

Agilent EEsof EDA

De-Embedding Series-Connected / Transmission Configured Devices

This document is owned by Agilent Technologies, but is no longer kept current and may contain obsolete or inaccurate references. We regret any inconvenience this may cause. For the latest information on Agilent's line of EEsof electronic design automation (EDA) products and services, please go to:

www.agilent.com/find/eesof



De-Embedding Series-Connected / Transmission Configured Devices

Michael D. Brunsman

Characterization and Modeling Group CMOS Platform Device Development Digital DNA Laboratory Motorola Semiconductor Products Sector

> 3501 Ed Bluestein Blvd. Austin, Tx. 78721

1.0 ABSTRACT

A new, measurement-based method for de-embedding the electrical characteristics of series-connected / transmission-configured devices has been developed. Previously utilized on-wafer de-embedding schemes which involved Y-parameter subtraction of 'open' characteristics followed by Z-parameter subtraction of 'short' characteristics have been replaced by a single 'thru' measurement based approach. The procedure is unique in that it incorporates a "splitting" algorithm which partitions the 2-port characteristics of a single 'thru' structure into left and right error adapters which preserve the electrical characteristics seen at the input and output of the 'thru', and creates an ideal "virtual interface" midway along the length of the 'thru'. The transmission characteristics of the left and right networks are such that when cascaded, they replicate the electrical characteristics of the 'thru'. Validation of both the "splitting" algorithm and the de-embedding process has been completed in ADS via the use of cascaded and reciprocal 2-port networks. This paper presents these results. The complete de-embedding methodology has been successfully implemented in IC-CAP for device characterization and modeling purposes.

2.0 ADS ANALYSIS CONFIGURATIONS

The following networks were configured in ADS to facilitate analysis of the de-embedding characteristics.



Figure 1. Raw, Measured 'Thru' S-Parameters



Figure 2. Cascaded Input & Output 2-Port Networks

Motorola General Business Information



Figure 3. Input 2-Port, Thru, Output 2-Port ADS De-embedding



Figure 4. Input 2-Port, Thru, Output 2-Port IC-CAP De-embedded S-Parameters



Figure 5. SOLT Calibration Thru S-Parameters (Post Open-short De-Embedding)

8/21/03

3.0 ANALYSIS DATA

3.1 Verification of SOLT Calibration

3.1.1 'OPEN' Standard Capacitance

A plot of the equivalent capacitance associated with ports-1 and 2 of the SOLT 'open' standard is shown in Figure 6 below. The capacitance is "extracted" directly from the imaginary part of the measured S-Parameters after S-to-Y conversion.



Figure 6. ISS 'OPEN' Port-1 / Port-2 Equivalent Capacitance Characteristics

<u>NOTE</u>: The value of the 'OPEN' capacitance entered into the WinCal software for the I40 GSG 150uM pitch probes is –6.7fF

3.1.2 'SHORT' Standard Inductance

A plot of the equivalent inductance associated with ports-1 and 2 of the SOLT 'short' standard is shown in Figure 7 below. The inductance is "extracted" directly from the imaginary part of the measured S-Parameters after S-to-Z conversion, following de-embedding with the 'open' S-Parameters.



Figure 7. ISS 'Short' Port-1 / Port-2 Equivalent Inductance Characteristics

NOTE: The value of the 'SHORT' inductance entered into the WinCal software for the I40 GSG 150uM pitch probes is 8.2phy.

3.1.3 'LOAD' Standard Resistance

A plot of the equivalent resistance associated with ports-1 and 2 of an SOLT precision (laser trimmed) 'load' standard is shown in Figure 8 below. The resistance is "extracted" directly from the real part of the measured S-Parameters after S-to-Z conversion, following de-embedding with the 'open' S-Parameters.



Figure 8. ISS 'LOAD' Port-1 / Port-2 Equivalent Resistance Characteristics

<u>NOTE</u>: The value of the 'LOAD' resistance entered into the WinCal software for the SOLT calibration is 50 Ohms.

3.1.4 'THRU' Standard Delay

A plot of the forward and reverse delay associated with the SOLT 'thru' standard is shown in Figure 9 below. The delay is "extracted" directly from the phase of the measured S-Parameters after S-to-Z conversion, following de-embedding with both an 'open' and 'short' S-Parameters.



Figure 9. ISS 'THRU' Delay Characteristics

<u>NOTE</u>: The value of the 'THRU' delay entered into the WinCal software for the I40 GSG 150uM pitch probes is -1psec.

