Copy of: Proc. IEEE 1998 Int. Conference on Microelectronic Test Structures, Vol.11, March 1998

Strategy to Disentangle Multiple Faults to Identify Random Defects within Test Structures

Christopher Hess, Larg H. Weiland

Institute of Computer Design and Fault Tolerance (Prof. Dr. D. Schmid)
University of Karlsruhe, P. O. Box 6980, 76128 Karlsruhe, Germany
Phone: +49-721-6084217; FAX: +49-721-370455; email: hess@ira.uka.de; http://goethe.ira.uka.de/ddg

***Abstract* - Defect inspection is required for process control and to enhance chip yield. Electrical measurements at test structures are commonly used to detect faults. To improve the accuracy to evaluate the defects that have caused such faults, this paper presents a strategy to analyze single and multiple faults to precisely determine the number, layer and location of randomly distributed defects within a test structure layout. For that we first discuss the possibilities to analyze faults within known test structure layouts. Then, we present modified test structure layouts to improve the analysis of multiple faults. Finally we introduce a methodology to disentangle multiple faults by calculating and comparing the probability of possible defect locations within large test structure layout areas.**

1 Introduction

DEFECTS (e. g. particles) can become the cause of electrically measurable faults (killer defects) dependent on the chip layout and the defect size. These faults are responsible for manufacturing related malfunctions of chips. So, defect density distributions are important for yield enhancement and to control quality of process steps and product chips [StRo95]. Test structures will be used to detect faults as well as to identify and localize defects. Faults will be detected measuring voltage and current values between different pads of a test chip. Due to the known test structure layout, specific measurement limits will be expected, if no defect occurs. Measured values outside the limit will be called a fault. A **single fault** enables a precise assignment to a specific region within the test structure that contains the defect, which caused the measured fault. So - without doubt - it will be possible to provide the layer and the layout elements containing a defect. If such a layout specific assignment is not clear, the measured fault will be called a **multiple fault**, because there are multiple possibilities where one or even more defects occur within the test structure layout. In most cases, single faults are the result of measurements between two pads only. Multiple faults occur, if more than just two pads are involved in the collection of faulty measurement values.

To provide precise defect densities for yield enhancement, it is necessary not only to handle single faults, but also to handle multiple faults. For that, we present strategies to disentangle multiple faults that occur within test structures. So, Section 2 describes known test structures to measure faults and evaluate defects. Section 3 introduces novel algorithms to disentangle multiple short circuit faults. Section 4 discusses methodologies to improve the evaluation of defects that result in open circuits. Finally we present some experimental results and conclude our approach.

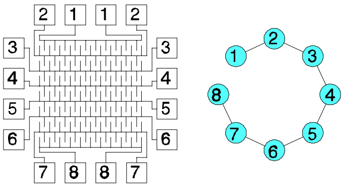
2 Test Structures to Provide Fault Analysis

Generally two different principles will be used to design test chips to detect faults and defects that may occur during semiconductor manufacturing. The first principle to design test chips contains many small test structures, each electrically distinguishable to each other and therefore connected to an individual 2-by-N pad array [Bueh83]. Comb lines will be used to detect short circuits and meandrous lines and strings of vias and contacts will be used to detect open circuits. Each test structure just covers a small chip area to ensure that no more than one defect may occur [KhMT94]. This design principle prevents the occurrence of multiple faults. But, more than half of the test chip area will be covered by the 2-by-N pads itself and not by the defect sensitive test structures. So, many test chips are necessary, to successfully detect random faults. For this reason, 2-by-N organized test chips will mainly be used to detect systematic faults only. They should not be used to detect any random problems, due to their enormous waste of chip area.

To save money when investigating random problems, it is required to more efficiently use the area within a test chip. For that, a second principle to design test chips provides a large consecutive area within boundary pads that contains just one large test structure layout. So, for example memory chips are used to detect defects. [KMGS94] provides procedures to detect and count defects. But, memory chips require a well working frontend and backend process. To just investigate selected process steps, large area comb lines or meandrous lines and strings of vias and contacts will be implemented within boundary pads in specific layers only [IpSa77]. But, due to the large area covered by a single test structure, it remains unknown how many defects caused a measured fault. So, defect density values tend to be to small. Furthermore, it is difficult to assign a multiple fault to the defects that are responsible for the measured fault pattern. So, we have to discuss ways to improve the evaluation of defects, that result either in short circuits or open circuits.

