PSDesigner: A Framework for Transistor Co-Optimization.

Richard Burch, Sharad Saxena, P.K.Mozumder, Karthik Vasanth, Joseph Davis, Suraj Rao

Silicon Technology Development
Texas Instruments, Dallas, TX 75265

ABSTRACT

One challenge for modern CMOS technology is the manufacture of dissimilar transistors on a single chip at minimum cost. For example, high performance transistors for critical paths and low power transistors can be combined to create a high performance chip with reduced power consumption. One approach is to separately design each transistor and duplicate the lithography steps to allow for separate gate oxide, drain extender, channel, etc. However, cost concerns require the minimum number of duplicated lithography steps. We have developed a transistor co-optimization methodology and tool, *PSDesigner*, that can be used to simultaneously design multiple, coupled transistors to meet disparate goals. PSDesigner facilitates the exploration of the cost-performance trade-offs that result from duplicating different lithography steps.

FRAMEWORK

PSDesigner is a co-optimization framework utilizing user-defined models to represent important transistor performances (Ion, Ioff, Vt, etc.) as functions of design variables (gate oxide thickness, implant doses & energies, etc.). To allow rapid exploration of the complete design space, we utilize simple response surface models (RSMs) created from either experimental data or tuned device simulations.

METHODOLOGY

PSDesigner supports a top-down design methodology, referred to as process synthesis [1-5]. In this methodology, the user selects the structure of the design and specifies the desired performances and any constraints on the design variables. Decomposition is used to determine all combinations of design variables that satisfy all constraints. These acceptable combinations of design variables form a multi-dimensional space of solutions that we term the *acceptability region* of the design. The models inside PSDesigner can be used to map the acceptability region of the design into the performance space, forming the *capability region* of the design. The capability region represents all performance combinations that can be created and that satisfy all constraints. The ability to view all possible designs in either the design variable or the performance space is key advantage of this approach, when compared to a traditional optimization approach.

In PSDesigner, this process synthesis methodology has been modified to support concurrent top-down design (termed *co-optimization*) of multiple transistors that have separate performance models and constraints, but share some or all of their design variables. In this modified methodology, the designer chooses which variables will be shared and each device is decomposed separately. Then the *acceptability regions* of each device are made mutually consistent. Finally, hierarchical optimization is performed to choose specific device designs. The individual tasks in this methodology are detailed in the following sections.

Design Selection

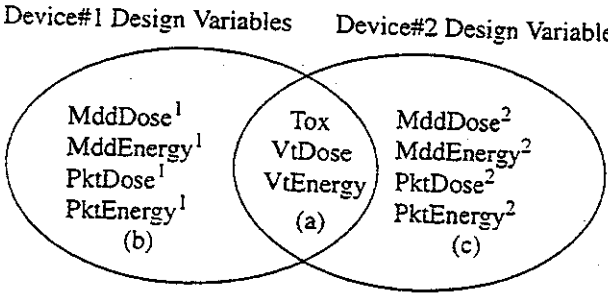
During *Design Selection*, the designer chooses which design variables will be shared between both devices and which will be independent. Physically, the design variables are closely tied to lithography steps (figure 1). If a lithography step is duplicated, the corresponding group of design variables become independent and can have different values for each transistor. To maximize flexibility, the designer would like to have as many important design variables independent as possible; however, to minimize cost the designer wants to minimize the number of duplicated lithography steps.

Figure 1: Lithography steps determine design variables

Duplicated Lithography Step.	Design Variables
Gate Oxide	Gate Oxide (Tox)
Drain Extender	Drain Extender Dose (Mdd-Dose) & Energy (MddEnergy) Pocket Implant Dose (PktDose) & Energy (PktEnergy)
Channel Implant	Threshold Adjust Dose (VtDose) & Energy (VtEnergy) Punchthrough Dose (PtDose) & Energy (PtEnergy)

One possible design selection is shown in figure 2. In this example, the drain extender lithography is duplicated. Therefore, pocket and drain extender implants are different for the two devices.

