Intra-Cell Defects Diagnosis

Zhenzhou Sun, Alberto Bosio, Luigi Dilillo, Patrick Girard, Serge Pravossoudovitch, Arnaud Virazel, Etienne Auvray

J Electron Test (2014) 30:541–555
DOI 10.1007/s10836-014-5481-5

Intra-Cell Defects Diagnosis

Z. Sun · A. Bosio · L. Dilillo · P. Girard · S. Pravossoudovich · A. Virazel · E. Auvray

Received: 29 April 2014 / Accepted: 2 September 2014 / Published online: 30 September 2014

Abstract The diagnosis is the process of isolating possible sources of observed failures in a defective circuit. Today, manufacturing defects appear not only in the cell interconnection, but also inside the cell itself (intra-cell defect). State of the art diagnosis approaches can identify the defect location at gate level (i.e., one or more standard cells and/or interconnections can be provided as possible defect location). Some approaches have been developed to target the intra-cell defects. In this paper, we propose an intra-cell diagnosis method based on the "Effect-Cause" paradigm aiming at locating the root cause of the observed failures inside a logic cell. It is based on the Critical Path Tracing (CPT) here applied at transistor level. The main characteristic of our approach is that it exploits the analysis of the faulty behavior induced by the actual defect. In other word, we locate the defect by simply analyzing the effect induced by the defect itself. The advantage is the fact that we are defect independent (i.e., we do not have to explicitly consider the type and the size of the defect). Moreover, since the complexity of a single cell in terms of transistor number is low, the proposed intra-cell diagnosis approach requires a negligible computational time. The efficiency of the proposed approach has been evaluated by means of experimental results carried out on both simulations-based and industrial silicon data case studies.

Keywords Intra-cell defects · Diagnosis · Test · Faulty behaviors

Responsible Editor: S. Blanton

This paper is an extended version of previously published paper. Main contributions of this paper with respect to [17, 18] are:
• A complete analysis of the faulty behavior induced by intra-cell defects reported in Section 2;
•Delay faulty behaviors are considered in Section 3;
•A comprehensive set of experiments is reported in Section 4 based on both simulation and industrial silicon data.

Z. Sun · A. Bosio (✉) · L. Dilillo · P. Girard · S. Pravossoudovich · A. Virazel
Laboratoire d'Informatique de Robotique et de Microélectronique de Montpellier, Université Montpellier II / CNRS, 161 rue Ada, Cedex 5, 34392 Montpellier, France
e-mail: Alberto.Bosio@lirmm.fr

Z. Sun
e-mail: sun@lirmm.fr

L. Dilillo
e-mail: dilillo@lirmm.fr

P. Girard
e-mail: girard@lirmm.fr

S. Pravossoudovich
e-mail: pravo@lirmm.fr

A. Virazel
e-mail: virazel@lirmm.fr

E. Auvray
ST Microelectronic Grenoble, Grenoble, France
e-mail: etienne.auvray@st.com

1 Introduction

The ever-increasing growth of the semiconductor market results in an increasing complexity of digital circuits. Smaller, faster, cheaper and low-power consumption are the main challenges in semiconductor industry. The reduction of transistor size and the latest packaging technology (i.e., System-On-a-Chip, System-In-Package, Trough Silicon Via 3D Integrated Circuits) allows the semiconductor industry to satisfy the latest challenges. Although producing such advanced circuits can benefit users, the manufacturing process is becoming finer and denser, making chips more and more prone to defects.

Physical defects like shorts and opens will occur during each step of the fabrication process. These defects can be randomly caused by contaminations or due to systematic

process-design interaction [12]. In modern deep submicron technologies, systematic defects are becoming more likely to appear than random defects. This is caused by the reduced chip sizes, the use of new complex process technologies, new materials and the increasing number of vias and contacts [20]. Today, systematic defects appear not only in the cell interconnection, but also inside the cell itself (intra-cell defect). In the literature, existing works prove that these defects can escape classical test solutions. In [6] a statistic carried out over 1 million tested devices shown that a significant number of defects appear inside the standard cell (i.e., intra-cell defects). [8, 9] show that those defects cannot be detected by using the approaches based on classical fault models (i.e., stuck-at fault model, transition fault model, bridging fault model) [4] or other approaches like N-detect based test and gate-exhaustive based test [10].

The test is one of the most critical tasks in the semiconductor production process. It is not only necessary to seek for fault free devices but it plays a key role in analyzing defects in the manufacturing process as well. The feedbacks derived from the test process are the only way to analyze and isolate many of the defects in today's processes enabling to obtain a fast and efficient yield ramp-up.

Fault Diagnosis plays a crucial role in this scenario, since test can only provide information on the system behavior (good/no good). Fault diagnosis starts from the test response with the aim to locate the faulty part of the circuit and then identify which is the source of the observed failures. Unraveling the location and cause of the defect helps to improve both the circuit design and the manufacturing process, thus leading to a lower cost, an improved yield and a shorter time-to-market. State of the art fault diagnosis approaches can identify the defect location at gate level (i.e., one or more standard cells and/or inter-connections can be provided as possible defect location) [2, 5]. The fault diagnosis results (i.e., possible defect locations) are then further used in defect analysis, where the circuit is physically examined to determine the mechanism of the failure. Physical Failure Analysis (PFA) is physical analysis that corresponds to the physical identification of the defect. It mainly consists in selective de-layering and cross-sectioning of a die. Different types of imaging tools, as Focused Ion Beam (FIB [14]) etching, are employed for this step. PFA is not only crucial and time-consuming but also destructive and irreversible [19]. Therefore, a preliminary diagnosis procedure is mandatory to correctly guide the PFA to eventually save time and increase success rate.

As previously discussed, state of the art fault diagnosis approaches are able to locate the possible defect at gate level (i.e., inter-cell). Therefore, in the case of circuit affected by intra-cell defects, results of inter-cell fault diagnosis may cause problems and impact the efficient of the PFA. So that, PFA may take more time to identify the actual defects. Moreover, in the worst case, PFA may fail (i.e., it does not identify the root cause of the observed error) and it can destroy the circuit. The solution is to develop an efficient diagnosis approach able to target intra-cell defects.

To the best of our knowledge, commercial tools only target inter-cell fault diagnosis, while in the literature some research works already addressed the intra-cell fault diagnosis problem [1, 7, 13].

The approach of [13] is based on the use of a defect dictionary. The dictionary is created by means of defect injection campaign at transistor level. During diagnosis, the observed failures are used to search in to the defect dictionary the most suitable defect location and type. However, the need of pre-computed defect dictionary for each cell and defect type makes this approach highly complex. Moreover, if the injected defect is not accurate enough it can lead to erroneous results during diagnosis.

The approach of [1] is based on the use of a fault dictionary. The main difference w.r.t. [13] is that the dictionary is created exploiting a switch-level simulation (i.e., the transistors are considered as switches). Thus, instead of injecting defects, only fault models are injected. The advantage is the reduced simulation time compared to [13]. Two types of fault are considered, the stuck-at and dominant bridging fault. These faults are modeled at switch-level in order to be simulated and to create the related fault dictionary. However, defects leading to delay faults are not targeted. Moreover, to include delay faults or other types of faults a switch-level model of them has to be defined.

The approach of [7] proposes to convert transistor level netlist into an equivalent gate level netlist. Then classical inter-cell fault diagnosis tools can be applied. The main drawback of this approach is that the set of transformation rules depends on targeted defect types.

Due to the drawbacks and limitations of previous works on intra-cell diagnosis, it is necessary to develop an efficient and accurate intra-cell diagnosis solution to ensure the PFA success rate. In this work we propose a new intra-cell diagnosis approach able to provide accurate defect localization in order to improve the efficiency of PFA and to eventually save cost and time. The main characteristic of our approach is that it exploits the analysis of the faulty behavior induced by the actual defect. In other word, we locate the defect by simply analyzing the effect induced by the defect itself. Thus, conversely to previous work on intra-cell diagnosis [13], there is no need to characterize the library to create a defect dictionary. The faulty behavior analysis is performed by applying the Critical Path Tracing (CPT) at transistor level. Compared to [1, 7] there is no need to pre-compute any fault dictionary and to convert the netlist.

The paper is organized as follows: Section 2 gives the basics of defects and fault modeling that is the base of the proposed approach. Section 3 depicts the overall diagnosis

flow and then it details each step of it. Section 4 reports the validation carried out on several experiments while Section 5 concludes the paper.

