Reliability Challenges for Electric Vehicles: From Devices to Architecture and Systems Software

Georg Georgakos
Infineon Technologies
Neubiberg, Germany

Ulf Schlichtmann
Institute for Electronic Design Automation
Tech. Universität München, Germany

Reinhard Schneider
Samarjit Chakraborty
Institute for Real-Time Computer Systems
Tech. Universität München, Germany

ABSTRACT
Today, modern high-end cars have close to 100 electronic control units (ECUs) that are used to implement a variety of applications ranging from safety-critical control to driver assistance and comfort-related functionalities. The total sum of these applications is several million lines of software code. The ECUs are connected to different sensors and actuators and communicate via a variety of communication buses like CAN, FlexRay and now also Ethernet. In the case of electric vehicles, both the amount and the importance of such electronics and software are even higher. Here, a number of hydraulic or pneumatic controls are replaced by corresponding software-implemented controllers in order to reduce the overall weight of the car and hence to improve its driving range. Until recently, most of the software and system design in the automotive domain – as in many other domains – relied on an always correctly functioning or a *zero-defect* hardware implementation platform. However, as the device geometries of integrated circuits continue to shrink, this assumption is increasingly not true. Incorporating large safety margins in the design process results in very pessimistic design and expensive processors. Further, the processors in cars – in contrast to those in many consumer electronics devices like mobile phones – are exposed to harsh environments, extreme temperature variations, and often, strong electromagnetic fields. Hence, their reliability is even more questionable and must be explicitly accounted for in all layers of design abstraction – starting from circuit design to architecture design, to software design and runtime management and monitoring. In this paper we outline some of these issues, currently followed practices, and the challenges that lie ahead of us in the automotive and electric vehicles domain.

Categories and Subject Descriptors
B.4.5 [**Hardware**]: Reliability, Testing, and Fault-Tolerance – *Diagnostics, Error-checking, Hardware reliability, Redundant design.*

General Terms
Performance, Low Power, Design, Reliability, Verification.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.
DAC '13, May 29 – June 07 2013, Austin, Texas, USA.
Copyright 2013 ACM 978-1-4503-2071-9/13/05 …$15.00.

Keywords
Electric vehicles, automotive electronics, process variations, aging, embedded systems, software, cross-layer.

1. INTRODUCTION
The volume of electronics and software in modern cars is increasing at a tremendous rate. Today, most of the innovation in the automotive domain is in the area of electronics and software rather than in mechanical engineering, which used to be the case until a few years ago. High-end cars now have reached the mark of close to 100 ECUs, containing almost 250 processors and GPUs [1]. These ECUs are connected to various communication buses like CAN, LIN, MOST, FlexRay and Ethernet and are used to run various applications related to safety-critical control, driver assistance and comfort-related functionalities. These applications sum up to several million lines of software code and are expected to grow at an exponential rate as more driver assistance functions are being introduced everyday. In fact, *autonomous driving*, that is technologically feasible since the last couple of years, is expected to become a reality very soon. Such applications rely on a number of cameras and radars that produce large volumes of streaming data, which needs to be processed in real-time thereby further increasing the need for computation power inside the car.

Electric vehicles: In the case of electric vehicles, electronics and software – both in volume, as well as in importance – are even larger. Since battery life and hence driving range is of crucial importance in electric cars, a number of pneumatic and hydraulic controllers in such cars are replaced by software-based control in order to reduce the overall weight of the car. This further increases the need for computation power, but also introduces more ECUs and cabling. Current design methods in the automotive domain follow a *federated architecture*, where each function is implemented on a separate ECU. This enables the OEM (original equipment manufacturer) to outsource the different functions to different Tier 1 suppliers – with the role of the OEM being that of function integrator. But it also increases the number of ECUs in the car and the volume of cables needed to connect these ECUs. For electric cars this is especially a problem, not only because of the complexity and the distributed nature of the resulting architecture, but also because of the overall weight of the ECUs and the cables, which are now non-negligible. As a result, there is an increasing push to move towards more *integrated* architectures, where multiple software functions are integrated onto a single ECU. An application therefore is now distributed into different tasks, running on different ECUs and communicating via shared communication buses. Such integrated architectures and the move towards fewer ECUs and shorter

cables introduce more powerful multicore ECUs that use the latest processor design and fabrication technologies. Further, very soon it will also no longer be cost-effective to use simple microcontrollers in the ECUs but powerful commodity processors from the consumer electronics domain will be used.