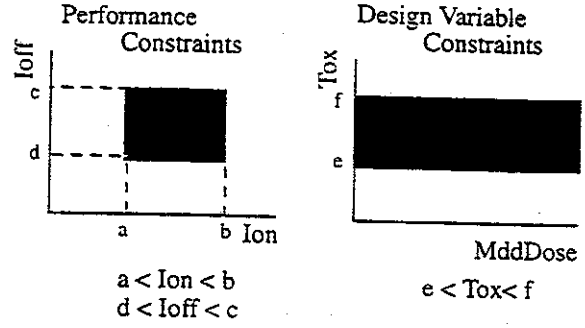
Figure 2: The design variables in this co-optimization can be divided into those that are (a) shared by both devices, (b) specific to device #1, and (c) specific to device #2.



Constraint Specification

After selecting the design, the user specifies constraints on performances and design variables (figure 3). The constraints on performances specify the desirable device and constraints on design variables limit the design variable combinations that would be acceptable to the user. If no constraint is provided for a performance, it is assumed that any value of the performance is acceptable. In addition to any user-specified constraints on design variables, each design variable is constrained to the range over which the RSM was built. Currently, simple box constraints are supported in PSDesigner; however, more complex constraints are consistent with the framework and methodology.

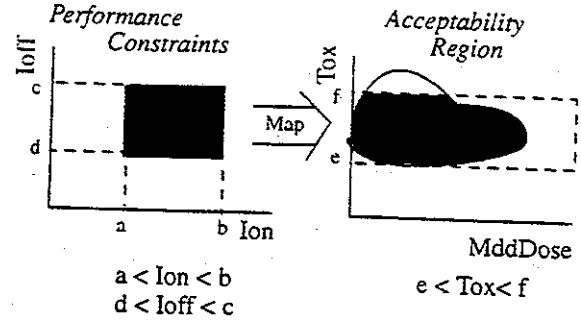
Figure 3: Examples of both performance constraints and design variable constraints are shown.



Decomposition

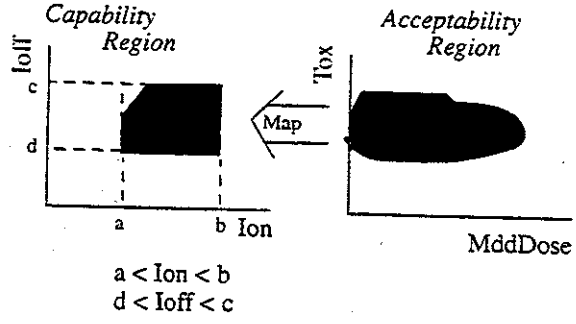
Once performance and design variable constraints have been specified, the designer performs decomposition (figure 4) which maps the performance constraints and design variable constraints into the design variable space. The resulting acceptability region contains all combinations of design variables that would produce an acceptable design.

Figure 4: Decomposition maps performance constraints into the design variable space to create the acceptability region. Design variable constraints can clip this region.



The acceptability region represents all possible designs that satisfy all constraints on both performances and design variables. It is often useful to view all possible designs in the performance space. This capability region (figure 5) is obtained from mapping the acceptability region back into the performance space.

Figure 5: The capability region of the design can be created by mapping the acceptability region back into the performance constraint space.



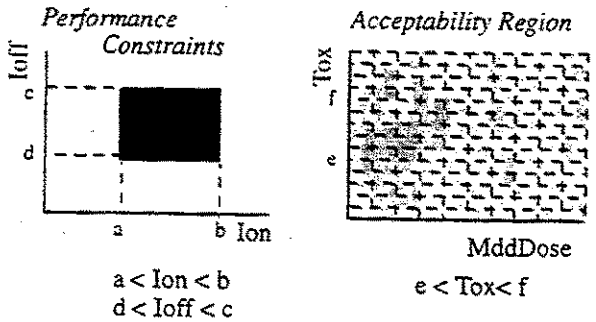
A variety of algorithms can be used to perform decomposition. The appropriate algorithm is dependent on the particular method that is used for acceptability region representation. The acceptability region can be represented as a multi-dimensional grid, a hierarchical multi-dimensional grid, or even as piecewise-linear regions. Alternate representations, such as hierarchical multi-dimensional grids and piecewise-linear regions are more efficient, but are also more complex. For simplicity, this paper will restrict itself to considering the non-hierarchical multi-dimensional grid representation and appropriate algorithms.

