AUTOMATIC METHODOLOGY FOR PLACING THE GUARD RINGS INTO CHIP LAYOUT TO PREVENT LATCHUP IN CMOS IC' S

Ming-Dou Ker, Hsin-Chin Jiang*, Jeng-Jie Peng*, and Tzay-Luen Shieh*

Integrated Circuits & Systems Laboratory
Institute of Electronics
National Chiao-Tung University, Taiwan

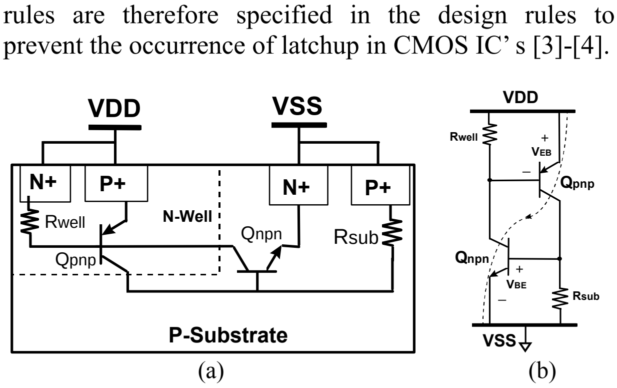
* Analog IP Technology Section
SoC Technology Center
Industrial Technology Research Institute, Taiwan

ABSTRACT: A program methodology is proposed to automatically place the guard rings in the chip layout to improve latchup immunity of the CMOS IC' s. The additional guard rings between the I/O cells and the internal circuits had been practically proven to significantly increase the latchup immunity of CMOS IC' s. Therefore, the layout spacing from the I/O cells to the internal circuits can be reduced to a reasonable distance to save the total chip size. In this paper, a "Guard Ring Automation" program to realize the additional guard rings in the layout is proposed to make the layout more automatically and accurately.

1. INTRODUCTION TO LATCHUP

Latchup in CMOS IC' s is formed by the parasitic p-n-p-n structure between the VDD and VSS [1]. The device cross-sectional view of latchup path in a p-substrate bulk CMOS technology is shown in Fig.1(a), whereas the 1st-order equivalent circuit of the latchup path is illustrated in Fig.1(b). If the latchup is fired in CMOS IC' s, it causes a low-impedance path with a high current loss from VDD to VSS to burn out the chip.

To prevent the occurrence of latchup in CMOS IC' s, some advanced process techniques (such as the epitaxial substrate, retrograde well, trench isolation, or SOI) had been used to increase the holding voltage of the parasitic p-n-p-n structure in CMOS IC' s [2]. Although such advanced techniques can effectively solve the latchup issue in CMOS IC' s, the production cost of CMOS IC' s with such advanced techniques is more expensive. Thus, the most consumer IC products are still manufactured in the bulk CMOS process. In the bulk CMOS process, the guard rings in the I/O cells and the p-substrate/n-well pickups in the internal circuits are generally used to prevent latchup occurrence in CMOS IC' s. Some layout rules are therefore specified in the design rules to prevent the occurrence of latchup in CMOS IC' s [3]-[4].



(a) (b)

Fig.1 (a) The cross-sectional view, (b) the equivalent circuit, of a latchup structure in a p-substrate bulk CMOS technology.

To verify latchup immunity of CMOS IC' s, a over-shooting (or undershooting) current is applied to I/O pins of CMOS IC' s to investigate whether latchup is fired, or not [5]. The schematic diagram to show such latchup testing is shown in Fig.2. For general CMOS IC' s, they should not be triggered into latchup by a trigger current of ±100mA on the I/O pins [5]. The lathcup specification for general consumer IC' s is listed in Table I.

In Fig.2, the overshooting/undershooting trigger current on the I/O pin is conducted into the drain regions of the output devices. When the overshooting (undershooting) current is applied to the I/O pin, the P+ drain / n-well (N+ drain / p-substrate) junction in the output PMOS (NMOS) is forward biased to generate the trigger current into the substrate. This substrate current can fire the latchup paths in the I/O cells or the internal circuits.