2 From Defects to Fault Models

In this section, we present the main causes of physical defects and how these defects are usually modeled at transistor-level domain. Then, we discuss about the faulty behavior induced by defects and how the fault models represent them. This analysis is important in order to give the basics on how the proposed approach is defect independent while targeting transistor-level circuit descriptions.

A physical defect can be caused by several phenomena such as metal line broken/deformed, contact or via broken/deformed. These phenomena will lead to either an unexpected connection between two or more nets (short) or a missing connection on one net (open) [4]. A net, at transistor-level domain, correspond to different elements: polysilicon (the transistor gate terminal), active (the transistor drain and source terminals), metal line (interconnections between transistors), contact (connection trough active and metal level 1) or via (connections trough metal levels). Usually, these defects are modeled at transistor-level domain as: (i) an unexpected connection between two nets associated to a specific resistance value (resistive-short), (ii) an unexpected resistance value on a given net (resistive-open) [4]. Depending on the considered resistance value, the effect induced by the defect can vary. Thus, the choice of meaningful resistance values is crucial to have an accurate model of physical defects. As discussed in the previous section, existing works on intra-cell diagnosis rely on this way to model defects to create the defect dictionary.

Figure 1a) shows an example of physical defects and the related model at transistor-level domain for a complex gate composed of four primary inputs (A, B, C and D) and one primary output (Z). In the transistor-level netlist four defects are highlighted in red. Please note that the four defects are only an example, the proposed approach can target any possible defects.

Defect D1 models an unexpected connection between net118 and ground. The behavior of this defect depends on the resistance value R1. If R1 is lower than a threshold value RT, then the Vgs (voltage level between gate and source terminal of transistor T7) is always lower than Vth of T1. Thus T1 remains switched off (faulty behavior). If R1 is grater than RT, then Vgs depends on the voltage level of Net118 (correct behavior). From a logical point of view the first case (when R1 < RT) is equivalent to have Net118 stuck at logic value “0”. A similar consideration can be done for defect D2. Depending on R2 value, it impacts the voltage level of net88. From a logical point of view it is equivalent to have net88 stuck at logic value “1”.

Defect D3 models an unexpected connection between net110 and net106. The behavior of this defect depends on the resistance value R3 and also on the applied stimuli (i.e., it can activated or not). For example Fig. 1b) gives the circuit when the stimuli “0111” are applied to inputs ABCD respectively. Transistors T8 and T10 are switched-on while T9 is switched-off and output Z is set to logic ‘1’. However, due to the presence of this defect, the output is now connected to the ground through R3. Thus the defect is activated. Once the defect is activated we analyze the R3 value to determine its behavior. If R3 is lower than R_{min} then, the output will be set

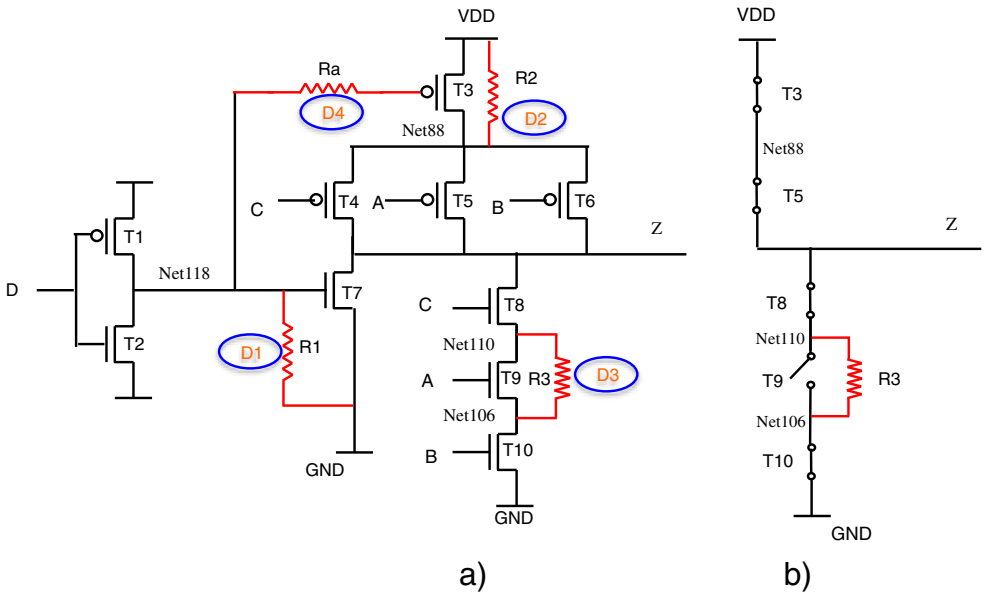

Fig. 1 Physical defects modeling at transistor-level domain b) Equivalent circuit when the stimuli “0111” are applied

to logic value '0'. In this case, from a logic point of view the Net106 force its logic value '0' to the Net110 and then the output is changed. On the other hand, if R3 is greater than R_{min} but lower than R_{max} ($R_{min} < R3 < R_{max}$) then, output Z will be affected by an undesired transition (from logic '1' to '0') due to the long discharge time. Finally, if R3 is grater than R_{max} then output Z will take the correct logic value and the circuit behavior is faulty free. Note that the value of R_{min} and R_{max} depends on the circuit technology.

Finally, defect D4 affects the net Net118. For certain values of R4 the signal propagation trough gate of transistor T4 is delayed.

To summarize, the analyzed faulty behavior induced by the four defects of Fig. 1a) are:

- D1, D2: the faulty behavior results in a net always set to a given logic value (either '1' or '0');
- D3: the faulty behavior results in a net set to a given logic value (in the example logic '0") depending on the input configuration or in a signal propagation delay affecting primary output Z;
- D4: the faulty behavior results in net where the signal propagation is affected by a given delay.

Even if we target the transistor-level domain, we exploit the knowledge of the analyzed faulty behavior to be defect independent. Thus, we can avoid to explicitly considering the resistance value. We will show in the next section that the proposed intra-cell approach identifies the possible locations of a defect.

Usually faulty behaviors induce by defects are represented by means of fault models. From above the example we can list the exploited fault models:

- Stuck-at fault model [4]: the logic value of a given net appears to be stuck at a constant logic value ('0' or '1'), referred as stuck-at-0 or stuck-at-1 respectively (e.g., defects D1 and D2 in Fig. 1).
- Dominant Bridging fault model [4]: this fault involves two nets called aggressor and victim. The logic value of the victim is set to the logic value of the aggressor (e.g., defect D3 in Fig. 1).
- Delay fault model [4]: the transition from a given logic value V to the opposite logic value !V is delayed. Two types of delay faults are defined: slow-to-rise transition fault model (slow transition from logic '0' to logic '1') and slow-to-fall transition fault model (slow transition from logic '1' to logic '0').

Since the proposed diagnosis approach provides possible defect locations, for each of them we will associate one or more fault models according to the observed faulty behavior. In the next sections, we detail the proposed intra-cell diagnosis approach.

3 Intra-Cell Diagnosis Flow

In this section, we present the whole intra-cell logic diagnosis approach. It is able to locate the root cause of observed failures inside a logic cell. Since the proposed approach works at transistor-level, it cannot be applied to the whole circuit due to its complexity (i.e., billions of transistors). However, it can be easily applied to a single target logic cell (i.e., up to fifty transistors) identified by a logic-level diagnosis tool.

Figure 2 sketches the overall diagnosis flow. First of all, the *test* applies test patterns to the DUT (Device Under Test) to distinguish between the correct circuit behavior and the faulty circuit behavior caused by defects. These defects induce failing output responses for one or more input test patterns. Input test patterns leading to observed faulty behavior are called failing test patterns and stored in to a file called datalog. Input patterns for which no faulty behavior is observed are called passing test patterns. Then, the *inter-cell fault diagnosis* exploits datalog information to determine a list of suspected logic cells (i.e., candidates). Any available commercial diagnosis tool can be adopted. For each suspected cell, we have to know logical values applied to it when failing and passing test patterns are applied to the DUT (i.e., DUT simulation). *DUT simulation* aims at determining the local set of failing/passing patterns for each suspected logic cell reported by inter-cell diagnosis. Finally, the *intra-cell diagnosis* is executed for each Suspected Gate (SG) and the pre-determined local failing/passing test patterns set (*lfp* and *lpp*). The intra-cell diagnosis result is a list of candidates at transistor level. For each suspected net a set of fault models able to explain observed failures is associated. We present in details the main steps in next subsections.

