



## SHAKER, SMASH and GDS Strata for migrating a RAM from one 0,8 $\mu\text{m}$ to a 0,5 $\mu\text{m}$ technological process

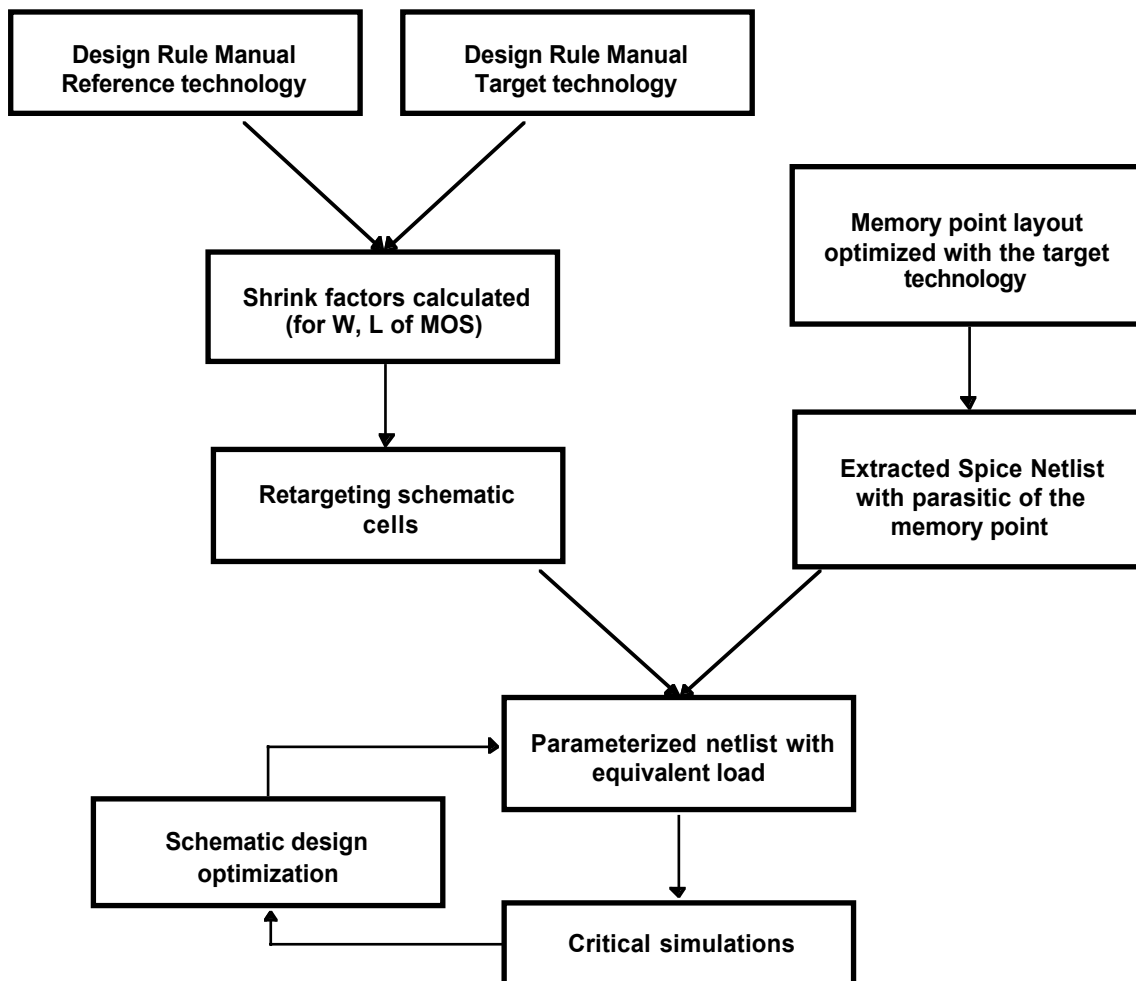
### INTRODUCTION

The purpose of this application note is to show how to proceed for migrating a RAM from one technological process to another. The whole methodology is explained in the Retargeting Process Sheet. This methodology is applicable for whatever circuitware, analog or mixed-signal.

The goal of such a migration is to keep the same electrical functionalities while optimizing the layout surface. This leads to an effective cost reduction through a double gain of time and surface.