The reliability problem: However, such processors from the consumer electronics domain are currently faced with a number of reliability-related problems stemming from process variations, aging, and radiation-induced soft errors, which could be safely neglected in the low-end microcontrollers in older ECUs that had much larger device geometries and used old fabrication technologies. Hence, design methods and software development techniques in the automotive domain always assumed a zero-defect and fully reliable hardware layer, which is increasingly no longer true.

To make matters worse, the electronics in a car are exposed to harsh conditions, extreme temperature variations, and often, strong electromagnetic fields, which further aggravates the reliability problem. Further, the electronics in subsystems like *battery monitoring and management* in electric vehicles are always "on" for the entire lifetime of the car, which is in the range of 10-15 years and sometimes even more. This makes issues like aging an important concern. Automotive being a highly cost sensitive domain, large safety margins to overcome these problems are not feasible and more integrated and intelligent solutions that tackle the reliability issue at multiple layers of design abstraction are necessary. While current processor design for automotive ICs is in the 65nm domain, the industry is rapidly moving towards processors with smaller geometries (40nm and below in the near future). For devices with these geometries power consumption is a major issue. Hence, power management techniques like dynamic voltage and frequency scaling – that are routinely used in consumer electronics devices like mobile phones – will have to be introduced in automotive ECUs (currently the power consumed by the electronics in a car is largely neglected and hence no runtime power management techniques are deployed). Once this happens, the issue of reliability will become a bigger challenge since the processor will have to operate at both high and low supply voltages, which is explained in more detail in Section 2.

Furthermore, in electric vehicles, conventional control systems with mechanical backup systems are likely to be replaced by solely electronic components, so called *X-by-wire systems* such as brake-by-wire or steer-by-wire. This leads to significantly reduced weight and space and enables a tight integration of several applications. On the other hand, such systems, e.g., brake-by-wire, are highly safety-critical and impose stringent reliability requirements on the used hardware and software. Hence, fault-tolerant and reliable software and platform design will become inevitable key issues to realize X-by-wire systems.

Today, processors in automotive ECUs are increasingly adopting multicore architectures to keep pace with growing performance and reliability demands. Similarly, cost and quality requirements necessitate efficient model-based development techniques to realize seamless model-based software development involving automatic code generation. Generally, the system design process is typically distributed among several layers, starting at a high-level application layer implementing the core functionality of the system at a high level of abstraction, e.g., in form of Simulink models or as textual representations, down to the microarchitecture layer on which the embedded software is finally implemented. In this context, the specification, design, test and verification phases across the different layers are driven by various automotive standards such as OSEK/VDX, AUTOSAR, ISO26262, CMMI (to name but a few), each imposing individual constraints and requirements.

Further, the growing complexity due to increased ECU consolidation and associated integration of more and more SW-components with different levels of criticality on complex multicore architectures requires integrated design approaches in hardware and software to guarantee *freedom from interference* between SW-components and to meet specified performance requirements. In this context, interference between SW-components may be due to concurrent memory accesses to shared memory regions or because of preemption of tasks scheduled on shared processing resources resulting in increased execution times.

In order to realize safety-related architectures, dedicated multicore platforms operating in dual core *lock-step mode* have become popular. That is, two cores (master and checker) execute the same code while being synchronized, to detect potential errors produced by a faulty core. This is especially important to satisfy high functional safety requirements, e.g., in a braking ECU. On the other hand, multicore architectures of course provide parallel processing on multiple cores giving rise to increased computational performance. Similarly, dedicated hardware units such as DMA controllers, DSPs, and GPUs enable additional execution speed-up. Further, virtualization technologies are being studied with the goal to implement an effective layer of isolation between single user-level applications with different criticality levels.