3 Evaluation of Short Circuits

To explain the principle to extract defects that result in short circuits, it is worth to model a test structure using the neighborhood graph introduced by [HeWe94] and [HeSt94] and summarized in the following table:

layout elements to detect defects	comb lines (e. g. [Bueh83], [LYWM86])
test structure and model of test structure	 Fig. 1: Left: Known test chip arrangement. Right: **Neighborhood graph** (model).
nodes of graph	**maximal conductive component (MCC)**: Maximal set of conductive components (layout objects) that are connected to each other by the designer.
edges of graph	**pair of MCCs** (=possible undesigned short circuits): Two nodes are connected by an edge if anywhere inside the test chip these MCCs are adjacent with only nonconducting material between them.

Tab. 1: Neighborhood graph to model a test structure.

An undesigned short circuit defect is only detectable between two different MCCs - therefore called **MCC-pair**. The number of different MCCs is limited to the number of pads. To increase the number of MCC-pairs or separable short circuits, respectively, all MCCs have to be arranged inside a test chip in a way that each MCC is once adjacent to every other MCC with only nonconducting material between them. This results in the complete neighborhood graph containing all possible edges as can be seen in the following figure.

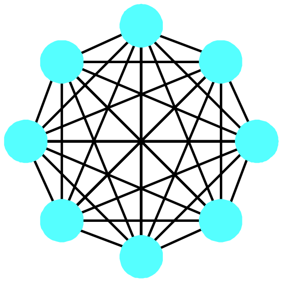


Fig. 2: Complete neighborhood graph (here e. g. *m=8* nodes and $\binom{m}{2}$*=28* edges).

The complete neighborhood graph will be implemented within Checkerboard Test Structures described at [Hess94], [HeSt94], [HeWe95b] and [HeWB97a]. Checkerboard Test Structures partition the chip area into a large number of subchips, each containing different edges of the complete neighborhood graph, which enables the separation and localization of defects anywhere inside or in between numerous layers. There is no limitation to the number of layers and no requirement of any active semiconductor devices. More than 90% of the total chip area is completely filled with defect sensitive structures. So, there is a large defect sensitive area to detect random defects even if the average defect density is low.

If just two pads are connected by a short circuit, it is possible to conclude the subchip that contains the defect, because each MCC-pair can be clearly assigned to a specific subchip and its containing test structure layout objects. But, a short circuit measured between more than just two pads results in a **multiple fault** that indicates an undefined number of defects. So, the following methodology to disentangle multiple faults will help to separate and localize the minimum number of defects that are the reason for a measured multiple fault.

```
PROCEDURE   GET_MCC_PAIR (S)
S      := {$MCC_i$|1≤i≤k ∧ k∈ℕ}; set of shorted MCCs
k      := number of MCCs in S
i,j    := temporary indices of MCCs in S
$P_n$     := {p,q,row,col,subset,weight};  pair of
                                            shorted MCCs
p,q    := MCCs
row    := row index of subchip
col    := column index of subchip
subset := subset index
weight := factor to count a defect
n      := number of possible pairs of MCCs
n = 0
FOR i=1 TO k STEP i=i+1 DO
  FOR j=i+1 TO k STEP j=j+1 DO
    n            = n+1
    $p_n$           = $MCC_i$
    $q_n$           = $MCC_j$
    ($row_n$,$col_n$)  = GET_POSITION ($p_n$,$q_n$)
  REPEAT
REPEAT
RETURN ({p,q,row,col}$_n$ ∈ $P_n$)
```

Fig. 3: Procedure to extract MCC-pairs *P* out of the set *S* of MCCs measured in a multiple short circuit.

```
PROCEDURE   GET_SUBSET ($P_n$)
$P_n$              := {p,q,row,col,subset,weight}
n               := number of pairs of MCCs
j, start, ptr := temporary indices of P
start        = 1
ptr          = start+1
$subset_{start}$  = 1
WHILE start < n DO
  j = ptr
  WHILE j ≤ n DO
    IF COMPARE_POSITION($P_{start}$,$P_j$) = true THEN
      $subset_j$   = $subset_{start}$
      EXCHANGE $P_j$ WITH $P_{ptr}$
      ptr       = ptr+1
    ENDIF
    j = j+1
  REPEAT
  IF start+1 = ptr THEN
    $subset_{ptr}$  = $subset_{start}$ + 1
    start      = ptr
    ptr        = ptr+1
  ELSE
    start = start+1
  ENDIF
REPEAT
RETURN ({subset}$_n$ ∈ $P_n$); $P_n$ also sorted by {subset}
```

Fig. 4: Procedure to determine subset indices indicating comparable MCC-pairs and also to sort all MCC-pairs P_n by these index values.