### PART 1 - CAE Steps on Schematics

Flowchart below shows the whole electrical characterization process to follow. It is then explained in details showing how DOLPHIN tools interact.







```
XVG SA0 DC NSA0 PCH D0 ND0 NPCH RADEXVG0
XNAND SA10 SEL RANAND{NAND}
CR READY VSS {COUT}
CS DO VSS {COUT}
XCPC LM0 LM1 LMB1 NSA10 NSMUX PCH SA10 SECR SLEC SMUX STS
XCPL SA0 SA1 SA2 SA3 SA4 SA5 SA6 SA7 SA8 NSA0 NSA1 NSA2 NSA3
+ NSA4 NSA5 NSA6 NSA7 NSA8 PCH B1 NB2 NB1 B2 CAPALIGN
XDL A10 CS NSA10 SA10 RAALDMD0{SCOL}
XBTS SE NE SEL STS DO RABTS2A1
XLEC B NB SLEC NE NSI SE SI RAALOC20
XC A0 A1 A2 A3 A4 A5 A6 A7 A8 CS EMUX NRST NSI RD SI TS WR
+ LMB1 NSA0 NSA1 NSA2 NSA3 NSA4 NSA5 NSA6 NSA7 NSA8 NSMUX PCH READY
+ SA0 SA1 SA2 SA3 SA4 SA5 SA6 SA7 SA8 SECR SLEC SMUX STS RAINBC{BC}
XH0 PCH B1 NB1 RAPRECH{SLI}
XH1 PCH B2 NB2 RAPRECH{SLI}
XDM1 ND0 NPCH LM1 RAAMLIM{SCOL}
XDM0 D0 NPCH LM0 RAAMLIM{SCOL}
XMUX SMUX B B1 B2 NSMUX NB NB1 NB2 RAMUX21{SLI}
XDEX NSA1 NSA2 NSA3 NSA4 NSA5 NSA6 NSA7 NSA8 DC RADX{SLI}
XM LM0 LM1 B1 B2 NB1 NB2 RA4PTMEM
XB LMB1 RA2PTMEM
XEC DI SECR SEL B NB RAAECR2{SLI}
```

• Command file

Containing the control variables to iterate (for instance line number, column number, ...), the simulations to run (temperature, Alimentation, ...), the characteristics to extract (access time, propagation time, ...) and the datasheet to write (see Ex. 1.3).

Ex. 1.3 - Command file

**Parameters to substitute and simulations to run**

\* Wed Sep 24 18:12:11 METDST 1997

```
.DEFAULT 1
*SELECTION priority
TITLE SMASH simulation RAM00
SUBCKT rabts2a1 ra4ptmem ra2ptmeb
radexvg0 capalign capacol raaloc20
radmop01 radmco01 radmco01 raamlim0 raaldmo0
raprech0 ramux210 raaecr20 radex00 radx0
rainbc0 rainbc0 ranand2
MODEL TYP
PARAMETER
VA = 5.0
VTN = 0.7
VTP = 0.8
TEMP = 25
X = 2
Y = 10
NCOL = 8
NLIG = 8
SCOL = 0
SLI = 0
BC = 0
NAND = 2
COUT = 1p

.SIMULATION 1
DEFAULT 1
FROMREF ram TOSIMUL RAM00 OPTION TRAN
TITLE Ram, 8lines*8columns, 25deg, TYP, 5.0v
```

**Extraction instructions**

```
.EXTRACTION
*****
* - E1 - *
* Access time extraction *
* All times are measured at 50% of total swing *
*****

* 0 read access time from CS rising edge
TP_H L=TFDO FROM NODE=CS
TO NODE=DO
AFTER={FALLINGEDGE=2 NODE=WR}

* 1 read access time from CS rising edge
TP_H H=TRDO FROM NODE=CS
TO NODE=DO
AFTER={FALLINGEDGE=6 NODE=CS}
```



```

*****
*                               - E20 -                               *
* Equivalent voltage generator current integration                       *
* Input capacitance measurement                                       *
*****

* load on RD
INTEGR=QRD NODE=(VRD
AFTER=(FALLINGEDGE=5 NODE=CS)
BEFORE=(RISINGEDGE=6 NODE=CS)

* load on A10 column address
INTEGR=QA10 NODE=(VA10
AFTER=(FALLINGEDGE=2 NODE=EMUX)
BEFORE=(RISINGEDGE=4 NODE=CS)

* load on A0 line address
INTEGR=QA0 NODE=(VA0
AFTER=(FALLINGEDGE=3 NODE=CS)
BEFORE=(FALLINGEDGE=2 NODE=EMUX)

```

### Results Calculation

```

.RESULTS

*****
*           ACCESS TIME: TACCES           *
* Maximum of 0 or 1 read access times    *
*****
Tacces_1=MAX(TFDO_1,TRDO_1)

*****
*           HOLD TIME: THRDWR, THAD      *
*****
* calculation of addresses hold time
THAD_1=MAX(THAdLigR_1,THAdLigF_1,THAdCoIF_1)

```

### Datasheet Generation

```

.DATASHEET
CAPA={COU_1}
NLIG={NLIG_1}
NCOL={NCOL_1}
VA={VA_1}
TEMP={TEMP_1}

-----
access times from CS :      ACCESS_TIME      = {Tacces_1} s
-----
propagation time between CS and READY rising edge :TP_CS_READY  = {TRREADY_1} s
propagation time between CS and READY falling edge:
- READ :      READY_TIME_LEC      = {TFREADY_L_1} s
- WRITE :     READY_TIME_ECR      = {TFREADY_E_1} s
-----

```

#### • Library file

Containing the path for finding the electrical models to use, subcircuits, simulator to use.