To avoid the latchup paths in CMOS IC' s fired by the overshooting or undershooting current on the I/O pins, the double guard rings are often used to surround the output PMOS and NMOS in I/O cells. But, the internal circuits in the bulk CMOS IC' s is still sensitive to latchup, because the PMOS and NMOS in the internal circuits are not surrounded by the guard rings. To avoid the substrate current generated from the I/O cells to cause latchup in the internal circuits, the internal circuits should keep an enough far distance away from the I/O cells. To further reduce the distance between the I/O cells and internal circuits to save chip size, the additional guard rings should be placed between the I/O cells and the internal circuits, as that shown in Fig.3 [6].

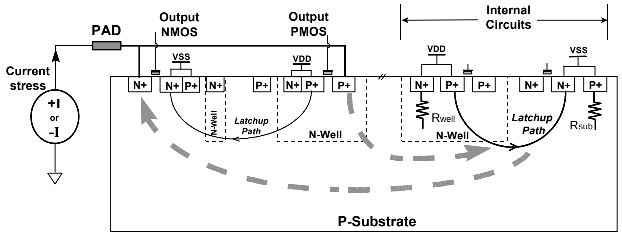


Fig.2 The latchup testing for CMOS IC' s with positive or negative trigger currents at the I/O pins.

Table I

Trigger Source	Specification
+I trigger at I/O pin	I normal + 100mA
-I trigger at I/O pin	-100mA

0-7803-7057-0/01/$10.00 ©2001 IEEE.

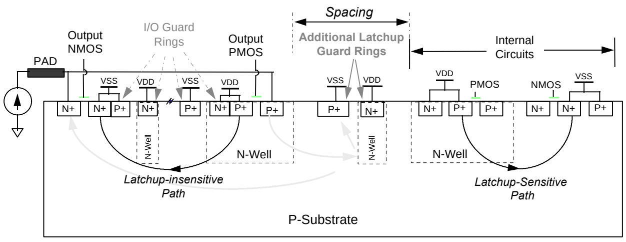


Fig.3 The additional guard rings placed between the I/O cells and the internal circuits to avoid the substrate current generated form I/O cells but to cause latchup in the internal circuits.

The trigger currents generated from the overshooting / undershooting currents on the I/O pin are collected by the additional guard rings. Therefore, the internal circuits are safe to the latchup issue. By using the additional guard rings, a shorter distance between the I/O cells and internal circuits can be drawn to reduce the total chip size of the high-pin-count IC's in bulk CMOS process [6].

2. GUARD RINGS AUTOMATION

Guard rings are formed by the p+ diffusions in the p-substrate connected to VSS and the n+ diffusions in the n-well connected to VDD. To effectively absorb the trigger current in the well or substrate, the contacts for connection to the guard rings should be added as many as possible. But, such guard rings are traditionally added by full-custom handwork. Such handwork procedure costs much time and vigor. Therefore, an automation program is developed in this work, which can reduce the time and increase the layout accuracy during chip development. In order to meet the layout environment without extra cost, the SKILL language [7] is used to develop this program, which is named as "Guard Ring Automation".

2.1 Instance and Mosaic

To provide the connection from the power lines to the guard rings, a cell called as "*instance*" was setup in a cell library. The shapes of such instances in different CMOS processes are shown in Fig.4(a) and Fig.4(b). The instance cell in Fig.4(a) only includes the thin-oxide region, contact, and metal 1, which is suitable for applying in the process has a single metal layer. If the process has more metal layers, the VIA and other metal layers can be stacked on the instance. But, in some processes, the VIA1 can not be stacked on VIA2. In this case, the instance is changed to that shown in Fig.4(b), where the VIA1 is stacked on the contact but not stacked to VIA2. Such instances for connection from the power lines to guard rings (under the power line) can be drawn in different ways according to the design rules of CMOS processes. The instances shown in Fig.4(a) and 4(b) are displayed in the master layout views. To simplify the display in the top-level design, a simple rectangle shape is used instead of the real master view to make the design more easily and get faster speed in display.