7

3.2 ADS Verification of 'Thru' Characteristics

The following "legend" defines the labeling convention for the S-parameter data sets in ADS:

- S11, S22 \rightarrow Raw 'thru' measured S-Parameters (post SOLT calibration)
- S33, S44 → Cascaded Left / Right 2-port S-Parameters
- S55, S66 → ADS De-Embedded 'thru' S-Parameters (using Left / Right 2-Port network S-parameters as deembedding elements)
- S77, S88 → IC-CAP De-Embedded 'thru' S-Parameters (using Left / Right 2-port network S-parameters and ABCD cascading)
- S99, S1010 \rightarrow SOLT calibration 'thru' (1psec transmission delay) standard S-Parameters

3.2.1 Reflection Characteristics of 'thru' and Cascaded Left / Right 2-port Networks:

The following data shows the reflection characteristics of the 'thru' (as measured after SOLT calibration) compared to what results when the left and right 2-port "splitting" networks are cascaded. As expected, the input and output S-Parameters of the resulting cascaded 'thru' are preserved by the "splitting" algorithm.





Figure 12. S11 Magnitude Data

Figure 13. S11 Phase Data



Figure 14. S22 Magnitude Data



3.2.2 Transmission Characteristics of 'thru' and Cascaded Left / Right 2-port Networks:

The following data shows the transmission characteristics of the 'thru' (as measured after SOLT calibration) compared to what results if the left and right 2-port "splitting" networks are cascaded. As expected, the transmission S-Parameters (thru loss and phase shift) of the resulting cascaded 'thru' are preserved by the "splitting" algorithm.







Figure 18. S21 Magnitude Data

0.0

-0.5

-1.0

-1.5

-2.0

-2.5 -

-3.0

ΰ

2

Δ

dB(S(3,4)) dB(S(1,2))



freq (100.0MHz to 10.10GHz)

Figure 17. S12 Data



Figure 19. S21 Phase Data



Figure 20. S12 Magnitude Data

6

freq, GHz

10

12

8

Figure 21. S12 Phase Data

Motorola General Business Information

3.3 Verification of 'Thru' De-Embedding

3.3.1 Reflection Characteristics of ADS and IC-CAP De-Embedded 'thru' and SOLT Cal. 'thru':

The following data shows the reflection characteristics of the 'thru', de-embedded using the reciprocal deembedding networks in ADS and the cascaded ABCD matrices as implemented in IC-CAP, compared to the SOLT calibration 'thru' (1psec transmission delay) standard. As expected, the input and output S-Parameters of the deembedded 'thru' look like an ideal 50 ohm system, with port match characteristics exceeding those attained via the SOLT calibration.



Figure 24. S11 Magnitude Data

Figure 25. S11 Phase Data



Figure 26. S22 Magnitude Data



3.3.2 Reflection Characteristics of ADS and IC-CAP De-Embedded 'thru' and SOLT Cal. 'thru':

The following data shows the transmission characteristics of the 'thru', de-embedded using the reciprocal deembedding networks in ADS and the cascaded ABCD matrices as implemented in IC-CAP compared to the SOLT calibration 'thru' (1psec transmission delay) standard. As expected, the transmission S-Parameters of the ADS and IC-CAP de-embedded 'thru' show zero insertion loss and zero electrical length (residual phase shift). The characteristics of the SOLT data reflects a zero insertion loss and phase shift associated with the 1ps delay line.





Figure 30. S21 Magnitude Data



Figure 32. S12 Magnitude Data



Figure 31. S21 Phase Data



Figure 33. S12 Phase Data

4.0 COLCLUSIONS

The various analyses configured in ADS were structured in order to validate not only the de-embedding technique, but also the algorithms used to generate the input and output 2-port networks as well as the algorithm used to de-embed measured data using the left and right 2-port networks. Based on the data presented, it is clear that the goals of the new de-embedding technique have been met. In summary these were:

- To preserve the input (S11) and output (S22) characteristics of the 'thru'.
- To create an ideal, reflectionless, "virtual interface" midway along the length of the 'thru'.
- To force the transmission characteristics of the input and output error adapters such that when cascaded, they replicate the insertion loss and phase of the 'thru'.

In application, the new de-embedding technique has been successfully applied to the characterization and modeling of small geometry 2-terminal, series-connected devices fabricated on Silicon substrates. It has been found that the application of this technique has resulted in a significant reduction of test time, an increase in measurement accuracy and repeatability, and the ability to quickly determine intrinsic device characteristics via direct extraction. The implications are that the technique can be applied to facilitate the generation of statistical data on in-process RF test structures based on direct extraction without the need for additional post processing.

Appendix A.