One simple, but effective, decomposition algorithms is as follows:

1. Create an empty acceptability region with grid sizes determined from a sensitivity analysis of the device performances (figure 6). Usually minimum grid sizes

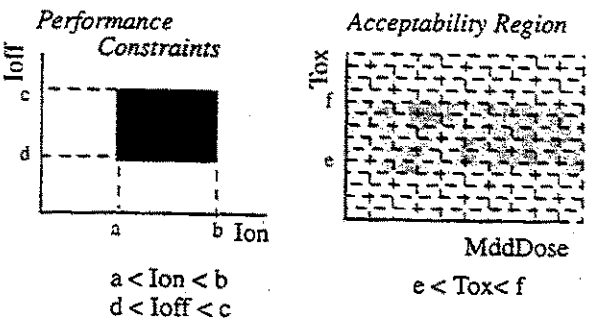
are specified for each design variable. For example, gate length may exceed the desired sensitivity even for increments of 0.01 microns; however, 0.01 microns is usually the minimum grid size of interest.

Figure 6: Create an empty acceptability region with grid sizes based on sensitivity



2. Mark each grid point that falls outside design variable constraints as unacceptable (white) (figure 7).

Figure 7: Apply design variable constraints.



3. For each grid point that does not violate design variable constraints, calculate all performances. Mark the grid as acceptable (black) if all performances meet performance constraints. Mark the grid as unacceptable (white) if any performances fail to meet performance constraints. Figure 8 illustrates this for two grid values. Figure 9 shows the final acceptability region.

Figure 8: Apply design variable constraints.

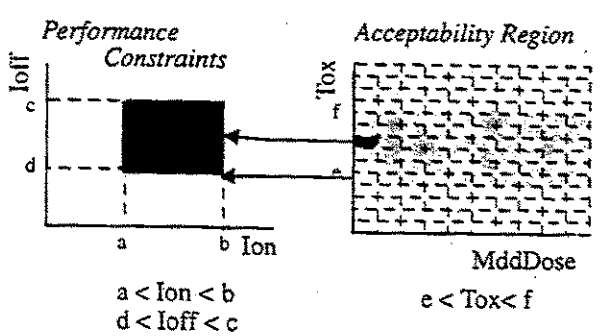
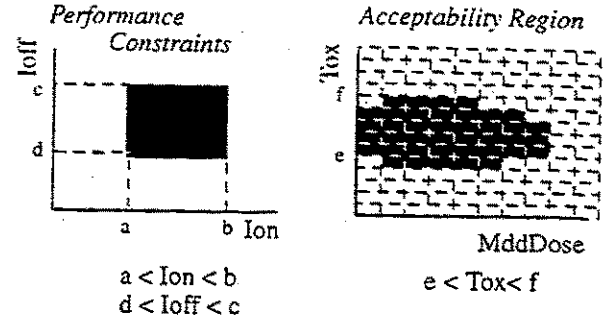


Figure 9: Final acceptability region (non-hierarchical grid representation).

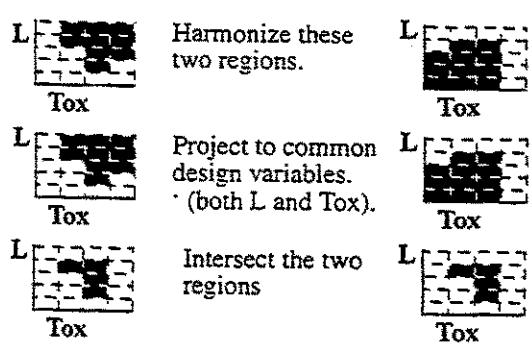


Acceptability Region Harmonization

During co-optimization, decomposition is carried out independently for each device. However, it is possible, even probable, that some of the designs in one device's acceptability region may be inconsistent with all acceptable designs in the other device's acceptability region. The process of making these acceptability regions consistent is termed *acceptability region harmonization*. The resulting *harmonized acceptability regions* represent all acceptable devices and are created by projecting each acceptability region into the sub-region of shared designables and intersecting with all other regions. If the harmonized acceptability regions are empty, then it is impossible to manufacture devices that meet all performance constraints.