The set *S*=*{MCC*$_i$*|1≤i≤k* $\wedge$ *k*∈ℕ*}* will contain the indices of all *k* MCCs that are involved in a multiple fault. The first step is now to extract all *n* MCC-pairs each described in the data set *P={p,q,row,col,subset,weight}* where *p* and *q* represent the two MCC indices. (ref. procedure "**GET_MCC_PAIR**" of Fig. 3). The sub-procedure "**GET_POSITION**" (ref. Fig. 3) provides the location of an MCC-pair within a test structure layout. If e. g. Checkerboard Test Structures are used, [HeSt94], [HeWe95b] and [HeWB97a] provide the necessary localization procedures that yield the row and column index *{row,col}* of the subchip containing the MCCs *p* and *q*. The second step is to evaluate those MCC-pairs that are neighbored within the test chip area. For that, the procedure "**GET_SUBSET**" will give the same subset index to all neighbored MCC-pairs *P={p,q,row,col,***subset***,weight}* (ref. Fig. 4).

```
PROCEDURE   COMPARE_POSITION ($P_i$,$P_j$)

$P_2$     :=  {p,q,row,col,subset,weight}
i,j    :=  index of P

cmp    :=  Position of MCC pair comparable (yes/no)

cmp = false

CASE   design = Checkerboard Test Structure
   AND designed subchip rows ≥ designed subchip columns
  THEN
  IF    (|$row_i$ - $row_j$| ≤ 1 AND $col_i$ = $col_j$)
     OR (|$col_i$ - $col_j$| ≤ 1 AND $row_i$ = $row_j$) THEN
    cmp = true
  ENDIF
CASE   design = Checkerboard Test Structure
   AND designed subchip rows < designed subchip columns
  THEN
  IF |$col_i$ - $col_j$| ≤ 1 AND $row_i$ = $row_j$ THEN
    cmp = true
  ENDIF
ENDCASE
RETURN (cmp)
```

Fig. 5: Sub-procedure to evaluate comparable subchip positions within Checkerboard Test Structures.

```
PROCEDURE   GET_BUNDLE ($P_n$)

$P_n$   := {p,q,row,col,subset,weight};  has to be sorted
                                        by {subset}
n    := number of pairs of MCCs

j    := temporary index of P

$B_t$   := {$MCC_i$|1≤i≤b≤k ∧ b∈ℕ}
$b_t$   := number of MCCs in B
t    := number of bundles

t = 0

FOR j=1 TO n STEP j=j+1 DO
  IF $subset_j$ ≠ t THEN
    t    = t + 1
    $b_t$   = 0
  ENDIF
  IF $p_j$ ⊄ $B_t$ THEN
    $b_t$       = $b_t$ + 1
    $MCC_{t,bt}$  = $p_j$
  ENDIF
  IF $q_j$ ⊄ $B_t$ THEN
    $b_t$       = $b_t$ + 1
    $MCC_{t,bt}$  = $q_j$
  ENDIF
REPEAT
RETURN ($B_t$)
```

Fig. 6: Procedure to determine MCC-bundles containing all MCCs of neighbored MCC-pairs.