Input Splitting Algorithm Transform

```
LINK XFORM "Input_2_port"
View block_edit 0 "0,108"
data
HYPTABLE "Link Transform"
{
element "Function" "Program"
BLKEDIT "Program Body"
ł
UPDATE_MANUAL
!
thru=s.m
size_thru=SIZE(thru)
!
COMPLEX In_2_port.22[SIZE(thru)]
COMPLEX Mag_In_2_port.22[SIZE(thru)]
COMPLEX Angle_In_2_port.22[SIZE(thru)]
COMPLEX Real_In_2_port.22[SIZE(thru)]
COMPLEX Imag_In_2_port.22[SIZE(thru)]
!
 ! Write split value to setup variables for Output Partitioning
1
LINPUT "Enter % of split to input 2-port: [ 0 = 50%, 1 = 30%, 2 = 25%, 3 = 20%", split, split
1
if split == 0 THEN ! 50% split of Input / Ouptut 2-ports
   var1=2
   var2=0.5
end if
1
if split == 1 THEN ! 30% split of Input / Ouptut 2-ports
   var1=3
   var2=0.3333
end if
!
if split == 2 THEN ! 25% split of Input / Ouptut 2-ports
   var1=4
   var2=0.25
end if
if split == 3 THEN ! 20% split of Input / Ouptut 2-ports
   var1=5
   var2=0.20
end if
!
1
i1=0
i=0
WHILE i < SIZE(thru)
  In_2_port.11[i]=thru.11[i1]
  !
     Mag_In_2_port.12[i]=(((real(thru.12[i1])^var1)+(imag(thru.12[i1])^var1))^var2)^var2
```

}

```
Angle_In_2_port.12[i]=atn(imag(thru.12[i1])//(real(thru.12[i1])))//var1
     Real In 2 port.12[i]=cos(Angle In 2 port.12[i])*Mag In 2 port.12[i]
     Imag_In_2_port.12[i]=sin(Angle_In_2_port.12[i])*Mag_In_2_port.12[i]
  In_2_port.12[i]=(Real_In_2_port.12[i])+j*(Imag_In_2_port.12[i])
  !
     Mag_In_2_port.21[i]=(((real(thru.21[i1])^var1)+(imag(thru.21[i1])^var1))^var2)^var2
     Angle_In_2_port.21[i]=atn(imag(thru.21[i1])//(real(thru.21[i1])))//var1
     Real_In_2_port.21[i]=cos(Angle_In_2_port.21[i])*Mag_In_2_port.21[i]
     Imag_In_2_port.21[i]=sin(Angle_In_2_port.21[i])*Mag_In_2_port.21[i]
  In_2_port.21[i]=(Real_In_2_port.21[i])+j*(Imag_In_2_port.21[i])
  L
  In_2_port.22[i]=0+j0
  i1=i1+1
  IF il>=size_thru THEN il=0
  i = i + 1
END WHILE
!
RETURN In_2_port
```

Appendix B.

Output Splitting Algorithm Transform

```
LINK XFORM "Output_2_port"
{
View block edit 0 "0,0"
data
{
HYPTABLE "Link Transform"
{
element "Function" "Program"
BLKEDIT "Program Body"
UPDATE MANUAL
PRINT "running 'deemb'"
1
thru=s.m
!
size_thru=SIZE(thru)
1
COMPLEX Out_2_port.22[SIZE(thru)]
COMPLEX Mag Out 2 port.22[SIZE(thru)]
COMPLEX Angle_Out_2_port.22[SIZE(thru)]
COMPLEX Real_Out_2_port.22[SIZE(thru)]
COMPLEX Imag Out 2 port.22[SIZE(thru)]
1
! Write split value to setup variables for Output Partitioning
!
LINPUT "Enter % of split to input 2-port: [ 0 = 50%, 1 = 30%, 2 = 25%, 3 = 20%", split, split
1
if split == 0 THEN ! 50% split of Input / Ouptut 2-ports
   var1=2
   var2=0.5
end if
!
if split == 1 THEN ! 30% split of Input / Ouptut 2-ports
   var1=(3/2)
   var2=(2/3)
end if
1
if split == 2 THEN ! 25% split of Input / Ouptut 2-ports
   var1 = (4/3)
   var2=(3/4)
end if
1
if split == 3 THEN ! 20% split of Input / Ouptut 2-ports
   var1 = (5/4)
   var2=(4/5)
end if
!
i1=0
i=0
WHILE i < SIZE(thru)
  Out_2_port.11[i]=0+j0
  I.
     Mag_Out_2_port.12[i]=(((real(thru.12[i1])^var1)+(imag(thru.12[i1])^var1))^var2)^var2
```

```
Angle_Out_2_port.12[i]=atn(imag(thru.12[i1])//(real(thru.12[i1])))//var1
     Real Out 2 port.12[i]=cos(Angle Out 2 port.12[i])*Mag Out 2 port.12[i]
      Imag_Out_2_port.12[i]=sin(Angle_Out_2_port.12[i])*Mag_Out_2_port.12[i]
   Out_2_port.12[i]=(Real_Out_2_port.12[i])+j*(Imag_Out_2_port.12[i])
   !
     Mag_Out_2_port.21[i]=(((real(thru.21[i1])^var1)+(imag(thru.21[i1])^var1))^var2)^var2
     Angle_Out_2_port.21[i]=atn(imag(thru.21[i1])//(real(thru.21[i1])))//var1
     Real_Out_2_port.21[i]=cos(Angle_Out_2_port.21[i])*Mag_Out_2_port.21[i]
      Imag_Out_2_port.21[i]=sin(Angle_Out_2_port.21[i])*Mag_Out_2_port.21[i]
   Out_2_port.21[i]=(Real_Out_2_port.21[i])+j*(Imag_Out_2_port.21[i])
   Out_2_port.22[i]=thru.22[i1]
   i1=i1+1
   IF il>=size_thru THEN il=0
   i = i + 1
 END WHILE
 !
RETURN Out_2_port
}
dataset
{
```