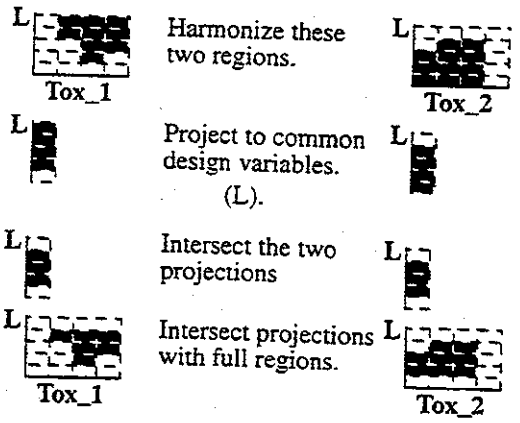
If the acceptability regions are represented as a non-hierarchical grid, then some of the acceptable grids in one device's acceptability region may be inconsistent with acceptable grids in another device's acceptability regions. Indeed there may be no consistent points. Even though each decomposition was independently successful, there are no solutions that can consistently satisfy all devices. The process of acceptability region harmonization for non-hierarchical grids can be made clearer by considering two simple two-dimensional cases. In the first case (figure 10), both regions have exactly the same design variables.

Figure 10: Consider a simple 2-dimensional case where all design variables are shared.



In the second case (figure 11), each region has one shared design variable and one different design variable. Higher dimensionality examples can be extrapolated from these simple examples.

Figure 11: Consider a simple 2-dimensional case where one design variable is shared.

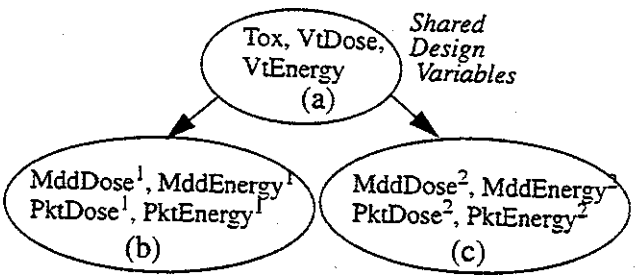


Optimization

If the harmonized acceptable regions are not empty, then acceptable devices can be made. The designer must select specific devices by simultaneously optimizing over all acceptability regions. With traditional optimization, the computational complexity could be prohibitive. Our approach breaks the design variables into dependent and independent groups and then utilizes a hierarchical optimization approach, reducing computational complexity.

A dependency tree of the design variables (figure 12) is constructed. With some limits on objective functions and for a fixed value for all designables in node (a), the design variables that maximize the total objective can be found by separately maximizing the objective for design variables in node (b) and for design variables in node (c). This gives a new function, which is often implicit, in terms of the design variables in node (a). This function can now be maximized to find the maximum over all design variables.

Figure 12: Simple 2-transistor optimization tree.



This type of optimization approach reduces computational complexity and is approximately independent of the number of devices being optimized.

Many objective functions can use this type of hierarchical optimization. Let A, B, and C represent the variables in nodes (a), (b), and (c). Mathematically, hierarchical optimization of an objective function, F(A,B,C), is possible whenever the following decomposition is true.

$$\max_{A,B,C}(F(A,B,C))$$
$$= \max_{A}(\max_{B,C}(F(B,C|A=a)))$$
$$= \max_{A}(J[\max_{B}G(B|A=a), \max_{C}H(C|A=a)])$$

where G, H, and J are functions of real variables.

The specific hierarchical optimization algorithm implemented depends on the method chosen for acceptability region representation. For a simple non-hierarchical grid representation of the acceptability region, the algorithm is a hierarchical search.

1. Project each acceptability region into regions with shared design variables (figure 12). Since the acceptability regions are harmonized, these *shared regions* will be identical.
2. Get the first grid point in the shared region.
3. Section (figure 13) each device's acceptability region with the value of the shared grid point and project the region into the variables that are not shared. This will create a sub-region for each device.
4. Search each grid point in each sub-region determine the design variable values that maximize the objective function.
5. Calculate the objective function value using the fixed values of shared variables and the maximum values of all other variables.
6. If this objective function value is greater than the previous best objective function, then store these design variable values as the current maximum.
7. Get the next grid point in the shared region, and continue with step 3.

Figure 13: To section an acceptability region at a value of one or more variables, mark every grid that does not contain that value as unacceptable.