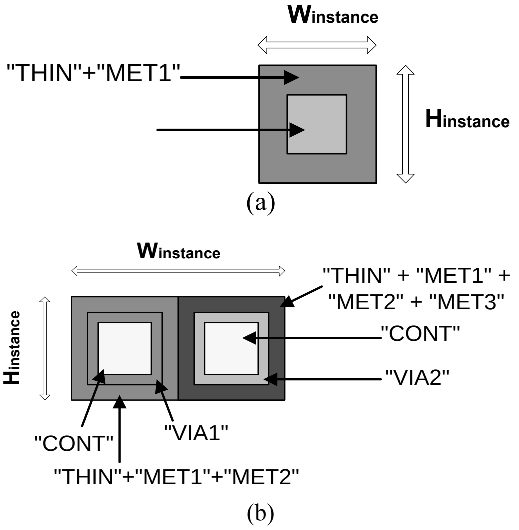
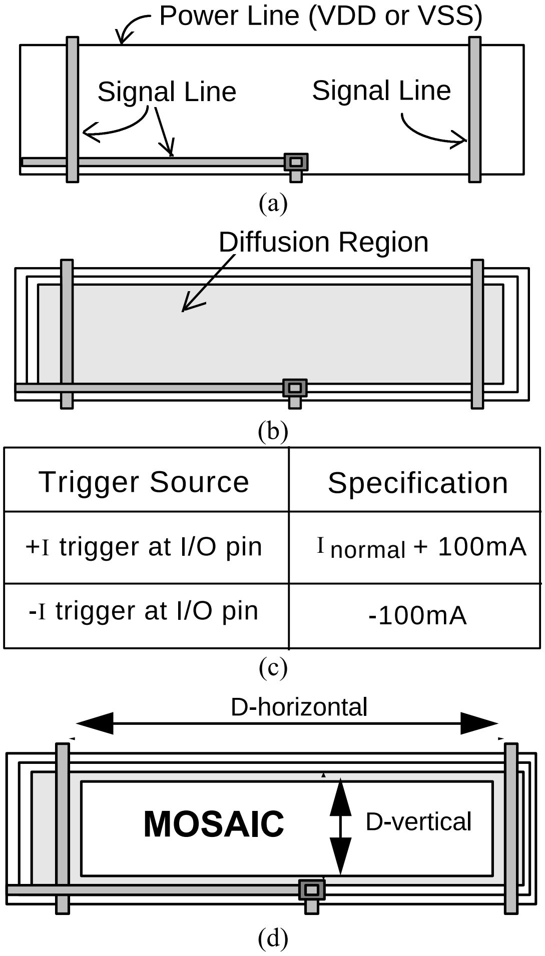


Fig.4 Layout of the instances for connection from the power lines to the guard rings with (a) a single metal layer, and (b) multiple metal layers but forbidden VIA stack.



Trigger Source	Specification
+I trigger at I/O pin	I normal + 100mA
-I trigger at I/O pin	-100mA

Fig.5 (a) The location to be added the additional guard rings. (b) The diffusion region is added under the power line. (c) The guard ring connection is formed by the instance method. (d) The guard ring connection is formed by the mosaic method.