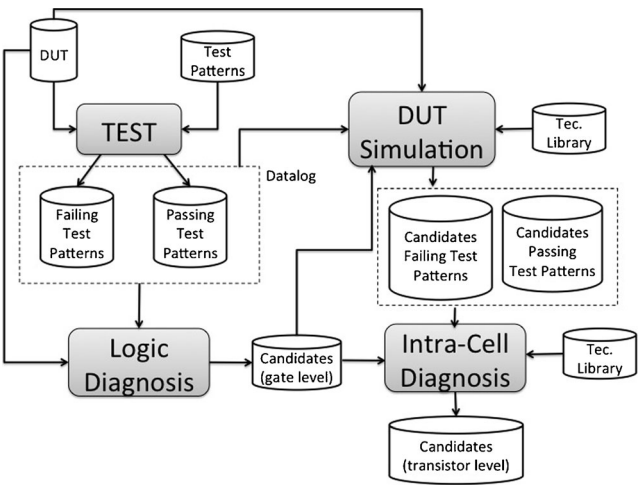

Fig. 2 Overall diagnosis flow

3.1 DUT Simulation

The intra-cell diagnosis is applied on a single candidate identified as the Suspected Gate (SG). The preliminary step of the proposed intra-cell diagnosis approach aims at determining the local failing/passing patterns defined as *lfp* and *lpp* respectively.

Figure 3 shows the SG located in the circuit. When a failing test pattern *fp* is applied to the circuit PIs, the fault affecting the SG is sensitized and its effect is then propagated to at least one circuit PO. This is guaranteed by the fact that during the test a failure is observed when the *fp* is applied to the circuit. Since the intra-cell diagnosis is applied to the SG only, we have to known the logical values of the SG inputs (called local patterns) when the *fp* is applied to circuit PIs. Thus, a logic simulation of the *fp* is required to get those values. This simulation has to be performed for each given failing test pattern.

In a similar way, we have to know which are the logical values applied to the SG in the case of passing test patterns. A test pattern does not detect any fault for two reasons: (i) the fault is not sensitized (ii) the fault is sensitized but is effect is not observed because the fault effect cannot be propagated to reach the primary outputs. For the passing test pattern, we have to discriminate the reason why the fault located in the SG has not been detected (i.e., discriminate between (i) and (ii)). Thus, we have to verify if the fault effect can be propagated or not. If yes, then the pattern could be considered as a local passing test pattern because any failures should be observed during the test.

However, one more consideration must be done. In section 2 we described the faulty behavior considered in this work. Some of them depend only on the local gate input values (i.e., stuck-at and bridging faults). Some others depend not only on the local gate input values but also on the previous local values (i.e., delay faults).

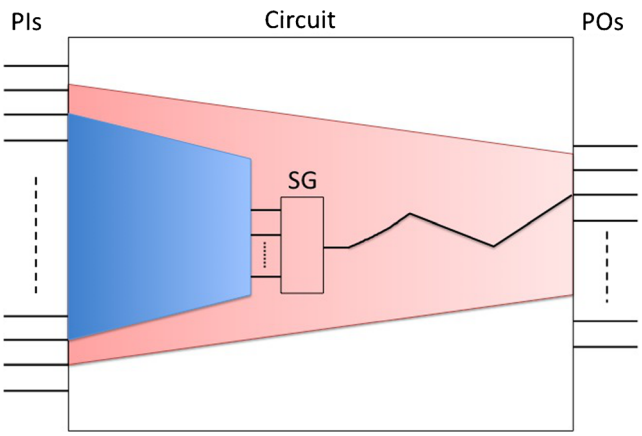

Fig. 3 Local Failing/Passing test patterns

Taking in to account this consideration, we can now classify a given *lpp* as follows:

- If we assume that the defect affecting the suspected gate SG leads to a static faulty behavior (i.e., stuck-at, bridging) then, *lpp* is classified as local passing pattern;
- If we assume that the defect affecting the suspected gate SG leads to a dynamic faulty behavior (i.e., delay) then, *lpp* can not be classified as local passing test pattern, because we do not consider the previous pattern.

At this stage of the diagnosis flow, we must consider as valid both the above assumption. Therefore, we will store the *lfp* and *lpp* in to different data structure associated to the static faulty behavior and to the dynamic faulty behavior respectively.

Finally, for the case of defects leading to a dynamic faulty behavior, a sequence of test patterns has to be applied to detect the defects. It may be possible that the same local pattern could be declared falling and passing. For this case, we knew that the defect affecting the circuit is a dynamic type so that only delay fault will be targeted thus discarding both stuck-at and bridging faults. The analysis performed to determine local failing and passing test patterns leads to the taxonomy shown in Fig. 4.

After the test, patterns can be classified into two categories: failing and passing test patterns.

Definition 1 Failing test patterns are used to determine local failing test patterns. Zone 2 in Fig. 4 shows this type of test patterns.

Definition 2 Passing test patterns are used to determine local passing test patterns. The passing test patterns illustrated in zone 1 of Fig. 4.

Definition 3 When at least one local test pattern is declared at the same time failing and passing ($lfp \cap lpp \neq \varnothing$) as shown in the zone 3 of Fig. 4, it means that the defect affecting the circuit leads to a dynamic faulty behavior. In this case, we can discard the static faulty behavior (i.e., static and bridging) to be the root cause of observed failures.

Definition 4 If $lfp \cap lpp = \varnothing$, then defect affecting the circuit can lead either to a dynamic or static faulty behavior. In this case, we must consider both static and dynamic faulty behavior can be to be the root cause of observed failures.

In the next section we will describe in detail the applied intra-cell diagnosis approach and how the information extracted during DUT simulation are exploited.

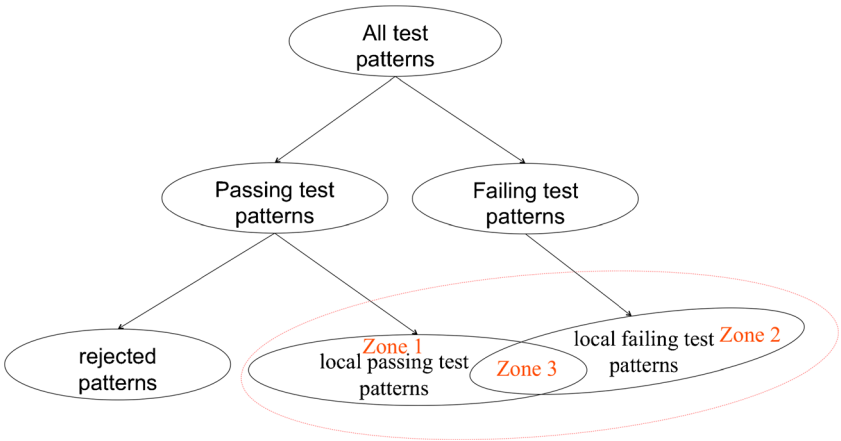

Fig. 4 failing and passing local test patterns taxonomy

3.2 Effect-Cause Intra-Cell Diagnosis Algorithm

In this section we present the proposed intra-cell diagnosis approach. It is based on the "Effect-Cause" paradigm and it exploits the Critical Path Tracing (CPT) algorithm [2] here applied at transistor level. We first present the classical CPT algorithm (applied at logic level). Then, we show the CPT implementation at transistor level. Finally we give the intra-cell diagnosis procedure based on the CPT.

3.2.1 Critical Path Tracing

At gate level, the CPT starts from a failing primary output. It traces back critical nets passing trough critical gate inputs, and finally it stops when it reaches the primary inputs. A conceptual view of the CPT application is given in Fig. 5. A gate inputs *Gi* is critical if the logic inversion of *Gi* value changes the gate output value. After the CPT, each critical net could be the possible root cause of observed failures.

In our work the CPT is applied to a transistor level netlist, thus instead to deal with logic gates, we have to consider a network of transistors. Let us first define how the transistor's terminals can be declared critical or not. At transistor level domain, the critical net determination is similar than the one at gate-level domain. The transistor can be modeled as a switch in digital circuits. It consists of three terminals: *source*, *gate* and *drain*. Depending on the gate logic value the source terminal is connected or not to the drain terminal. Thus, drain and source terminals are set at the same logic value. When the MOS transistors are used to build logic cell, then the logic value is transferred from the source to the drain controlled by the gate. In other words, the source and gate can be considered as inputs and the drain can be considered as output of the transistor.