The communication network between the ECUs is built up of bus systems such as LIN, CAN, MOST, FlexRay, Ethernet and one or more gateways to interconnect the different network domains. Since gateways often represent single-point-of-failures in the communication network special attention needs to be paid to designing fault-tolerant and redundant communication architectures in order to improve the reliability in the entire network.

Organization: In this paper we discuss the various reliability issues and why they arise (Sect. 2) as well as current practices to mitigate them (Sect. 3). Sect. 4 discusses reliability especially in the automotive context. The challenges that lie ahead of us are addressed in this paper as well in Sections 3 and 4.

2. RELIABILITY: BACKGROUND AND CIRCUIT IMPACT

This section will provide some background on the causes and effects for the reliability challenges that we address in this paper. The following section describes mitigation techniques which represent the current state of the art.

Manufacturing variations have been with us since the early days of ICs. They are, together with the need to consider the range of operating conditions (supply voltage V_{DD} and operating temperature T), the reason why the corner-based design methodology has been introduced. They happen due to imperfections in the manufacturing process, which either have fundamental physical reasons, or are a result of tradeoffs between manufacturing cost and quality. Typically they are variations of physical parameters, which in turn result in variations of electrical parameters. These affect circuit properties that are of interest to the designer (e.g., delay, static/dynamic power). Some examples

of manufacturing variations are e.g. fluctuations in doping profiles of the transistor channel (resulting in variations of threshold voltage V_{th}), variations in gate oxide thickness *Tox* (V_{th} change), width/thickness of metal lines (changes in resistance and therefore wire delay as well as electromigration risk) [2][3].

Various classifications of these variations exist [4]. An often used classification refers to their spatial correlation: wafer-to-wafer (all parameters on one wafer are identical, but differences exist between wafers), die-to-die (parameters within one die are (essentially) identical, but differ between dies), and within-die. The first two classes are often referred to as global variations, whereas the within-die variations are also known as local variations. The poster-child for within-die variations is random dopant fluctuations (RDF) of the transistor gate channel doping, which affect V_{th}. From 32nm onwards, the number of dopant atoms is below 100 [5]. As the dopant atoms are deposited by processes, their exact number can differ significantly from one transistor to the next, even between adjacent transistors. Minimum-sized transistors are affected the most.

The trend is that manufacturing variations increase from one technology generation to the next [6][7][8]. This is especially true for their relative values, since the nominal values of parameters typically decrease with successive process technology generations [2]. However, this is not a monotonous trend. Sometimes innovations in process technology, new materials or improved transistor structures result in a significant reduction in variations. But the overall trend of increasing relative variations remains. Especially worrying is the fact that within-die variations are increasing as a percentage of overall variations [9]. This is a problem since the standard corner-based design methodology was developed for wafer-to-wafer and die-to-die variations and cannot properly handle within-die variations.

Defects during the manufacturing process are usually caused by impurities of the process materials or particle contamination during the process itself. These defects might directly cause a functional fail, e.g., an open of an interconnect or contact. But even worse they might create a weak spot in a device, such as a local gate oxide thinning of a MOS transistor or a partly open via hole. These latent faults may not be detected during initial product test, and transform to a functional fail or a parameter drift quite early during the lifetime of the device in the field. Most of the reliability mechanisms described in the following can be influenced or accelerated by these defects which are also described as extrinsic effects in contrast to the intrinsic behavior which describes the reliability mechanism in an ideal case without any type of an overlaying defect.

Aging effects are changes in transistor or wire parameters over the lifetime of the device. Today, the following are the most relevant effects:

Negative / Positive Bias Temperature Instability (NBTI / PBTI). NBTI affects PMOS transistors, PBTI NMOS transistors. In the past, PBTI was not an issue, but with the introduction of high-k metal gates below 40nm it is evolving into a problem as well. The aging happens when the transistor is in inversion (the stress condition), e.g. for the PMOS when the gate terminal is negatively biased regarding source or drain. The resulting changes in the transistor can be modeled as a degradation of V_{th}. The severity of the degradation depends on many factors, most importantly V_{DD}, T, and percentage of the time the transistor is in inversion. Transistor degradation due to NBTI or PBTI can be reversed when the stress condition is removed. This is known as recovery. Recovery is taking place slower than degradation, and only partially reverses the previous degradation.