#### • Subcircuit files

Describing the hierarchy of the IC if any.

#### • Models file

Containing all electrical models to use.

#### • Simulation pattern file

Defining the stimuli signals for all simulations.

```

*****
*           ACTIVATION           *
*****

global vdd vss
VDD VDD 0 DC {VA}
VSS VSS 0 DC 0

*****
VCS CS 0 PWL ON 0 {2*Y}N 0 {2*Y+X}N {VA} {4*Y}N {VA} {4*Y+X}N 0
+ {6*Y}N 0 {6*Y+X}N {VA} {8*Y}N {VA} {8*Y+X}N 0
+ {10*Y}N 0 {10*Y+X}N {VA} {12*Y}N {VA} {12*Y+X}N 0
+ {18*Y}N 0 {18*Y+X}N {VA} {20*Y}N {VA} {20*Y+X}N 0
+ {22*Y}N 0 {22*Y+X}N {VA} {24*Y}N {VA} {24*Y+X}N 0
+ {26*Y}N 0 {26*Y+X}N {VA} {28*Y}N {VA} {28*Y+X}N 0
+ {30*Y}N 0 {30*Y+X}N {VA} {33*Y}N {VA} {33*Y+X}N 0
+ {35*Y}N 0 {35*Y+X}N {VA} {37*Y}N {VA} {37*Y+X}N 0
*****
VNRST NRST 0 PWL ON {VA} {14*Y}N {VA} {14*Y+X}N 0 {16*Y}N 0 {16*Y+X}N {VA}
*****
VA0 A0 0 PWL ON 0 {14*Y}N 0 {14*Y+X}N {VA} {16*Y}N {VA} {16*Y+X}N 0
VA1 A1 0 PWL ON 0 {14*Y}N 0 {14*Y+X}N {VA} {16*Y}N {VA} {16*Y+X}N 0
*****

VDIN DI 0 PWL ON {VA} {5*Y}N {VA} {5*Y+X}N 0 {21*Y}N 0 {21*Y+X}N {VA}

*****
.PRINT V(VCSH) V(XC.XC.H) V(XC.XC.NH) V(XC.XC.BK1L) V(XC.XC.BK1E)
*****
.TRACE TRAN V(NRST) V(CS)
*****
.TEMP {TEMP}
.TRAN 0.1N {40*Y}N

.CREATEICDFILE
.PUREANALOG
.OP EPS_V=1u VMIN=0 VMAX={VA} DELTAV=-1 EPS_I=100p MAXITER=500 BIASINFO=NONE

```



.METHOD TRAP  
EPS 1u 100m 1n  
H 1p 1f 50p 250m 2  
\*.OPHELP GDSMOS 1e-6 1e-12 1e-3

**SHAKER generates different output files:**

- Each ending phase Execution message file.
- Ending the Substitution phase Simulation files to submit and Information file.
- Ending the Submission phase Output files from simulator.
- Ending the Extraction phase File displaying the list of extracted data (see Ex. 1.4).
- Ending the Results phase Datasheet presenting the calculated values.

Ex. 1.4 - Extracted data

```
***** extraction file for RAM00.shk *****
TFDO_1 = 7.88294340E-09
TRDO_1 = 8.16400236E-09
FDO_1 = 9.07025415E-10
RDO_1 = 1.45415932E-09
TRREADYL_1 = 2.90852840E-09
TRREADYE_1 = 2.91994876E-09
TFREADY_L_1 = 9.46740029E-09
TFREADY_E_1 = 6.22539019E-09
RREADY_1 = 1.33095786E-09
FREADY_1 = 9.42126827E-10
THADLIGR_1 = 3.19314562E-09
THADLIGF_1 = 3.40922279E-09
THADCOLF_1 = 1.26270254E-09
VDDREADY_1 = -2.94951323E-11
VSSREADY_1 = 2.66873328E-11
VDDMUX_1 = -1.27747212E-12
VSSMUX_1 = 1.13104460E-12
VDDECR_1 = -4.88195792E-12
VDDSOUT_1 = -6.66498801E-12
VSSSOUT_1 = 5.61365952E-12
VDDREST_1 = -3.59069437E-11
VSSREST_1 = 4.08777209E-11
VDDSEL_1 = 0.00000000
```

Once the estimated characterization has been done with SHAKER, we compare the results to the electrical datasheet obtained in 0,5 μm technological process. We can make some adjustments in the command file of SHAKER if necessary.