In Fig. 6 we show a transistor level netlist of the library cell AO8DHVTX1. It is a standard cell in a 90 nm technology library. It is composed of 10 transistors, 4 inputs (A, B, C and D) and one primary output (Z). We resort to this netlist in order to give a full

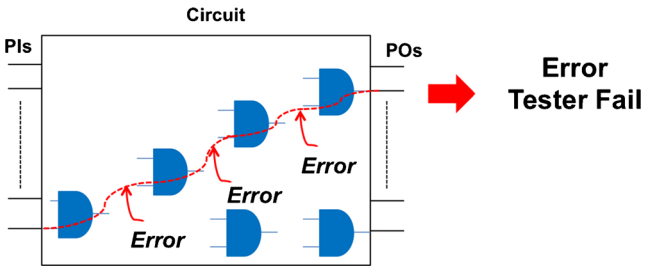

Fig. 5 Gate level CPT application

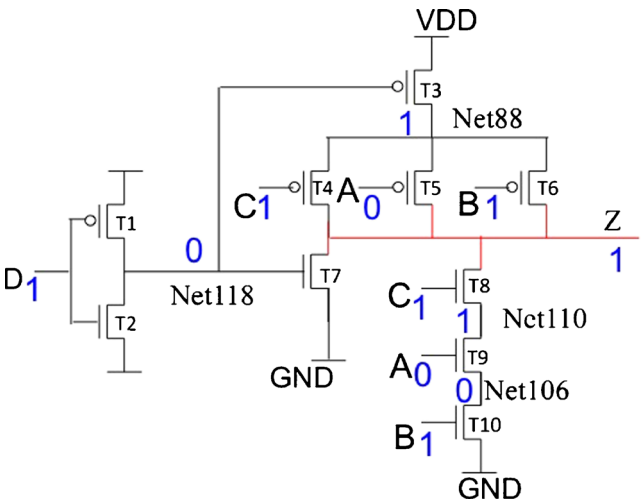

Fig. 6 Transistor-level domain CPT application example (1)

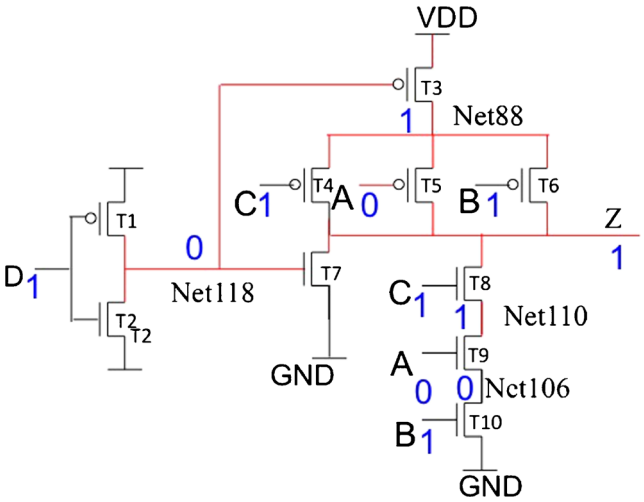

Fig. 7 Transistor-level domain CPT application example (2)

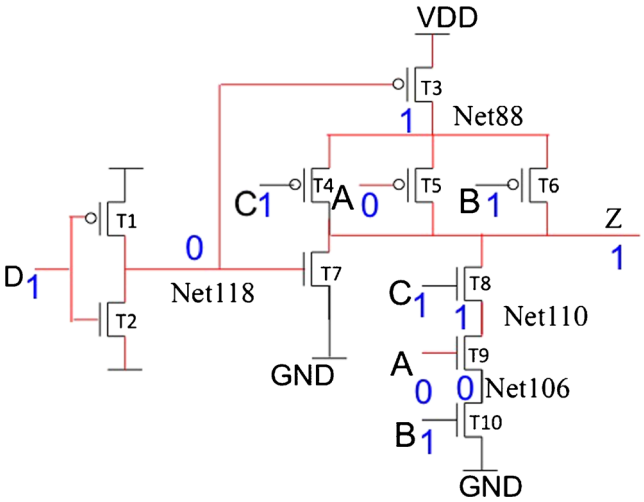

Fig. 8 Transistor-level domain CPT application example (3)

example about the CPT application. During the example we define T*x*D, T*x*S and T*x*G as the drain, source and gate of the transistor *x* respectively. In the example, the logic values carried by each net are shown in blue.

For a given input vector "0111", the CPT starts from the output Z to reach the inputs by tracing each critical net. The output Z is first of all set as critical net (highlighted in red) in Fig. 6.

The Z net is connected to 5 transistors (T4, T5, T6, T7, T8) drain terminals, so that the drain terminals of these transistors are set to be critical (T4D, T5D, T6D, T7D, T8D). If one of these terminal value changes, then also the output Z changes too. The next step is determining for each of these transistors if their gate and source terminals are critical. Transistors T4 and T6 are in the same configuration (i.e., the same logic values applied to the terminals). Inverting gate terminal does not change the output, thus T4G and T6G are not critical nets. The same consideration is done for the source terminals (i.e., T4S and T6S are not critical). Conversely, terminals T5S, T5G, T7G and T8S are critical as shown in Fig. 6. Please note that when one terminal is critical then the net to which the critical terminal is connected will also be set to critical. So that, the *Net88* and *Net118* are both set to critical nets as shown in Fig. 7.

We continue to trace back the nets. The next transistors to be analyzed are T1, T2, T3 and T9. For the same reason illustrated before, T1G, T2G, T2S, T3G, T3S and T9G are critical terminals as shown in the Fig. 8.

After this step, the CPT stops because it reaches the primary inputs. The determined critical nets and transistor terminals are stored in the following critical nets list: {Z, T4D, T5D, T6D, T7D, T8D, T5S, T5G, T7G, T8S, Net88, Net118, Net110, T1G, T2G, T2S, T3G, T3S, T9G, Input A, Input D}.

The next sub-section discusses how the CPT is exploited during the intra-cell diagnosis and how the information related to the critical nets are used.

3.2.2 Intra-Cell Diagnosis Algorithm

The proposed intra-cell diagnosis approach requires two main inputs: the SG description at transistor level and the local set of failing/passing test patterns. Figure 9 gives the pseudo code of the intra-cell diagnosis procedure.

The procedure is divided in two blocks identified by the two "for loop" statements in Fig. 9. The first one targets the *lfp*. For each *lfp*, a fault-free simulation is performed by

```
Intra_cell_diagnosis (SG, lfp, lpp) {

1  GSL = GDSL = all SG nets and transistor terminals
2  GBSL = all SG couple of nets
3  For each lfp {
4     fault_free_simulation (SG, lfp)
5     CPT (SG, lfp, CSL, BSL, DSL)
6     GSL = GSL ∩  CSL
7     GBSL = GBSL ∩  CBSL
8     GDSL = GDSL  ∩ CDSL
9  }
10 For each lpp {
11    fault_free_simulation (SG, lpp)
12    CPT (SG, lpp, CVL, CBVL)
13    GSL = GSL \ CVL
14    GBSL = GBSL \ CBVL
15 }
16 fault allocation (GSL, GBSL, GDSL)
17 }
```

Fig. 9 Intra-cell diagnosis pseudo code

using a switch-level simulation. In the switch-level simulation, the transistors (i.e., nMOS and pMOS) behave as on-off switches. All the detail about this type of simulation can be found in [3].

Since the simulated netlist is composed of few transistors the required simulation time is negligible. The fault-free simulation is mandatory to determine the logic value of each net. Then, Critical Patch Tracing (CPT) is executed starting from the SG output. It traces back internal nets to reach the SG inputs. Each traced net is set as critical. A net is critical if the inversion of its logical value causes the inversion of the output value. Each critical net is marked as a suspect. A suspect can be the root cause of observed errors or simply a net that propagate the fault effect. Each suspect is stored in a list with its logic value. The logic value is kept in the list because it will be used during the fault model allocation. The result of the CPT is the Suspect List (SL). The SL is defined as follows:

$$SL = \{(net_0, LV_0), (net_1, LV_1), ...(net_{n-1}, LV_{n-1})\} \quad (1)$$

where:

- net_i: is the critical net (transistor-level interconnection nets and transistor terminals).
- $LV_i = \{0, 1, U\}$: is the logic value of the critical net. U is the unknown value.

As stated before, a given suspect can simply propagate the fault effect (i.e., it is not the root cause of observed errors). This case can happen for two reasons:

1. The suspect net belongs to the propagation path of the actual faulty net. In this case, the suspect is logically equivalent to the actual faulty net.
2. The suspect net is indeed the victim of a bridging fault. In this case, another net (i.e. the aggressor) "forces" a faulty value on the victim.