Figure 1 and Figure 2 show some influences on NBTI [10]. The circuit designer is interested in the transistor delay degradation caused by an aging effect such as NBTI. This delay degradation has two components: NBTI causes a shift of transistor V_{th} (Figure 1), which then in turn results in a degradation of the delay of the logic gate containing the respective transistor (Figure 2). As the figures show, V_{DD} and T influence both components. However, V_{DD} has different impact on the two components: the V_{th} degradation increases with increasing V_{DD}, the sensitivity of delay to V_{th} increases with decreasing V_{DD}. This can be especially critical in a scenario where different supply voltages V_{DD} are used for purposes of power reduction. Assume that an IC is operated mostly using a high V_{DD} value for high performance modes. This will result in strong transistor degradation. When the IC then operates with lower V_{DD} in a low-power mode, the impact of the V_{th} shift is amplified by the increased sensitivity of delay to V_{th} changes (see Figure 2). This demonstrates that techniques for power reduction cannot be considered in isolation. Their effects on reliability due to aging need to be taken into account.

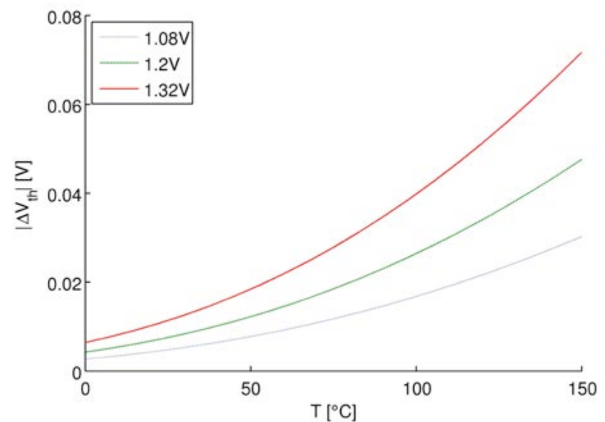

Figure 1: Exemplary V_{th} shift of a PMOS transistor in a 90nm node caused by NBTI stress representing a realistic end of live scenario and a 100% duty cycle in dependence on T, V_{DD}

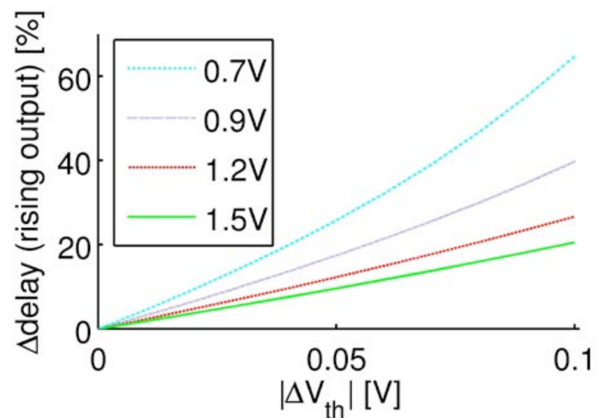

Figure 2: Delay degradation of a 90nm inverter in dependence on V_{th} shift and V_{DD} at an operating temperature of 85°

Hot Carrier Injection (HCI) affects both PMOS and NMOS transistors. Essentially, carriers (electrons or holes) are injected from the transistor channel into the gate oxide, resulting in a permanent change of the transistor characteristics, typically modeled as a degradation of the on-current I_{on}. This happens during the transition from the off- to the on-state of the transistor, so the switching frequency of the transistor strongly affects the degradation. The impact of HCI on digital circuits is similar to NBTI and PBTI as described above.