"*Mosaic*" was a type of array, which is created by a plurality of instances. To create a mosaic, the number of instances in the column and row directions of the guard ring regions should be calculated in advance. Then, the program can automatically generate a mosaic with different number of instances for placing in the guard ring region. By using the mosaic, a lot of instances can be merged into a single symbol, which has distinct advantage in layout accuracy and speed improvement during display. The number of instances in each mosaic is dependent on the guard ring layout area. The column and row number (K_{Column} and K_{Row}) in the layout are calculated as :

K_{Column} = fix(($D_{horizontal}$ – $2(S_{min} + S_{im}$) / $W_{instance}$) ;

K_{Row} = fix (($D_{vertical} - 2(S_{min} + S_{im}$)) / $H_{instance}$) ;

S_{min} = the minimum spacing between two metals ;

S_{im} = the spacing between instance and metal ;

$H_{instance}$ = the height of an instance;

$W_{instance}$ = the width of an instance;

$D_{horizontal}$ = the width of guard ring region; and

$D_{vertical}$ = the height of guard ring region.

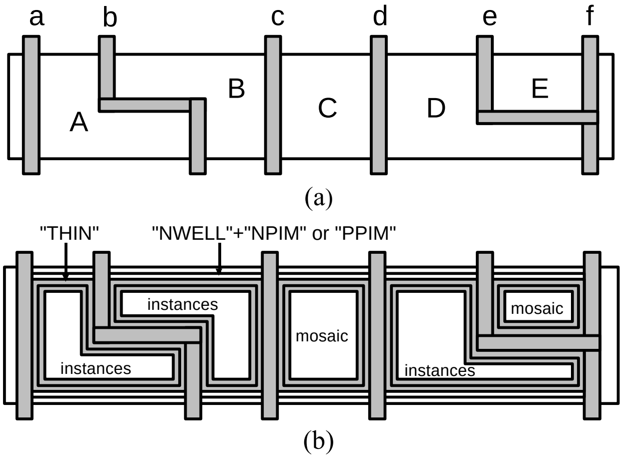
The function of “fix(..)” is to get the integer from the calculated number. Therefore, the number of instances in a specified mosaic is equal to $K_{Column} \times K_{Row}$.

The procedure to add the additional guard ring under a specified power lines is illustrated in Fig.5(a) ~ 6(d). In Fig.5(a), there are a power line and three signal lines in the layout. The guard ring is planed to be added in the layout area under the power line. The proposed “Guard Ring Automation” program can check the layout area and add automatically the diffusion layer under the power line region, as that shown in Fig.5(b). If the power line is VDD, the inserted diffusion layer is an N+ diffusion with N-well in the substrate. If the power line is VSS, the inserted diffusion layer is a P+ diffusion in the substrate. The program can choose the diffusion type (N+ or P+) automatically. In Fig.5(c), the guard ring connection from the diffusion layer to the power line is formed with the instance method by the program. In Fig.5(d), the guard ring connection from the diffusion layer to the power line is formed with the mosaic method by the program.

If there is a curved metal line across the region where is wanted to be added the guard ring, the instance method can provide more connection to the diffusion layer of the guard ring. So in some situation, it is more efficient to use the mosaic method. But sometimes, it becomes more efficient to use the instance method. The proposed Guard Ring Automation program can automatically divide a guard ring region into several sub-area regions dependent on the curved metal lines across the guard ring region. If the area is a rectangle, the program will use the mosaic automatically. But, if the area is not a rectangle, the program will use instance to get the most area efficiency in the guard ring placement. Thus, the proposed Guard Ring Automation program gets a balance between the speed and the area efficiency to add the additional guard rings for latchup prevention.

The example to add the guard ring under a specified power line with curved signal lines across it is shown in Fig.6(a) and 6(b). The original layout before adding the guard ring under the power line is shown in Fig.6(a). By using the Guard Ring Automation program, the guard ring is added under the specified power line as that shown in Fig.6(b). The program can automatically decide to use the mosaic method on a rectangle region, and to use the instance method on a non-rectangle region, to place the contact of guard ring as many as possible in the layout. The executing speed in the region, which using the instance, would be slower than that using the mosaic.