The second case is more complex than the first one from the diagnosis point of view. Here, we have to verify if a bridging fault is possible. For this reason, after the CPT a second list of suspects is created. The so-called Bridging Suspect List (BSL) contains all the possible couples "Victim/Aggressor" that can be involved in a bridging fault. The victim belongs to the actual SL, while the aggressor can be any net having an inverted logic value w.r.t. the logic value of the victim. The BSL is defined as follows:

$$BSL = \{(V_0/A_0), (V_1/A_1), ...(V_{m-1}/A_{m-1})\} \quad (2)$$

where:

- $V_i \in SL$: is the victim.
- A_i (Net_j, LV_j): is the aggressor. It can be any net of the gate having a logic value opposite to the logic value of the victim. Thus LV_j LV_i.

So far (eqs. 1 and 2), we defined as critical nets for which the inversion of their logical value causes the inversion of the output. With these two lists, we are able to identify any static faults (see Section 2). To complete our analysis, we have also to consider the case of dynamic faults affecting the analyzed gate and thus the circuit (see Section 2). As for the equation (1) we look for critical nets, where a critical net can be either a transistor terminal or an interconnection net. We define those nets as critical delay nets. Critical delay nets are added in a list called Delay Suspect List (DSL) defined as follows:

$$DSL = \{Net_0, Net_1, \ldots Net_{n-1}\} \quad (3)$$

where:

- Net_i is the critical net (transistor-level interconnection nets and transistor terminals)

Please note that in this case, the logical value of a critical delay net is not stored in the list because it is not used during the fault model allocation. Basically we do not distinguish between a slow-to-rise/slow-to-fall delay fault.

The three lists are stored in a so-called Current Suspect List (CSL), the Current Bridging Suspect List (CBSL) and the Current Delay Suspect List (CDSL).

The assumption of this work is the presence of only one defect on a given circuit. Thus, the root cause of observed errors has to be present in all lists provided by the CPT application. For this reason, we update the global suspect lists by performing an intersection (line 6, 7 and 8 in Fig. 9). The intersection between two suspects lists SL_a and SL_b is defined as follows:

$$\text{A suspect net } SN = (net_i, LV_i) \in SL_a \cap SL_b \text{ if and only if } SN \in SL_a \text{ and } SN \in SL_b. \quad (4)$$

This definition removes a net from the suspect list if the net is traced with different logic values (e.g., once with '0' and another with '1'). This is coherent with the stuck-at fault model. If a net is affected by a stuck-at fault, its value must be always the same during the

failing test patterns application. Otherwise, the net cannot be affected by a stuck-at fault. Thus it is removed from the suspect list.

The intersection between two bridging suspect lists elements $BSLE_a = (net_i, LV_i)/(net_j, LV_j)$ and $BSLE_b = (net_m, LV_m)/(net_n, LV_n)$ is defined as follows:

$$BSLE_a \cap BSLE_b = (net_i, LV_i \cap LV_m)/(net_j, LV_j \cap LV_n) \text{ if and only if } net_{i=} net_m \text{ and } net_{j=} net_n \quad (5)$$

The intersection between logic values is defined in the table of Fig. 10.

The above leads to keep a couple Victim/Aggressor (V/A) even if it appears in two lists with different logic values. Conversely to the case of a stuck-at fault, this case can occur if a strong dominant bridging fault [2] is the root cause of observed failures.

The intersection between two delay suspects lists is similar to the one for SL (suspects lists). The difference is that there is no logic value associated to the delay candidates. The intersection between two delay suspects DSL_a and DSL_b

$$\text{A delay suspect net } DSN = (net_i) \in DSL_a \cap DSL_b \text{ if and only if } DSN \in DSL_a \text{ and } DSN \in DSL_b. \quad (6)$$

The DSN is kept if and only if it belongs to all DSLs. Please note that the logic value is not stored in the DSL (eq. 6), thus, we don't need to consider the logic value of candidates during the DSLs intersection.

The result of the intersection is stored in the so-called Global Suspect List (GSL), the Global Bridging Suspect List (GBSL) and the Global Delay Suspect List (GDSL).

The second block of the intra-cell diagnosis procedure aims at applying the CPT for each *lpp* to vindicate the suspected elements. The main concept behind this block has been already exploited for the inter-cell diagnosis [2]. This step is applied for the GSL and GBSL list but not for the GDSL. This is because we can determine local passing pattern only for the case of static fault, thus we can vindicate only for GSL and GBSL as presented in the section 3.1. As for the first block, we create two suspect lists. The first one contains critical nets called Vindicate List (VL). In this case each critical net is vindicated to be the root cause of the observed failure [2]. The second list contains all possible couple V/A that can be vindicated to be involved in a bridging fault called Bridging Vindicate List (BVL). The victim belongs to the actual VL, while the aggressor can be any net having an opposite logic value w.r.t. to the logic value of the victim.

We compute the difference between the vindicate lists and the suspect lists (lines 13 and 14 of Fig. 9). Owing to the knowledge of the actual passing test patterns it is possible to narrow down the actual list of suspects.

The difference between a suspect list SL and a vindicate list VL is defined as follows:

$$\text{A suspect net } SN = (net_i, LV_i) \in SL \backslash VL \text{ if and only if } SN \in SL \text{ and } SN \notin VL \quad (7)$$

In the same way the difference between a bridging suspect list BSL and a bridging vindicate list BVL is defined as follows:

$$\text{A suspect bridging net } SBN = (net_i, LV_i)/(net_j, LV_j) \in BSL \backslash BVL \text{ if and only if } SN \in SL \text{ and } SN \in VL \quad (8)$$

Finally, the last step of the intra-cell diagnosis is the fault model allocation. Basically for each suspect we exploit the

LV_1 \ LV_2	0	1
0	0	01
1	01	1

Fig. 10 Logic values intersection

Table 1 Circuit Characteristics

Circuit	#Gate	#FlipFlop	#Scan chain
A	258	30	1
B	698804	56373	25

stored logic value to associate a fault model. Once again, three types of fault models are considered: (i) the stuck-at fault, (ii) the dominant bridging fault and (iii) the delay fault. During the fault model allocation, it could be happen that one or several suspected lists are empty. This means that the root cause of observed errors cannot be the fault model corresponding to the empty suspected list. For example, we presented in the section 3.1, when $lfp \cap lpp \neq \varnothing$ only dynamic faults are possible. So that, after suspected lists construction for *lfp* and *lpp*, GSL and GBSL will be empty and we consider only faults in the GDSL.

4 Experimental Results

The proposed intra-cell diagnosis procedure has been implemented in C++. It has been validated by means of simulations and actual silicon data.

The simulation-based validation exploits a defect injection campaign. Physical defects can take forms of missing or extra materials and they are often modeled by open- and short-circuits as presented in [11]. According to data published in [15, 16], physical defects can have a finite resistance value for open- circuits and a non-negligible resistance value for short-circuits. Defects are thus injected into the transistor-level netlist of a given gate of a DUT. Then, by using a spice simulator, the faulty gate is simulated in order to determine its truth table. The truth table is then used as library model, so that the whole faulty circuit is simulated at gate level to emulate the test phase. Observed failures are stored in the failure file (i.e., datalog). Injected defects lead to stuck-at, bridging and delay faults. The silicon-based validation has been carried out on STM circuits declared faulty during the production test.

4.1 Simulation-Based Validation

We used two circuits named A and B to perform the simulation-based validation. The circuits correspond to actual STM products and have been synthesized with a STM 90 nm technology. Table 1 shows the characteristics of the circuits in terms of gates, FFs and scan chains.

For each logic cell in the targeted technology library, we randomly injected one defect at time and we perform spice simulation to characterize the behavior of the faulty logic cell. Injected defects lead to have 3 types of faulty behaviors. The 30 % of them lead to stuck-at faults, the 30 % lead to bridging faults and the remaining 40 % lead to delay faults.

Then, we randomly select one gate in the target circuit and we replace it by the faulty gate. We perform the simulation and we store observed failures, if any, in the datalog. For each datalog we then run a commercial logic diagnosis tool to identify our suspected gate.

The test patterns applied for the experiments have been generated exploiting a commercial ATPG tool. The test sets target transition fault models and the test length is 25 and 500 patterns for circuit A and B respectively. After the DUT simulation step, we got in average 3 local failing patterns and 6 local passing patterns.

Tables 2, 3 and 4 show the achieved intra-cell diagnosis results for stuck-at, bridging and delay faults respectively. The first column reports the suspected gate name (given by the logic diagnosis tool). The second and third column report the number of input and output of the gate. Column 4 shows the gate complexity in terms of internal gate nets. The fifth column gives the actual injected in terms of location and type. Finally, column 6 shows the result of the proposed intra-cell diagnosis approach.