If Checkerboard Test Structures are used, the decision weather two MCC-pairs are neighbored or not will be made using the procedure "**COMPARE_POSITION**" (ref. Fig. 5). The third step is to create **MCC-bundles** *B*$_t$*={MCC*$_i$*|1≤i≤b≤k* $\wedge$ *b*∈ℕ*}* each containing the MCC indices of neighbored MCC-pairs having the same subset index (ref. procedure "**GET_BUNDLE**" of Fig. 6). Now, step four will determine the minimal set of *d* MCC-bundles *B* in a way that they together contain all *k* MCC-indices of *S*. To get a valid solution, it is also required that each MCC-bundle within the solution must have at least one common MCC index with at least one other MCC-bundle within the solution. For that, the procedure "**GET_DEFECT_NUMBER**" first checks whether there is an MCC-bundle containing all *k* MCCs of *S* (ref. Fig. 7). If there is no such MCC-bundle, the recursive procedure "**COMBINE_BUNDLE**" tries to find solutions for an initial number *d=2* of combined MCC-bundles and returns the number *mcl* of valid solutions (ref. Fig. 8).

```
PROCEDURE   GET_DEFECT_NUMBER (S,$B_t$)

S    := {$MCC_i$|1≤i≤k ∧ k∈ℕ}
k    := number of MCCs in S
$B_t$   := {$MCC_i$|1≤i≤b≤k ∧ b∈ℕ}
$b_t$   := number of MCCs in B
$occ_t$ := occurrence of B in solutions
t    := number of bundles
ptr  := index of B
d    := number of defects

tmp  := {$MCC_i$|1≤i≤k ∧ k∈ℕ}; temporary set of MCCs
i    := temporary index of B

mcl  := total number of solutions

mcl  = 0
d    = 0

/* check for single defect */

FOR i=1 TO t STEP i=i+1 DO
  IF $b_i$ = k THEN
    d    = 1
    mcl  = mcl + 1
    $occ_i$ = 1
  ELSE
    $occ_i$ = 0
  ENDIF
REPEAT

/* check for multiple defect */

IF mcl = 0 THEN
  d = 1
  WHILE mcl = 0 AND d < k-1 DO
    ptr  = 1
    d    = d+1
    tmp  = ∅
    mcl  = COMBINE_BUNDLE (S,$B_t$,ptr,d,tmp)
  REPEAT
END

/* delete MCC-pairs not involved in a solution */

IF mcl > 0 THEN
  FOR i=1 TO t STEP i=i+1 DO
    IF $occ_i$ = 0 THEN
      $B_i$ = ∅; ⇒ $b_i$ = 0
    ENDIF
  REPEAT
ENDIF

RETURN (d, mcl, $B_t$)
```

Fig. 7: Procedure to determine the minimal set of MCC-bundles to be responsible for a measured multiple fault.

```
PROCEDURE   COMBINE_BUNDLE (S,$B_t$,ptr,d,tmp)

S      := {$MCC_i$|1≤i≤k ∧ k∈ℕ}
k      := number of MCCs in S
$B_t$    := {$MCC_i$|1≤i≤b≤k ∧ b∈ℕ}
$b_t$    := number of MCCs in B
$occ_t$  := occurrence of B in at least one solution
t      := number of bundles
ptr    := index of B
d      := number of defects

tmp    := {$MCC_i$|1≤i≤k ∧ k∈ℕ}; temporary set of MCCs
i      := temporary index of B
j      := temporary number of solutions

mcl    := total number of solutions

mcl = 0

IF d > 1 THEN

  FOR i=ptr TO t-d+1 STEP i=i+1 DO

    IF tmp = ∅ OR tmp ∩ $B_i$ ≠ ∅ THEN

      IF i > ptr THEN

        EXCHANGE $B_i$ WITH $B_{ptr}$
        i = ptr

      ENDIF

      j = COMBINE_BUNDLE (S,$B_t$,i+1,d-1,(tmp ∪ $B_i$))
      mcl  = mcl + j
      $occ_i$ = $occ_i$ + j
      ptr  = ptr + 1

    ENDIF

  REPEAT

ELSE

  FOR i=ptr TO t STEP i=i+1 DO

    IF tmp ∩ $B_i$ ≠ ∅ AND tmp ∪ $B_i$ = S THEN

      mcl  = mcl + 1
      $occ_i$ = $occ_i$ + 1

    ENDIF

  REPEAT

ENDIF

RETURN (mcl)
```

Fig. 8: Recursive sub-procedure to combine MCC-Bundles.

```
PROCEDURE   WEIGHT_MCC_PAIR ($P_n$,$B_t$,d)

$P_n$    := {p,q,row,col,subset,weight}; MCC-pair
n      := number of pairs of MCCs
$B_t$    := {$MCC_i$|1≤i≤b≤k ∧ b∈ℕ}; MCC-bundle
$b_t$    := number of MCCs in B
t      := number of bundles
d      := number of defects

i      := temporary index of B
j      := temporary indices of P

h      := number of MCC-pairs in disentangled solutions

h = 0

FOR j=1 TO j<n STEP j=j+1 DO

  i = $subset_j$

  IF $B_i$ ≠ ∅ THEN

    h = h + 1

  ENDIF

REPEAT

FOR j=1 TO j<n STEP j=j+1 DO

  i = $subset_j$

  IF $B_i$ ≠ ∅ THEN

    $weight_j$ = d / h

  ENDIF

REPEAT

RETURN ({weight}$_n$ ∈ $P_n$)
```

Fig. 9: Procedure to weight each MCC-pair.