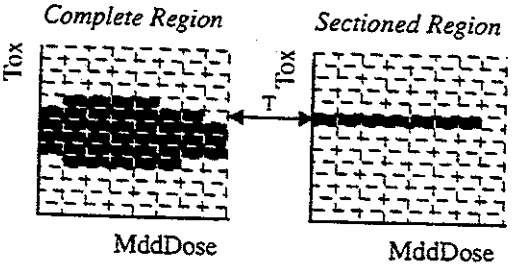
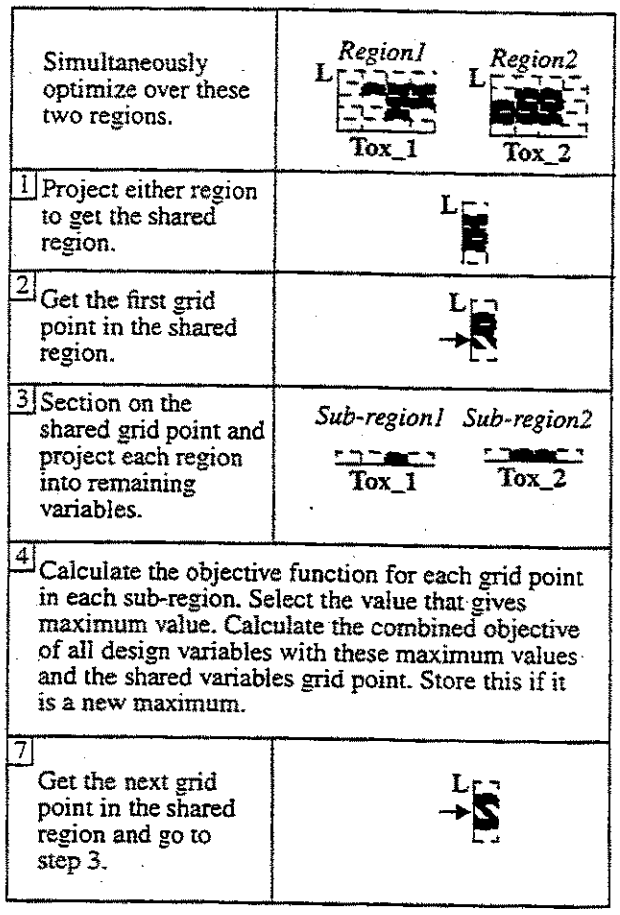


Figure 14 shows an example of this optimization algorithm for a simple two-dimensional case.

Figure 14: Example of hierarchical optimization for simple two-dimensional case.



ENHANCEMENTS

As a complete co-optimization framework, PSDesigner has many features that are not discussed in this paper. Some, such as alternate acceptability region representations, enhance the computational efficiency of the system. Other enhancements were developed to overcome specific limitations with the simple methodology presented in the previous sections. Two enhancement areas that significantly improve the functionality of PSDesigner are dimensionality reduction and complex co-optimizations.

Dimensionality Reduction

One problem in co-optimization can be the dimensionality of the problem. In many cases, the number of design variables may be prohibitively large to represent each variable in a sufficiently accurate acceptability region. Two features were implemented to allow the designer to manage the dimensionality problem.

Hill Climbing: In hill climbing, the designer leaves some variables fixed to likely values and co-optimizes a subset of the design variables. The designer then changes the subset of "active" design variables and repeats the co-optimization. Hill climbing can lead to a sub-optimal solution; however, it is a very effective means of dimensionality reduction.

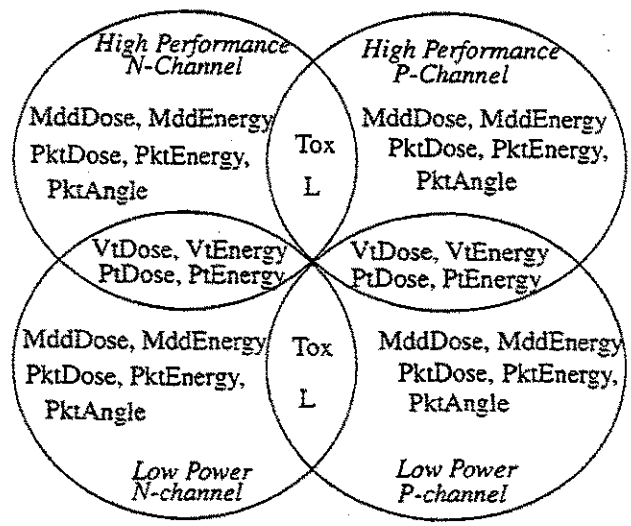
Variable Gridding: In the decomposition algorithm, the grid size was determined by sensitivity of the performances to each design variable. Since the time to perform each decomposition is proportional to the total number of grid points, more dimensions can be included in the decomposition if a coarser grid is utilized for the initial optimization.