In Fig.6(a), the signal lines such as a, b, c, d, e, and f pass through the region where to be added the guard ring. So the program divided this guard ring to five sub-area such as A, B, C, D, and E. At first, the N+ (or P+) diffusion layer of guard ring is added. Then, the program will decide to use the instance or mosaic in individual five sub-area regions. For the non-rectangle A, B, and D regions, the instance is used to add the guard ring. For the rectangle C and E regions, the mosaic is used to add the guard ring to improve the speed of execution and display. Finally, the guard rings are automatically added by the program as that shown in Fig.6(b).



(a)

(b)

Fig.6 The layout region (a) before, and (b) after, adding the guard ring under the power line by the Guard Ring Automation program.

2.2 Program Flowchart

The flowchart of this Guard Ring Automation program is shown in Fig.7. When the program is loaded into the OPUS, the program will build a pull-down menu. If the user invokes the command, there is a form of “Define Variables” would appear. The user can define some spacing variables or the default minimum values from the design rules. First, the program will determine the type of diffusion layer of each guard ring, and get the correct instance from a cell library. Then, the program will add the n-well, implant, and diffusion layer in the region where the guard rings to be added. The program will create a temp layer for the guard ring, and find all metal paths which overlapped the guard ring region, then chop the temp layer dependent on the metal paths. The program can get all temp layers and determine their shape. If the shape is a rectangle, the program calculates the width and height for the area to fill the mosaic with the most number of instances. For speed consideration, the program would add the guard rings with mosaic method if the area is a rectangle. If the area is not a rectangle, the program will use the instance method to fill the most instances in the guard ring region.

This Guard Ring Automation program is developed on a general layout environment. To run this Guard Ring Automation program, we must invoke OPUS first, and make sure the library is correct. Then, do the following steps to run the Guard Ring Automation program :

(1) Load the program to OPUS.

(2) Select to run VDD or VSS guard rings in a pull-down menu.

(3) Select the guard ring. If the guard ring had four sides, select them all.

(4) Input spacing variables in the pop-up menu “Define Variables”, or the default minimum values.

(5) The program adds the guard rings automatically.

(6) Done.

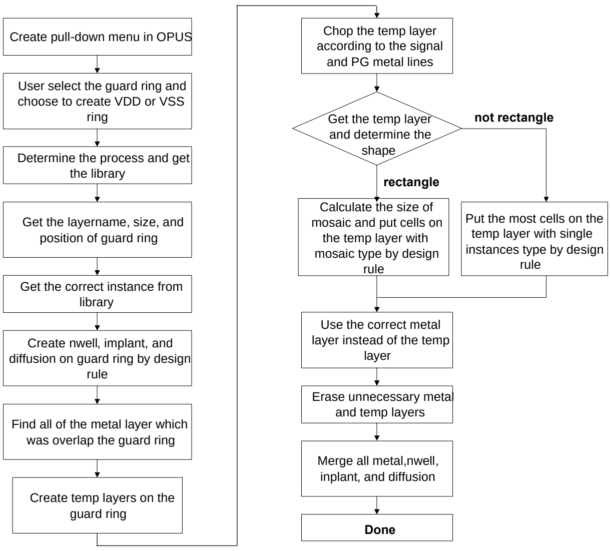

Create pull-down menu in OPUS
User select the guard ring and choose to create VDD or VSS ring
Determine the process and get the library
Get the layername, size, and position of guard ring
Get the correct instance from library
Create nwell, implant, and diffusion on guard ring by design rule
Find all of the metal layer which was overlap the guard ring
Create temp layers on the guard ring
Chop the temp layer according to the signal and PG metal lines
Get the temp layer and determine the shape
not rectangle
rectangle
Calculate the size of mosaic and put cells on the temp layer with mosaic type by design rule
Put the most cells on the temp layer with single instances type by design rule
Use the correct metal layer instead of the temp layer
Erase unnecessary metal and temp layers
Merge all metal,nwell, inplant, and diffusion
Done

Fig.7 The flowchart of the Guard Ring Automation program.