If a solution has not been found, *d* has to be increased by "1" and the procedure "**COMBINE_BUNDLE**" has to be run again until at least one solution will be found (*mcl>0*). At the end of this procedure *mcl* indicates the total number of possible solutions for the smallest possible number *d* of combined MCC-bundles. Furthermore all MCC-bundles, that are not part of at least one solution will be deleted ($B=\emptyset$). Now, the value of *d* gives the minimum number *d* of defects that have caused the measured multiple fault. And all still existing MCC-bundles indicate possible defect locations within the chip area. So, we get the MCC-pairs and also the subchips that may contain a defect. If there are more MCC-pairs than the minimum number *d* of defects, we finally have to weight each MCC-pair *P={p,q,row,col,subset,**weight**}* using the procedure "**WEIGHT_MCC_PAIR**" (ref. Fig. 9).

The following example will explain the procedure to disentangle multiple short circuits as illustrated in Figure 10 and described in Table 2. A multiple short circuit is measured between the pads or MCCs, respectively *S={3,13,16,18,20}*. The tracks of the MCCs are marked with differently dashed lines. First, all possible MCC-pairs *P* will be extracted and clearly assigned to a so-called **faulty subchip**. Then, MCC-bundles *B* will be determined by collecting the MCCs of those MCC-pairs, that are located in neighbored subchips. There is no MCC-bundle that contains all *k=5* MCCs, but the procedure "**COMBINE_BUNDLE**" detects, that the MCC-bundles B_3 and B_4 contain all MCCs of the multiple short circuit. So, the minimum number of defects is *d=2*. This is the only possible solution (*mcl=1*), because no other pair of MCC-bundles B_i,B_j will meet: $S=B_i \cup B_j \wedge B_i \cap B_j \neq \emptyset$. So, one defect located within the subchips {5,10} and {5,11} as well as another defect located within the subchips {2,14} and {2,15} are responsible for the measured multiple fault.

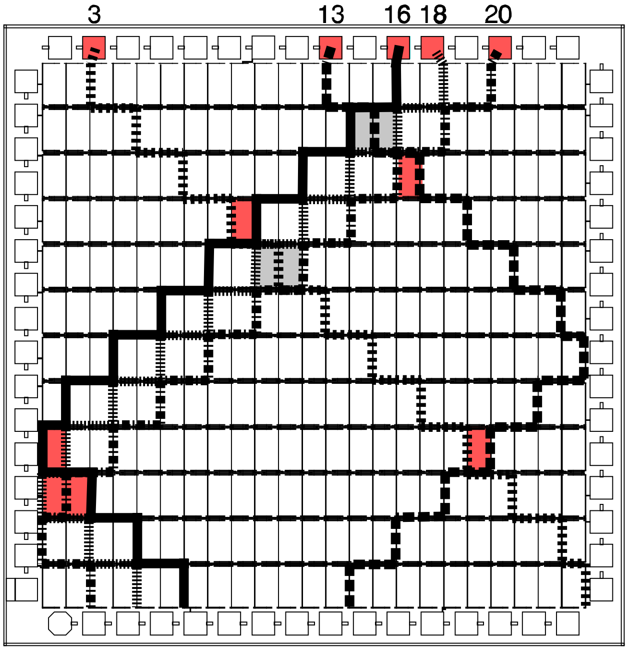


Fig. 10: Checkerboard Test Structure containing defects connecting 5 pads.

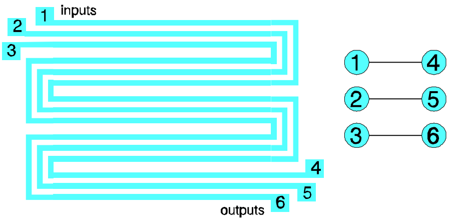
The described disentangling procedure should be individually done per layer, if defects may be located within a single layer only. The disentangling procedure will be done just once over all layers, if defects may be located within a layer as well as between adjacent layers. The disentangling procedure is applicable to all kind of test structures that enable the localization of short circuits or MCC-pairs, respectively. For that, only the algorithms "**GET_POSITION**" and "**COMPARE_POSITION**" have to be adapted to a specific test structure design. So, not just Checkerboard Test Structures but also SRAM based test structures as well as Harp Test Structures [HeWe97a] may be evaluated.

