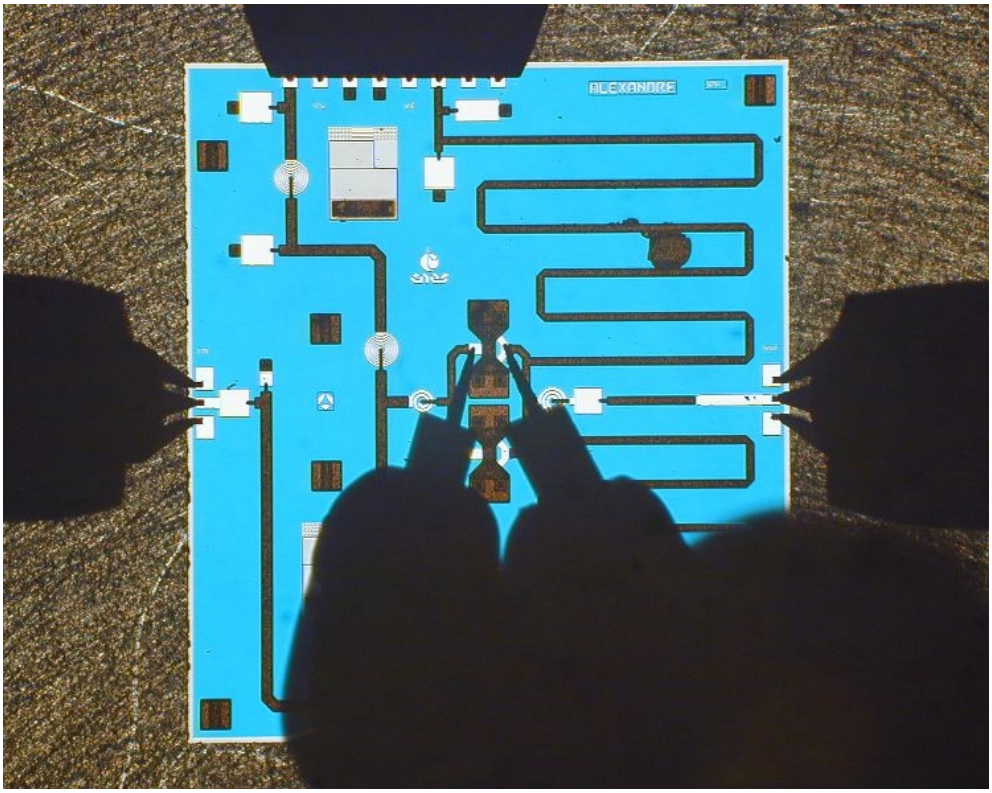


Fig. 57 - Vue d'ensemble du circuit analysé. Les sondes HIP sont posées simultanément en entrée et en sortie du transistor du haut.

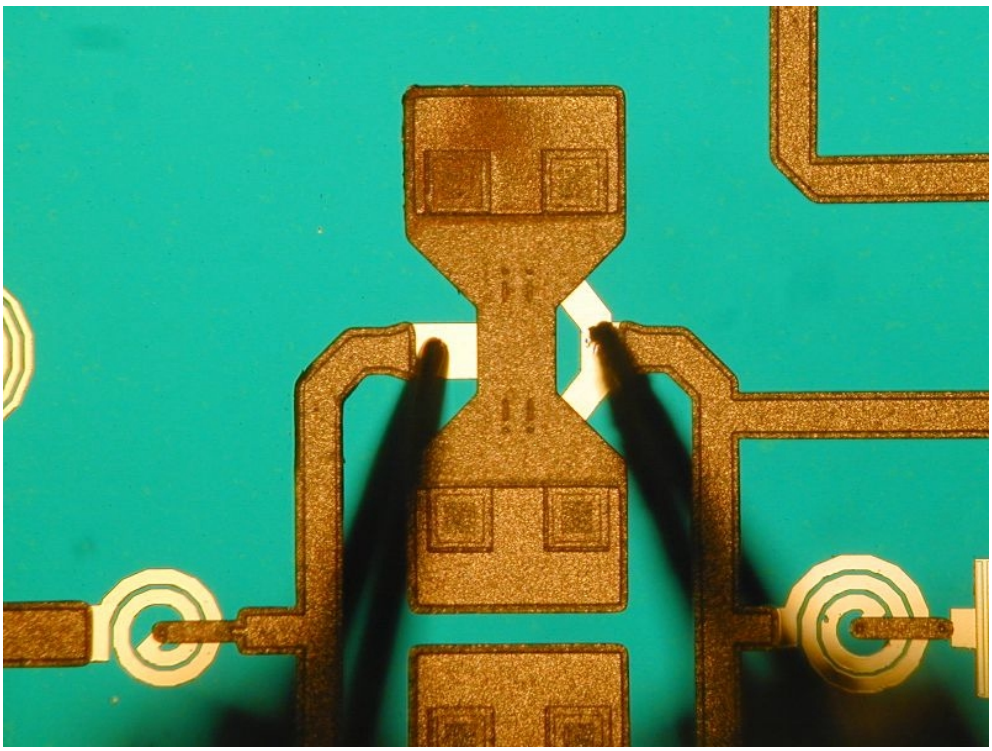


Fig. 58 - Zoom sur l'emplacement des sondes HIP