In order to validate the schematics, we check that critical simulations run on critical instantiations are correct for all simulation conditions.

Then we can process the Layout migration.





Ex. 2.2 - GDS Strata Command file used for the RAM migration

**GDS\_IN specifications**

```
GDS_IN
{
HIERARCHY
READ "Input.gds"
/* UNIT 1000 */
ALL_CELL

/* CELL RAPTMEBI */

GRID = 0.005
SHRINK = 0.625

/* ***** */
/* GDS Input Layers */
/* ***** */

NWELL=1
ACTIVE=2
M1TEXT=5
M2TEXT=6
P1=13
PPLUS=17
CONTACT=19
M1GND=21
M1VDD=24
VIA1=25
BOUNDARY=26
LOF1=28
LOF2=29
VIA2=32
M1PORT=36
M2PORT=45
M3VDD=46
M3GND=47
M1INT=49
M2INT=50
M3INT=51
M2VDD=55
M2GND=56
M3PORT=58
M3TEXT=60
P1PORT=66
P1TEXT=67
}
```

**OPERATIVE Instructions**

```
OPERATIONS
{
ACTIVE OVERSIZE 0.04
P1 OVERSIZE 0.025
CONTACT UNDERSIZE 0.03
/* Overlap adjustment P1 on CT */
P1CT = P1 AND CONTACT
P1CT OVERSIZE 0.4
POLY1 = P1 OR P1CT
MERGE POLY1
POLY1 OVERSIZE 0.3
MERGE POLY1
POLY1 UNDERSIZE 0.3
MERGE POLY1

VIA1 OVERSIZE 0.02
/* Creation of Nplus based on Pplus */
NPLUSI = NOT PPLUS
NPLUS = NPLUSI AND BOUNDARY

/* Creation of a Metal1 layer based on layers metal1 int, gnd, vdd */
/* width adjustment. */
M1I = COPY M1INT
M1G = COPY M1GND
M1V = COPY M1VDD
M1IG = M1I OR M1G
MERGE M1IG
M1 = M1IG OR M1V
MERGE M1
M1 OVERSIZE 0.01
MERGE M1

/* Overlap adjustment M1 on V1 */
M1V1 = M1 AND VIA1
M1V1 OVERSIZE 0.4
METAL1 = M1 OR M1V1
MERGE METAL1
METAL1 OVERSIZE 0.35
MERGE METAL1
METAL1 UNDERSIZE 0.35
MERGE METAL1
}
```



### GDS\_OUT Instructions

```
GDS_OUT
{
WRITE "Result.gds"
GRID=0.1

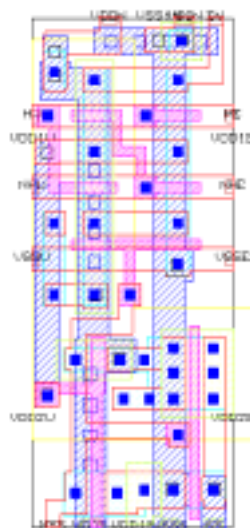
/* ***** */
/* GDS Output Layers */
/* ***** */
NWELL=5
/* PWELL=8 */
ACTIVE=10
POLY1=20
NPLUS=23
PPLUS=24
LOF1=63
LOF2=64
/* POLY2=30 */
CONTACT=34
METAL1=35
VIA1=36
METAL2=37
VIA2=38
METAL3=39
/* Boundary = CELBOX pour AMS */
BOUNDARY=41
P1PORT=68
P1TEXT=69
M1PORT=70
M1TEXT=71
M2PORT=72
M2TEXT=73
}
```

### GDS Strata generates the following output files:

- Layout database

Graphic database in GDSII stream format (see Ex. 2.3).

Ex. 2.3 - Output GDSII database



- Session Report

Text file containing messages generated by GDS Strata when executing the command file. Those messages are sent to the screen by default. The messages report the activity of the program as each command is successfully executed: Command file syntax analysis, name of processed cell,...

- Error Messages

Text file containing messages generated by GDS Strata when the syntax of the command file is incorrect.

Once the layout database has been generated by GDS Strata, there is a manual layout optimization. We then extract parasitic elements thanks to a layout extractor such as DW2000. The parameterized netlist we obtained from SHAKER at the first step is modified, and we can go through the final CAE.



### **PART 3. Final CAE**

---

This last step is to simulate the backannotated netlist obtained after parasitic layout extraction. We run the same process as the one described in PART 1. We then obtain the final Characteristic Datasheet of our RAM migrated in a 0,5  $\mu\text{m}$  technological process.

In order to validate the final characterization, we check that critical simulations run on critical instantiations are correct for all simulation conditions (same as in PART 1).