When used in combination, hill climbing and variable gridding allow the designer to effectively co-optimize transistors with a large number of design variables.

Complex Co-optimizations

In this paper, it has been assumed that only two transistors are being co-optimized. In many interesting applications, four or more transistors may need to be co-optimized. All of the steps in our methodology have been modified to support co-optimization for any arbitrary number of transistor or other devices. Figures 15 illustrates the complexity of a CMOS co-optimization with high performance and low power transistors for both P-channel and N-channel.

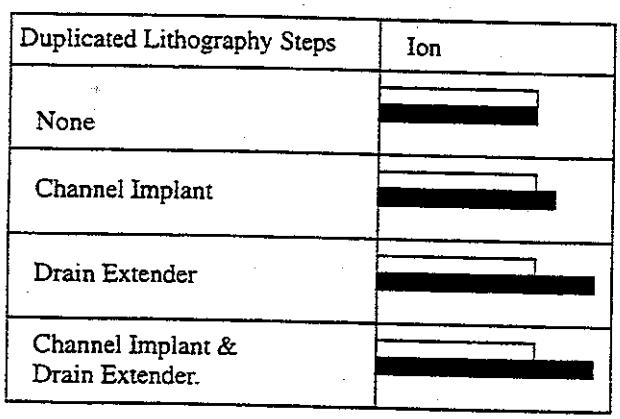
Figure 15: Design selection diagram for 4-transistor co-optimization. Each device has 11 variables. There are 32 non-shared design variables.



RESULTS

PSDesigner has been used for a variety of transistor co-optimizations. In one example (figure 16), a high performance (Ion) transistor and a low power transistor are co-optimized. The low power transistor must have an off current of less than 10 picoamps for its worst-case device. With PSDesigner, the engineer can clearly identify that, for this example, duplication of the drain extender lithograph step achieves the best performance at minimal cost. Duplicating both the channel implant and drain extender lithography steps would increase the cost of the technology, but would provide no added benefit to the design of the transistors. Without a co-optimization framework such as PSDesigner, this would be difficult to identify.

Figure 16: Results of transistor co-optimization for two transistors, one with high performance (drive current) and the other with low off current. Additional constraints, such as restricted allowable dose ranges, limit Ion. In this example, duplication of the drain extender is the most efficient and duplicating the channel implant lithography step provides no additional advantage.



CONCLUSION

PSDesigner is a co-optimization framework written to address one of the current challenges of the semiconductor industry, the manufacture of dissimilar devices on a single chip with a minimum of additional lithography steps. Because of it's general nature, PSDesigner is applicable to the co-optimization of very different technologies including standard logic, memory, and Bi-cmos. PSDesigner has demonstrated it's usefulness by reducing the lithography requirements of state-of-the-art technology with an expected saving of many millions of dollars.

PSDesigner can be extended to any other design problems that can be appropriately formulated. It can be used for any design effort where input-to-output models are applicable and where the desired component can be specified by constraints and targets on the outputs and/or inputs. Co-optimization in PSDesigner is applicable for any effort to simultaneously design two or more such components which share some inputs.

REFERENCES

[1] S.Saxena, P.K.Mozumder, A.Unruh, and R.Burch,"A Methodology for Top-Down Synthesis of Semiconductor Process Flows", *ISSM-95*.

[2] J.C.Davis, P.K. Mozumder, R.Burch, and C. Fernando, "Automatic Synthesis of Equipment Recipes from Specified Wafer-State Transitions," *Second International Workshop on Statistical Metrology*, June 1996.

[3] S. Saxena, A. Unruh, P.K.Mozumder, and R. Burch, "An Approach for Deriving Semiconductor Process Flows from Performance Requirements," *Proceeding of the First World Conference on Integrated Design and Process Technology*, 1995.

[4] S. Saxena, R. Burch, K. Vasanth, C. Fernando, S. Rao, J.C. Davis, P.K. Mozumder, (1997). "An application of Process Synthesis Methodology for First-Pass Fabrication Success of High Performance Deep Sub-Micron CMOS". *Proceedings of the IEEE Electron Devices Meeting (IEDM)*, 1997

[5] S. Saxena, R. Burch, P.K. Mozumder, K. Vasanth, S. Rao, J.C. Davis, C. Fernando, "Methods for the design of Microelectronic Devices and Process Flows for Manufacturability". *SPIE Conference on Microelectronic Device Technology*, 1997