subset index $\{subset\}_{10} \subset P_{10}$	MCC-pairs $\{p,q\}_{10} \subset P_{10}$	faulty subchip $\{row,col\}_{10} \subset P_{10}$	MCC-bundles B_7	weight $\{weight\}_{10} \subset P_{10}$
$\{1\}_1$	$\{3,13\}_1$	$\{9,19\}_1$	$\{3,13\}_1 \rightarrow \varnothing$	$\{0\}_1$
$\{2\}_2$	$\{3,16\}_2$	$\{4,9\}_2$	$\{3,16\}_2 \rightarrow \varnothing$	$\{0\}_2$
$\{3\}_3$	$\{3,18\}_3$	$\{5,10\}_3$	$\{3,18,20\}_3$	$\{0.5\}_3$
$\{3\}_4$	$\{3,20\}_4$	$\{5,11\}_4$		$\{0.5\}_4$
$\{4\}_5$	$\{13,16\}_5$	$\{2,14\}_5$	$\{13,16,18\}_4$	$\{0.5\}_5$
$\{4\}_6$	$\{13,18\}_6$	$\{2,15\}_6$		$\{0.5\}_6$
$\{5\}_7$	$\{13,20\}_7$	$\{3,16\}_7$	$\{13,20\}_5 \rightarrow \varnothing$	$\{0\}_7$
$\{6\}_8$	$\{16,18\}_8$	$\{9,1\}_8$	$\{16,18\}_6 \rightarrow \varnothing$	$\{0\}_8$
$\{7\}_9$	$\{16,20\}_9$	$\{10,2\}_9$	$\{16,18,20\}_7 \rightarrow \varnothing$	$\{0\}_9$
$\{7\}_{10}$	$\{18,20\}_{10}$	$\{10,1\}_{10}$		$\{0\}_{10}$

Tab. 2: Procedure to disentangle the multiple short circuit measured among the 5 pads of Figure 10. The white boxes mark the minimum set of MCC-bundles indicating that at least 2 defects have caused this multiple short.

4 Evaluation of Open Circuits

To explain the principle to extract defects that result in open circuits, it is worth to model a test structure using the geometry graph introduced by [HeWe94] which is summarized in Table 3.

layout elements to detect defects	meandrous lines or strings of vias (e. g. [IpSa77], [Bueh83], [LYWM86])
test structure and model of test structure	 1 inputs 2 3 4 5 6 outputs 1 4 2 5 3 6 Fig. 11: Left: Known test chip arrangement. Right: **Geometry graph** (model).
nodes of graph	**pads**
edges of graph	**conductive component (CC)** (=possible open circuits): Two nodes are connected by an edge if layout elements are connected to each other and placed between the two pads by the designer.

Tab. 3: Geometry graph to model a test structure.

An undesigned open circuit defect is only detectable on an implemented conductive component (CC) measuring the resistance between its two pads. But, in case of a measured open circuit it is generally impossible to determine how many defects have interrupted the conductive component between the pads. So, open circuits are always **single faults**, but they are the result of either a **single defect** or more than just one defect then called a **multiple defect**. To separate single defects and multiple defects it is necessary to implement a large number of CCs so that each CC just covers a small portion of the total test structure area. But, the number of different CCs is limited to the number of pads.

In contrast to the neighborhood graph, there is no possibility to increase the number of distinguishable CCs without either increasing the number of pads or using directed CCs which results in the so-called geometry digraph [HeWe94a] as can be seen on the left side of Figure 12. Directed CCs should restrict the measurement current to flow in one direction only. If diodes are available in addition to the backend polysilicon and metal layers, a diode array increases the number of distinguishable CCs for a given number of pads. In this case each diode stands for a distinguishable serpentine line or a via (or contact) string, respectively.

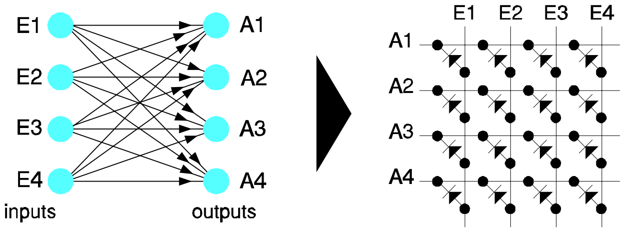


Fig. 12: Geometry digraph and diode array to detect defects that result in open circuits.

So, test structure designs based on a diode array reduce the area covered by a single CC and therefore even multiple defects will be measurable as a distinguishable set of single faults per defect (ref. [WWGH92], [HeWe94a]).