First of all the results show the accuracy of the proposed approach because in all the cases, the injected defect has been correctly identified.

The second comment is related to the resolution. Basically the resolution depends on the injected defect. For the case of defects leading to stuck-at faults (Table 2), two out of five cases have only one stuck-at fault reported and one case has two equivalent faults identified. For the other two cases, the proposed intra-cell diagnosis method pointed out 3 equivalent stuck-at faults. Please note that the diagnosis results contain

Table 2 Stuck-at-Faults Results

Suspected gate	Gate input	Gate output	Complexity	Injected fault	Results
AO7SVTX1	3	1	6	N16 Sa1	N16 Sa1; Input A Sa0
NR3ASVTX1	3	1	7	N022 Sa0	N022 Sa0; N029, Input A Sa1
AO6CHVTX4	3	1	8	N113 Sa0	Input C, N147 Sa1; N113 Sa0
AO8DHVTX1	4	1	9	Input A Sa1	Input A Sa1
AO5NHVTX1	3	1	9	N71 Sa0	N71 Sa0

Table 3 Bridging-Faults Results

Suspected gate	Gate input	Gate output	Complexity	Injected fault	Results
AO7SVTX1	3	1	6	Z-Gc	Z-Gc
AO7NHVTX1	3	1	7	N50-Gc	N50-Gc
AO6CHVTX4	3	1	8	N113-N109	N113-N109, N113-N125
AO5NHVTX1	3	1	9	N55-A	N55-A
AO9SVTX1	5	1	10	N22-N31	N22-N31

also some bridging and delay faults. The average size of suspect stuck-at and bridging faults list is 7.

For the case of defects leading to bridging faults (Table 3), four cases have only one fault identified and the other one case has 2 equivalent faults reported. Moreover, the proposed approach leads to an empty list of suspected stuck-at and delay. This happens because the behavior of a bridging fault cannot be modeled by a stuck-at fault (in most of the cases).

Finally, concerning the resolution of defects leading to delay faults, we have than the best result identifies two suspects, while the worst result has 5 suspects. For example, for the first case AO7NHVTX1, the injected fault is delay fault affecting N2D (the drain of the transistor N2). Intra-cell diagnosis has identified the transistor N2 and P5 as suspects. When a transistor is identified as suspect, all of the three terminals of this transistor are suspected. The proposed approach leads to an empty list of suspected stuck-at and bridging. This happens because the behavior of a delay fault cannot be modeled by a stuck-at and bridging fault (in most of the cases).

To conclude, the CPU time required for the intra-cell diagnosis is lower than 1 sec. The low execution time and the small number of suspect candidates will save time during FA procedure.

In the last part of the simulation-based validation, we run an extensive set of experiments. For each targeted library cell we randomly select in the circuit (circuit B in Table 1) 100 cell instances. For each cell instance we inject 10 random defects for a total of 1000 diagnosis run. The test set is the same than the one used for the previous experiments. Table 5 reports the select library cell name, the input output number and the complexity in the first four columns. Last column gives the average resolution expressed as the average number of candidates provided by the proposed approach. The select cells have different complexity from 6 up to 24 transistors and different number of input. The main objective of this experiment was to validate the efficiency of the approach on more complex cells. In this sense we achieved good results since in average we have about 1 candidate. However, two cases lead to have about 4 candidates. For these cases the reason why the resolution is worst than the others is the few number of inputs (i.e., only two) that corresponds to a limited set of local patterns (i.e., up to four).

Table 4 Delay-Faults Results

Suspected gate	Gate input	Gate output	Complexity	Injected fault	Results
AO7NHVTX1	3	1	6	N2D	N2, P5
AO8DHVTX1	4	1	9	Net118	Z, Net118, D
AO5NHVTX1	3	1	9	N0D	N0, N1, P7, Net55, Z
AO9SVTX1	5	1	10	P4S	Z, Net9, P4

Table 5 Experimental Results

Suspected gate	Gate input	Gate output	Complexity	Resolution
AO7SVTX1	3	1	6	1.5
AO7NHVTX1	3	1	6	2
NR3ASVTX1	3	1	7	1,8
AO6CHVTX4	3	1	8	2.1
AO8DHVTX1	4	1	9	1.7
AO5NHVTX1	3	1	9	1.4
AO9SVTX1	5	1	10	1.2
AN2BHVTX8	2	1	18	4.1
MUX21HVTX6	3	1	24	1.03
ND4ABCHVTX8	4	1	23	1.1
EOHVTX6	2	1	26	3.8
OR4ABCDHVTX4	4	1	14	1.3

4.2 Silicon-Based Validation

To finally prove the efficiency of the proposed approach, we used some actual faulty circuits to highlight the effectiveness of the proposed intra-cell diagnosis approach. All the experiments are carried out on STM actual circuits. Table 6 shows the characteristics of circuits in terms of technology node, gates, FFs, scan chains and test patterns number. Test patterns target stuck-at, transition and bridging faults. After the DUT simulation step we got a number of local passing patterns varying from 5 to 7 depending on the circuit and the applied test set. The number of local failing patterns varies from 2 to 4.

The details of these case studies are given in the following sub sections.

Table 6 Circuit Characteristics

Device	Technology	#Gate	#Flip flop	#Scan chain	#Patterns number
H	CMOS 90 nm	698,804	56,373	25	500
M	CMOS 90 nm	896,417	60,006	219	1055
C	CMOS 55 nm	1,995,419	183,868	43	1000

Table 7 Logic diagnosis vs intra-cell diagnosis vs actual defect

Sample	Logic diag	Intra diag	Actual defect
H1	U2890/Z (sa01)	A-Z, A-B, A-C	A-Z
H2	U280/Z (sa0), U280/C (sa1), U280/D (sa1)	Net61 (sa0), A-Net61	Net61 metal1 bridging with gnd
H3	U280/A (str)	N0S (open)	N0S metal 1 open

4.2.1 Circuit H

Three faulty cases of circuit H were used as case study for the proposed intra-cell diagnosis approach. Table 7 shows the logic diagnosis, intra-cell diagnosis results and the actual defect location with defect mechanism identified after physical analysis.

The first column gives different circuits names and, the second column presents the logic diagnosis results. The third column shows the intra-cell diagnosis results and the last column gives the actual defect location and the defect mechanism. Logic diagnosis gives one logic cell as candidate. Please note that for the case of H2, the logic diagnosis provides 3 candidates. However, the 3 candidates refer to the same cell "U280". Targeting these logic cells, intra-cell diagnosis is applied. For all the cases, the proposed intra-cell diagnosis gives the actual defect. The resolution is quiet good. One case has only one candidate. For the other two cases, there are two and three equivalent faults obtained.

For the sample H1, intra-cell diagnosis was applied to the given suspect gate U2890. After intra-cell diagnosis, 3 bridges inside the gate were identified as suspects (Table 7). i) Bridge between Input A and output Z, ii) bridge between Input A and Input B and iii) bridge between Input A and Input C. Input A is always the aggressor for every case. During Physical Failure Analysis (PFA), a FIB cross section is performed to verify the intra-cell diagnosis result. Figure 11 gives the photo showing the bridge defect identified by PFA. The actual defect is the bridge between input A and output Z. This result proves the effectiveness of our intra-cell diagnosis approach and the proposed diagnosis flow.

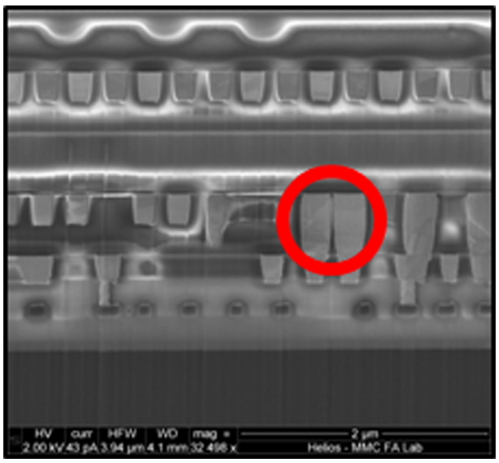

Fig. 11 Physical failure analysis result

For the sample H2, logic diagnosis identifies three nets leading to one suspect cell (Table 7). On the suspect gate the intra-cell diagnosis gives 2 suspects: Net61 stuck-at 0 and bridge between Net61 and Input A. The actual defect is the net61 shorted to GND and thus showing a behavior like a stuck at 0. For the case H3, logic diagnosis tool gives one gate as candidate associated to the slow to rise transition fault (StR) at the input A. In this case intra-cell diagnosis gives just one suspect the source terminal of transistor N0 that is the actual defect. Once again, this proves the efficiency and accuracy of our approach.