3. APPLICATION EXAMPLE

The proposed Guard Ring Automation program had been used to add the additional guard rings in some mass- production IC products. In the IC layout, there are VDD and VSS power lines surrounding the chip, which are placed between the I/O cells and the internal circuits, or among the circuit blocks. Such power lines are often drawn with a wide metal width, such as 20 ~ 30 μm, to provide the current for circuit operations in the chip.

By using the proposed Guard Ring Automation program, the additional guard rings can be quickly added into the chip. A practical layout example with the additional guard rings is shown in Fig.8. With the additional guard rings on a chip fabricated in a 0.5-μm bulk CMOS process, the trigger current on I/O pins to induce latchup in such a chip can be greater than ±500 mA.

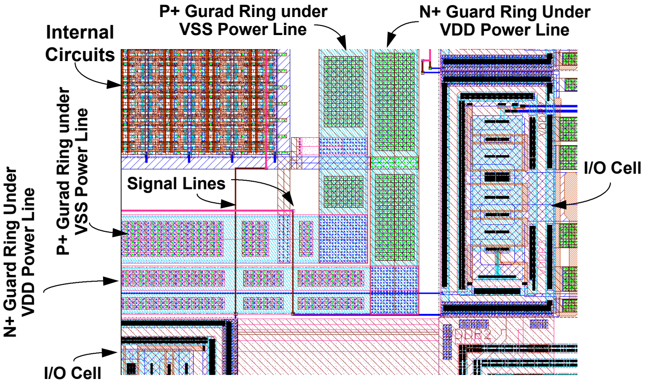

P+ Gurad Ring under VSS Power Line
N+ Guard Ring Under VDD Power Line
Internal Circuits
I/O Cell
Signal Lines
P+ Gurad Ring under VSS Power Line
N+ Guard Ring Under VDD Power Line
I/O Cell

Fig.8 The partial layout of a 8-bit micro-controller IC around the I/O cells and internal circuits after dding the additional guard rings by the Guard Ring Automation program.

4. CONCLUSION

In order to improve the latchup immunity for high-reliability applications and to reduce the layout cycle time, a Guard Ring Automation program has been developed and demonstrated in this work. This program provides a simple and accurate method for realizing the additional latchup guard rings in the CMOS chips. By using this program, the layout cycle time can be effectively shortened but also can prevent the mistakes due to the handwork in the layout.

5. REFERENCES

[1] M. J. Hargrove, S. Voldman, R. Gauthier, J. Brown, K. Duncan, and W. Craig, “Latchup in CMOS Technology,” *IEEE Int. Reliability Physics Symp.*, 1998, pp. 269-278.

[2] J. Y. Chen, *CMOS Devices and Technology for VLSI*, Prentice-Hall International, 1990.

[3] R. Menozzi, L. Selmi, E. Sangiorgi, G. Crisenza, T. Cavioni, and B. Ricco, “Layout dependence of CMOS latchup,” *IEEE Trans. on Electron Devices*, vol. 35, pp. 1892-1901, 1988.

[4] T. Aoki, “A practical high-latchup immunity design methodology for internal circuits in the standard cell-based CMOS/BiCMOS LSI’s,” *IEEE Trans. on Electron Devices*, vol. 40, pp. 1432-1436, 1993.

[5] *IC Latch-up Test*, EIA/JEDEC Standard No. 78, Electronic Industries Association, 1997.

[6] M.-D. Ker, W.-Y. Lo, and C.-Y. Wu, “A new experimental methodology to extract compact layout rules for latchup prevention in bulk CMOS IC’s”, in *Proc. of IEEE Custom Integrated Circuits Conf.*, 1999, pp. 143-146.

[7] SKILL Language User Guide 4.4.1, Cadence Design System, Inc, USA, 1997.