5 Experimental Results

Over the past years different test structure designs including Checkerboard Test Structures and Harp Test Structures were manufactured at several fabs. The following Figure 13 shows the percentage of single faults (k=2) and multiple faults (k>2) dependent on the ability to evaluate an unambiguous set of defect locations within the test structures. Within the Checkerboard Test Structures "A25", "A26", "D13", and "C11", up to 25% of the faults occur as multiple faults. About 2/3 of them may be assigned to an unambiguous set of defect locations. Within the Harp Test Structure more than 45% of the faults occur as multiple faults, because the design provokes multiple faults to determine the defect size between connected parallel lines [HeWe97a]. More than 90% of these multiple faults may clearly be disentangled because most of them are caused by a single large defect.

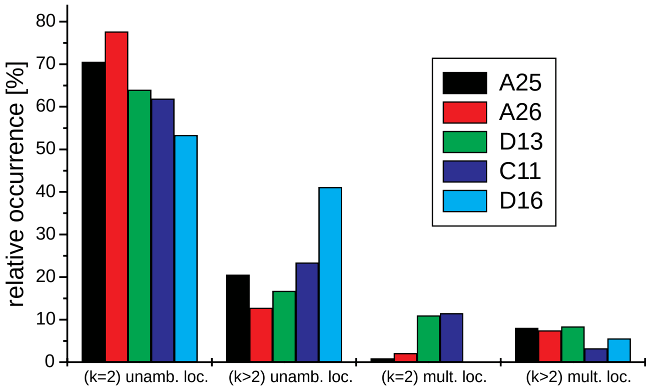


Fig. 13: Relative occurrence of single and multiple faults dependent on different test structure designs.

For the Checkerboard Test Structure design "C11", the following Figure 14 gives the distribution of defects and faults dependent on the type of fault. As mentioned in Section 4, the number of open circuits is identical to the number of defects that have caused them. Also, all single short circuits (*k*≤*2*) are the result of a single defect each. More than 80% of them are unambiguously locatable. We also get some single short circuits where just one MCC is involved (*k=1*) which is the result of connections to permanent power supply. There are multiple possible locations of those faults and defects, because a single MCC is implemented in several subchips (ref. dashed lines of Fig. 10). Some single short circuits (*k=2*) connecting MCCs coincidently placed in adjacent layers will result in the same problem. Multiple short circuits (*k>2*) are caused by an average of two defects per fault. So, the actual number of defects within a test structure is higher than the number of measured faults which has a detrimental impact on the determination of defect densities, if multiple faults are not disentangled.

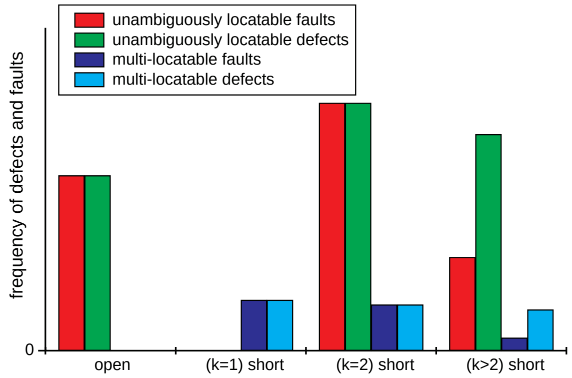


Fig. 14: Frequency of defects and faults in test structure design "C11".

Finally, Figure 15 gives the relative occurrence of defects within the different test structure designs after disentangling all detected multiple faults. Up to 40% of unambiguously locatable defects result in multiple faults. Comparing these data to the chart displayed in Figure 13 indicate the impact of multiple faults and multiple defects on the evaluation of defect densities. If test structures are used, that do not support the separation and localization of defects, both charts would result in nearly identical density distributions, which does not correspond to the actual occurrence of defects within semiconductor manufacturing processes.

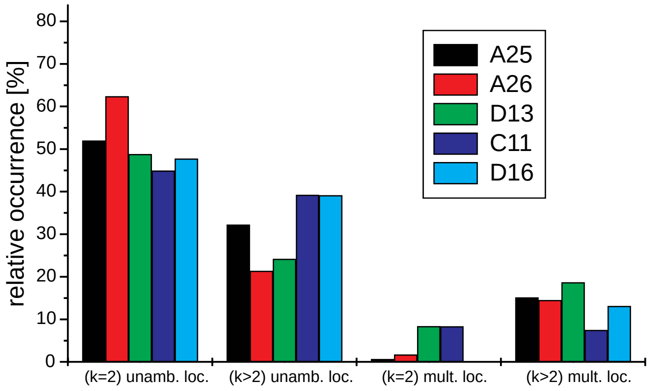


Fig. 15: Relative occurrence of defects dependent on single and multiple faults as well as the test structure design.