4.2.2 Circuit M

For circuit M, logic diagnosis identifies a stuck-at 0 fault at the output of the gate I552 (AO7HVTX1). The intra-cell diagnosis flow is then applied to this gate. Two open defects have been identified as suspects. Figure 12a) depicts the netlist at transistor level of the suspect gate AO7HVTX1 and shows the two identified open defect in red.

To verify the intra-cell diagnosis result, we performed the PFA. In this case, the PFA identified a multiple open defect (5 contacts are deformed and missing). Fig. 12b) reports in red the 5 open defects on the transistor level netlist of the suspect gate.

Since our approach is based on the single defect assumption, the result of the intra-cell diagnosis is two open defects equivalent to the real multiple defects. Even if the diagnosis is

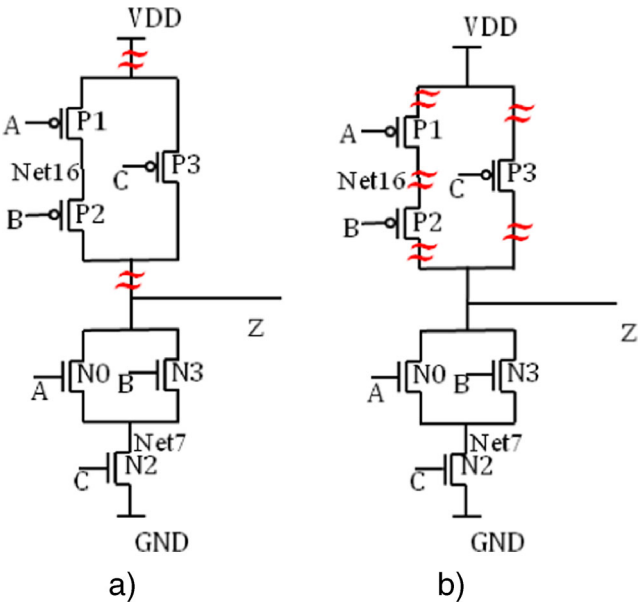

Fig. 12 AO7HVTX1 transistor netlist and **a** intra-cell diagnosis result **b** PFA results

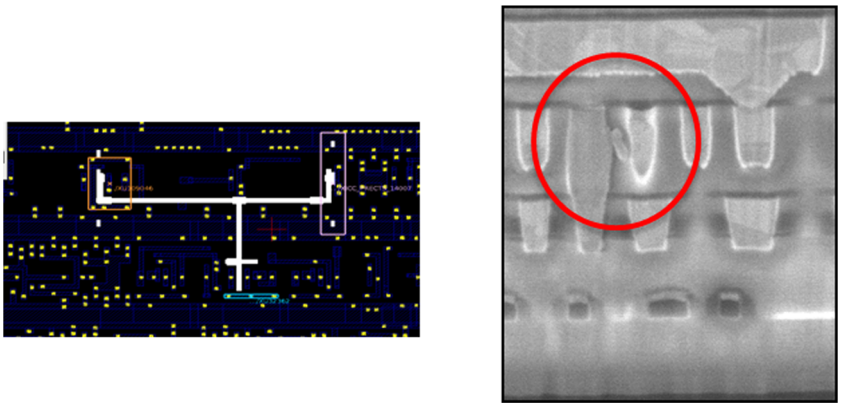

Fig. 13 Layout view of suspects (*left*) and actual defect (*right*)

not correct, it is very interesting to note that the suspect location identified by our approach include the actual defect position. That means that even in the presence of a multiple defect, the intra-cell diagnosis can provide useful information to correctly guide the PFA.

4.2.3 Circuit C

For circuit C, logic diagnosis gives a composite stuck-at fault model 01 at the output of gate U32362 as suspect. Then, the intra-cell diagnosis is thus applied to this gate. The intra-cell diagnosis flow provides an empty list of suspects. This result means that the actual defect is not inside a gate (i.e., it is an inter-cell defect).

Thanks to this result, the PFA is applied to search an inter-cell defect. The analysis has been carried out closest nets to the suspect gate. Layout of these nets is shown in Fig. 13 (left picture).

The PFA successfully identified the actual defect as a bridging fault as shown in Fig. 13 (right picture). In this case, the result of the intra-cell diagnosis avoids wasting time in investigating for a defect inside the suspect gate.

The second case of failure concerning the circuit C has been exploited to compare our approach w.r.t. the defect- and fault- dictionary based approaches. For this experiment we built a dictionary only for the cell identified by the inter-cell diagnosis (i.e., because the dictionary was not created during the design of the circuit). The defect-dictionary was built by using the well-known defects reported in [11] [15] [16]. The fault-dictionary was built by using the fault models described in [1]. To create the dictionaries we applied the serial simulation algorithm (i.e., we injected one defect/fault at a time) that has a complexity corresponding to $O(n^2)$ per pattern, where n is the number of defects/faults. The complexity is dominated by the bridging defect/faults for which we have to consider all the possible combination of two nets. On the other hand, the proposed approach requires two simulations per pattern (the first is the fault free simulation and the second one corresponds to the CPT application). Thus, our approach has a complexity of $O(1)$.

All the approaches report one candidate that was actually a short between two nets as reported in Fig. 14. This experiment is important because it shows that our approach can be precise as the one based on defect- and fault-dictionary. Moreover, for this case we do not have the pre-computed dictionary because at the time were the circuit was designed no dictionary were computed. Thus, it was more time consuming to build the dictionary even for only one cell than simply apply our approach.

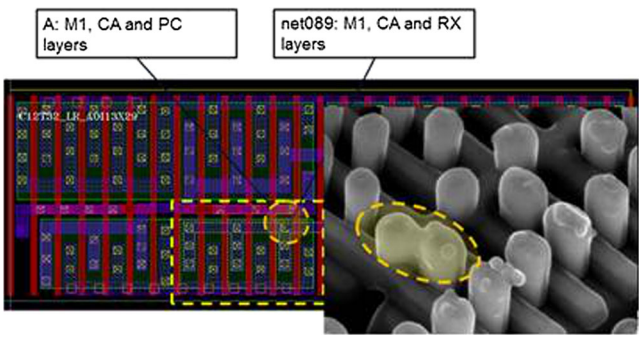

Fig. 14 Layout view of suspects (*left*) and actual defect (*right*)

5 Conclusion

In this paper, we have presented an intra-cell diagnosis approach. The proposed approach is based on the Critical Path Tracing here applied at transistor level. The CPT exploits the knowledge of the faulty behavior induced by a defect so that the proposed intra-cell diagnosis is not depended on a given defect. The diagnosis approach takes into consideration 3 fault models such as stuck-at, bridging and delay to represent the faulty behaviors induced by the physical defects (i.e., short and open).

The proposed intra-cell diagnosis approach has been validated by means of simulation and on actual silicon

data. It leads to a precise localization of the root cause of observed errors. Experimental results on actual cases show that the intra-cell diagnosis generally gives meaningful information further exploited by PFA engineers. This information allows saving times in searching the root cause of a faulty device.

Future works are devoted to extend the proposed approach to handle scan flip-flops and in how to improve the achieved diagnosis resolution.