Appendix C.

Left – Right Network Cascading Transform

```
LINK XFORM "Cascade_ABCD"
View block_edit 0 "0,0"
data
ł
HYPTABLE "Link Transform"
{
element "Function" "Program"
}
BLKEDIT "Program Body"
{
L_dummy = Input_2_port
 !
R_dummy = Output_2_port
 1
xtor=TwoPort(L_dummy, "S", "A") * TwoPort(R_dummy, "S", "A")
xtor=TwoPort(xtor, "A", "S")
 1
RETURN xtor
}
dataset
```

Appendix D.

LINK XFORM "S_dmbd_ABCD"

De-Embedding Transform

```
View block_edit 0 "0,0"
data
HYPTABLE "Link Transform"
{
element "Function" "Program"
BLKEDIT "Program Body"
ł
 UPDATE_MANUAL
 !
 total = s.m
                ! 'thru' data
 1
 L_dummy = Input_2_port
 R_dummy = Output_2_port
 size_dummy=SIZE(L_dummy)
 !
 PRINT "duplicate open dummy data array to match the actual data size..."
 COMPLEX L_dummy_act.22[SIZE(total)]
 COMPLEX R_dummy_act.22[SIZE(total)]
 !
 i1=0
 i=0
 WHILE i < SIZE(total)
   L_dummy_act.11[i]=L_dummy.11[i1]
   L_dummy_act.12[i]=L_dummy.12[i1]
  L_dummy_act.21[i]=L_dummy.21[i1]
   L_dummy_act.22[i]=L_dummy.22[i1]
   R_dummy_act.11[i]=R_dummy.11[i1]
   R_dummy_act.12[i]=R_dummy.12[i1]
   R_dummy_act.21[i]=R_dummy.21[i1]
   R_dummy_act.22[i]=R_dummy.22[i1]
   i1=i1+1
   IF il>=size dummy THEN il=0
   i = i + 1
 END WHILE
 1
xtor=TwoPort(L_dummy_act, "S", "A")^-1 * TwoPort(total, "S", "A") *
TwoPort(R_dummy_act, "S", "A")^-1
 !
 xtor=TwoPort(xtor, "A", "S")
 1
RETURN xtor
```

For more information about Agilent EEsof EDA, visit:

www.agilent.com/find/eesof



www.agilent.com/find/emailupdates Get the latest information on the products and applications you select.



www.agilent.com/find/agilentdirect Quickly choose and use your test equipment solutions with confidence.

www.agilent.com

For more information on Agilent Technologies' products, applications or services, please contact your local Agilent office. The complete list is available at:

www.agilent.com/find/contactus

(877) 894-4414
305 269 7500
(800) 829-4444

Asia Pacific	
Australia	1 800 629 485
China	800 810 0189
Hong Kong	800 938 693
India	1 800 112 929
Japan	0120 (421) 345
Korea	080 769 0800
Malaysia	1 800 888 848
Singapore	1 800 375 8100
Taiwan	0800 047 866
Thailand	1 800 226 008

Europe & Middle East

Austria	0820 87 44 11	
Belgium	32 (0) 2 404 93 40	
Denmark	45 70 13 15 15	
Finland	358 (0) 10 855 2100	
France	0825 010 700*	
	*0.125 €/minute	
Germany	01805 24 6333**	
	**0.14 €/minute	
Ireland	1890 924 204	
Israel	972-3-9288-504/544	
Italy	39 02 92 60 8484	
Netherlands	31 (0) 20 547 2111	
Spain	34 (91) 631 3300	
Sweden	0200-88 22 55	
Switzerland	0800 80 53 53	
United Kingdom	44 (0) 118 9276201	
Other European Countries:		
www.agilent.com/find/contactus		
Revised: March 27, 2008		

Product specifications and descriptions in this document subject to change without notice.

© Agilent Technologies, Inc. 2008 Printed in USA, August 21, 2003 5989-9466EN