6 Conclusion

Generally, errors in yield prediction are assigned to wrong critical area calculations and faulty selections of the models to describe defect densities. The occurrence of multiple faults that are the result of more than just one defect is the reason, why defect density distributions based on just counting the number of faults are insufficient. Our results clearly show, that the number of defects and therefore also the defect density is to small, if multiple faults are not disentangled, which may also be one reason for trouble in yield prediction.

To handle this problem, one way is to design test structures that prevent the occurrence of multiple faults or at least reduce the probability that they occur. A second way will analyze measured data of multiple faults using methodologies to calculate and compare the probability of possible locations of defects within a test structure layout. To determine the minimum number of defects that have caused a multiple fault, we describe procedures how to weight and count these defects to finally provide layer specific defect densities that correspond to the known product model for yield prediction. These procedures may be adapted to any test structure design that enables the localization of defects like SRAM, Checkerboard Test Structures, and Harp Test Structures.

Acknowledgment

Parts of this research were supported by *Deutsche Forschungsgemeinschaft* (DFG), Schm 623/3.

References

[Bueh83] Buehler, M. G.
Microelectronic Test Chips for VLSI Electronics
VLSI Electronics Microstructure Science, pp. 529-576, Vol. 9, Chap.9, Academic Press, 1983

[Hess94] Hess, C.
Strategy to Optimize the Development, Use, and Dimension of Test Structures to Control Defect Appearance in Backend Process Steps
Proc. Advanced Semiconductor Manufacturing Conference (ASMC), pp. 282-289, Boston (USA), 1994

[HeSt94] Hess, C., Ströle, A.
Modeling of Real Defect Outlines and Defect Parameter Extraction Using a Checkerboard Test Structure to Localize Defects
IEEE Transactions on Semiconductor Manufacturing, pp. 284-292, Vol. 7, No. 3, 1994

[HeWB97a] Hess, C., Weiland, L. H., Bornefeld, R.
Customized Checkerboard Test Structures to Localize Interconnection Point Defects
Proc. VLSI Multilevel Interconnection Conference (VMIC), pp. 163-168, Santa Clara (USA), 1997

[HeWe94a] Hess, C., Weiland, L. H.
Drop in Process Control Checkerboard Test Structure for Efficient Online Process Characterization and Defect Problem Debugging
Proc. ICMTS, pp. 152-159, San Diego (USA), 1994

[HeWe94] Hess, C., Weiland, L. H.
Modeling of Test Structures for Efficient Online Defect Monitoring Using a Digital Tester
Proc. ICMTS, pp. 108-113, San Diego (USA), 1994

[HeWe95b] Hess, C., Weiland, L. H.
Defect Parameter Extraction in Backend Process Steps using a Multilayer Checkerboard Test Structure
Proc. ICMTS, pp. 51-56, Nara (Japan), 1995

[HeWe97a] Hess, C., Weiland, L. H.
Determination of Defect Size Distributions Based on Electrical Measurements at a Novel Harp Test Structure
Proc. ICMTS, pp. 1-6, Monterey (USA), 1997

[IpSa77] Ipri, A. C., Sarace, J. C.
Integrated Circuit Process and Design Rule Evaluation Techniques
RCA Review, pp. 323-350, Volume 38, Number 3, September 1977

[KhMT94] Khare, J. B., Maly, W., Thomas, M. E.
Extraction of Defect Size Distributions in an IC Layer Using Test Structure Data
IEEE Transactions on Semiconductor Manufacturing, pp. 354-368, Vol. 7, No. 3, 1994

[KMGS94] Khare, J., Maly, W., Griep, S., Schmitt-Landsiedel, D.
SRAM-based Extraction of Defect Characteristics
Proc. ICMTS, San Diego, USA, 1994

[LYWM86] Lukaszek, W., Yarbrough, W., Walker, T., Meindl J.
CMOS Test Chip Design for Process Problem Debugging and Yield Prediction Experiments
Solid State Technology, pp. 87-92, March 1986

[StRo95] Staper, C. H., Rosner, R. J.
Integrated Circuit Yield Management and Yield Analysis: Development and Implementation
IEEE Transactions on Semiconductor Manufacturing, pp. 95-102, Vol. 8, No. 2, 199

[WWGH92] Ward, D., Walton, A. J., Gammie, W. G., Holwill, R. J.
The Use of a Digital Multiplexer to Reduce Process Control Chip Pad Count
Proc. ICMTS, San Diego (USA), 1992