References

1. Amyeen ME, Nayak D and Venkataraman S (2006) "Improving Precision Using Mixed-level Fault Diagnosis," IEEE International Test Conference (ITC), pp. 1–10
2. Bosio A, Girard P, Pravossoudovitch S, Virazel A (2010) A comprehensive framework for logic diagnosis of arbitrary defects. IEEE Trans Comput 59(3):289–300
3. Bryant R, Beatty D, Brace, K. Cho K. and T. Shefler (1987) "COSMOS: A Compiled Simulator for MOS circuits," IEEE Design Automation Conference (DAC), pp. 9–16
4. Bushnell M, Agrawal V (2000) Essentials of electronic testing for digital, memory and mixed-signal VLSI circuits. Springer, New York
5. Desineni R, Poku O and Blanton RD (2006) "A logic diagnosis methodology for improved localization and extraction of accurate defect behavior," IEEE International Test Conference (ITC), pp. 1–10
6. Eichenberger S, Geuzebroek J, Hora C, Kruseman B and Majhi A (2008) "Towards a World Without Test Escapes," IEEE International Test Conference (ITC), paper 20.1
7. Fan X, Moore WR, Hora C, Gronthoud G (2007) Extending gate-level diagnosis tools to CMOS intra-gate faults. IET Comput Digit Tech 1(6):685–693
8. Hapke F, Krenz-Baath R, Glowatz A, Schloeffel J, Hashempour H, Eichenberger S, Hora C and Adolfsson D (2009) "Defect-Oriented Cell-Aware ATPG and Fault Simulation for Industrial Cell Libraries and Designs," IEEE International Test Conference (ITC), paper 1.2
9. Hapke F, Redemund W, Schloeffel J, Krenz-Baath R, Glowatz A, Wittke M, Hashempour H and Eichenberger S (2010) "Defect-Oriented Cell-Internal Testing," IEEE International Test Conference (ITC), paper 10.1
10. Hapke F, Schloeffel J, Hashempour H and Eichenberger S (2011) "Gate-Exhaustive and Cell-Aware pattern sets for industrial designs," International Symposium on VLSI Design, Automation and Test (VLSI-DAT), pp.1–4
11. Hashempour H, Dohmen J, Tasic B, Kruseman B, Hora C, Van Beurden M and Xing Y (2011) "Test time reduction in analogue/mixed-signal devices by defect oriented testing: an industrial example," Design, Automation & Test in Europe Conference (DATE), pp. 1–6
12. Kruseman B, Majhi A, Hora C, Eichenberger S and Meirlevede J (2005) "Systematic defects in deep sub-micron technologies," IEEE International Test Conference (ITC), pp. 290–299
13. Ladhar A and Masmoudi M (2009) "Efficient and Accurate Method for Intra-gate Defect Diagnoses in Nanometer Technology and Volume Data," Design, Automation & Test in Europe Conference & Exhibition (DATE), pp 988–993
14. Livengood R and Medeiros D (1999) "Design for (physical) debug for silicon microsurgery and probing of flip-chip packaged integrated circuits," IEEE International Test Conference (ITC), pp. 877–882
15. Rodriguez-Montanes R, Bruis E and Figueras J (1992) "Bridging defects resistance measurements in a CMOS process," IEEE International Test Conference (ITC), pp. 892–899
16. Rodriguez-Montanes R, de Gyvez JP, Volf P (2002) Resistance characterization for weak open defects. IEEE Des Test Comput 19(5):18–26
17. Sun Z, Bosio A, Dilillo L, Girard P, Todri A, Virazel A and Auvray E (2012) "Fault Localization Improvement through an Intra-Cell Diagnosis Approach," International Symposium for Testing and Failure Analysis (ISTFA), pp. 509–519
18. Sun Z, Bosio A, Dilillo L, Girard P, Todri A, Virazel A and Auvray E (2013) "Effect-Cause Intra-Cell Diagnosis at Transistor Level," International Symposium & Exhibits on Quality Electronic Design (ISQED), pp. 476–483
19. Tam W-C, Poku O and Blanton RD (2008) "Precise Failure Localization using Automated Layout Analysis of Diagnosis Candidates," Design Automation Conference (DAC), pp. 367–372
20. Thompson KM (1996) Intel and the myths of test. IEEE Des Test Comput 13(1):79–81

Zhenzhou Sun was born in ZiBo, China. He received his MSc in electronics and embedded systems in 2010 at « ESISAR-Grenoble INP » in France. He is currently a Ph.D. student at laboratory LIRMM of Montpellier University in Montpellier, France. His research interests include, IC testing and logic diagnosis.

Alberto Bosio obtained his Master and Ph.D. diploma in Computer Engineering at Politecnico di Torino (Italy) in 2003 and 2006 respectively. After 1 year of post-doc he became an Associate Professor in the Laboratory of Informatics, Robotics and Microelectronics of Montpellier (LIRMM)-University of Montpellier France. The research activities target the definition of new methodologies and the implementation of tools able to improve the development of highly dependable systems, at different levels: for basic digital components, for systems on chip, up to microprocessor-based systems.

Luigi Dilillo has been a Ph.D. student at the microelectronics department of LIRMM (Laboratoire d'Informatique, de Robotique et de Microelectronique de Montpellier) in Montpellier France. He has also taught, with the role of monitor, computer sciences and electronics at the University of Montpellier II. After a period, as research fellow at the University of Southampton (UK) in the Electronics and Computer Science Department, he worked on electronic circuit reliability at CEA (French Commission for Atomic Energy). At this moment he is CNRS researcher at LIRMM laboratory. The fields of interest of his researches are MEMS and digital circuits. His current studies are on delay-fault testing, memory testing and test of low power circuits.

Patrick Girard received a Ph.D. degree in Microelectronics from the University of Montpellier, France, in 1992. He is currently Research Director at CNRS (French National Center for Scientific Research), and Head of the Microelectronics Department of the Laboratory of Informatics, Robotics and Microelectronics of Montpellier (LIRMM) - France. He is also co-Director of the International Associated Laboratory « LAFISI » (French-Italian Research Laboratory on Hardware-Software Integrated Systems) created in 2012 by the CNRS and the University of Montpellier 2 with the Politecnico di Torino, Italy. His research interests include all aspects of digital testing and memory testing, with emphasis on critical constraints such as timing and power. Reliability and fault tolerance, power modeling and estimation, as well as design and test of 3D ICs are also part of his new research activities. Patrick Girard is Technical Activities Chair of the Test Technology Technical Council (TTTC) of the IEEE Computer Society. He has served on numerous conference

committees, he is the founder and Editor-in-Chief of the ASP Journal of Low Power Electronics (JOLPE), and he is an Associate Editor of the IEEE Transactions on VLSI Systems and the Journal of Electronic Testing—Theory and Applications (JETTA—Springer). Patrick Girard has supervised 32 PhD dissertations and has published 6 books or book chapters, 45 journal papers, and more than 160 conference and symposium papers on these fields. Patrick Girard is a Senior Member of IEEE and a Golden Core Member of the IEEE Computer Society.

Serge Pravossoudovitch was born in 1957. He is currently professor in the electrical and computer engineering department of the University of Montpellier and his research activities are performed at LIRMM (Laboratoire d'Informatique, de Robotique et de Microélectronique de Montpellier). He got the PhD degree in electrical engineering in 1983 for his work on symbolic layout for IC design. Since 1984, he is working in the testing domain. He obtained the "doctorat d'état" degree in 1987 for his work on switch level automatic test pattern generation. He is presently interested in memory testing, fault diagnosis, design for testability and power consumption optimization. He has authored and co-authored numerous papers on these fields, and has supervised several PhD dissertations. He has also participated to several European projects (Microelectronic regulation, Esprit, Medea).

Arnaud Virazel was born in Montpellier (France) in 1974. He received a M.S. degree in Electrical Engineering and a Ph.D. degree in Microelectronics from the University of Montpellier, France, in 1997 and 2001 respectively. He is currently Associate Professor at the University of Montpellier II, and works in the Microelectronics Department of the LIRMM (Laboratory of Informatics, Robotics and Microelectronics of Montpellier—France). Arnaud Virazel has been on the technical program committee of the International Conference on Embedded and Ubiquitous Computing (EUC) in 2005, the International Symposium and Exhibits in Quality Electronic Design (ISQED) since 2008, the IEEE International Conference on Design and Technology of Integrated Systems in Nanoscale Era (DTIS) since 2008, the ACM/IEEE Design Automation and Test in Europe (DATE) since 2009 and the IEEE European Test Symposium (ETS) since 2010. He has also served as reviewer for most of the test conferences (ITC, VTS, DATE, ATS, ETS, DDECS, DELTA, IOLTS) and journals (JETTA, TCAD, TODAES, JOLPE, IET). Arnaud Virazel has been involved in several European research projects (MEDEA ASSOCIATE, MEDEA NanoTEST, MEDEA TOETS). His research interests include the various aspects of digital testing, with special emphasis on DfT, BIST, diagnosis, delay testing, power-aware testing and memory testing. He has published one book, more than 20 journal papers and 70 conference and symposium papers on these fields.

Etienne Auvrey obtained the engineering degree from Ecole Supérieure d'Electricité in 1979. He worked in Failure Analysis for 33 years starting at EFCIS in Grenoble in 1979 working on both physical and electrical failure analysis. In 1987, he focused his activity in the military space component of Thomson that extended FA expertise to CCD and IIIV devices. In 1995, the lab was externalized to Serma technology group and started FIB services in various domains: device modification, TEM sample preparation on any products (IC, MEMS, magnetic heads, etc.). In 2001, he joined STMicroelectronics as TPA FA lab manager and is currently in charge of FA methodologies in the automotive group.