Time Dependent Dielectric Breakdown (TDDB) affects both PMOS and NMOS transistors. However, the failure rate of NMOS transistors due to TDDB is much higher. TDDB is defined as the localized loss of isolating properties of the gate oxide, which leads to an increased gate leakage current. Typically the impact of this aging mechanism is modeled as a degradation of the gate current I_g. Unlike NBTI, there is no recovery from TDDB. There are two oxide breakdown (OBD) modes to distinguish: Soft breakdown (SBD) and Hard Breakdown (HBD). The underlying physical origins for both modes are the same, and they are distinguished by the resistance of the conducting path, or, in other words, the severity of breakdown. SBD has relatively higher resistance and lower breakdown current. In the SBD stage the transistor drifts in energy and delay, but is still functional. The HBD mode is always preceded by SBD. In the HBD case, the resistance is much lower and the current is exponentially higher. This fault is called catastrophic and results in a permanent malfunction of the respective circuit block.

As the gate oxide thickness is shrinking with successive process technologies, transistors are becoming more prone to SBD, which has become one of the most important reliability issues threatening the further reduction of transistor feature sizes. The principles of scaling of gate area, oxide thickness, and voltage for both soft and hard breakdowns have been investigated [17]. The study of relation between breakdown mode and its location suggests that the spot, located in the gate and drain/source overlapping region, will have the most severe impact on circuit functionality [18]. On a circuit level, the impact of TDDB is being studied mainly on SRAM cells, as they are most sensitive to TDDB [19][20].

Electromigration (EM) refers to mass transport in metal interconnects like wires and via holes [11]. With sufficiently high current densities, electrons may eventually cause atoms to move. Over time this can result in an increase in resistance as the interconnect cross-section becomes narrower. Eventually it can lead to an open, likely causing a catastrophic circuit failure. For a given current, the severity of EM increases with decreasing interconnect width. Therefore, EM problems can be avoided by sufficiently sizing wires or number of via holes. But of course this approach is at odds with the desire to keep wires as thin as possible to reduce IC area.

Stressmigration (SM) is similar to electromigration but the mass transport is not due to an electron flow but due to diffusion of vacancies in interconnects through mechanical stress or temperature gradients. An agglomeration of these vacancies may eventually create voids under or in via holes. This also leads to an increase in resistance first and might eventually cause a permanent open of an interconnect [24].

Soft Errors or **Single Event Upsets (SEUs)** cause intermittent, non-permanent failures in a circuit. An SEU happens when a charged particle strikes a semiconductor, resulting in a buildup of ionized material, which in turn causes a current. The charged particle might result from radioactive decay of packaging materials or from cosmic particles in the atmosphere. As the incidence of cosmic particles increases with height above the earth’s surface, in the past electronics for air and space applications were most at risk. However, with decreasing feature sizes, also electronics used in terrestrial applications are increasingly affected. While the literature is not uniform as to whether smaller feature sizes by themselves increase SEU risk, certainly the exponential increase in number of components on a chip due to shrinking feature sizes increases the risk for an IC of a given die size. Today, no safety-critical IC is designed without some sort of SEU protection anymore.

In the past, most concern has been about particle strikes directly hitting a storage cell, resulting in the flip of a single bit in memory. Meanwhile more than one bit may flip in a memory due to a single event. And increasingly, single event transients (SETs, pulses resulting from a particle strike which then propagate through the circuit) are also a source of concern, since they can also lead to an incorrect value in a storage element (e.g. a flipflop) if the transient reaches such a storage element at the time it latches.

Both SEUs and SETs may not necessarily cause a malfunction of a circuit. There are many types of masking that might happen. E.g. a flipped memory bit might be overwritten by a new value before it is accessed by a read operation. It is challenging to determine how much at risk a certain part of a circuit is. Research is conducted e.g. to efficiently determine the Architectural Vulnerability Factor (AVF) – which describes the masking of errors - of a circuit [33].

Electromagnetic Interference (EMI) is not a classical reliability issue but also requires a certain robustness of the design regarding high electromagnetic and pulse disturbances. They are typically injected or generated and distributed in complex systems consisting of one or several printed circuit boards, connectors and cables. The noise is either received from a harsh environment which contains radio frequency or pulse noise or is generated by the systems themselves mostly due to switching noise. Coupling of noise is due to jointly used interconnects of an emitting and receiving part of the system in a conducting way or through traces of interconnects, packages, connectors or cables acting as an antenna. Usually no physical destruction of the components occurs. Only a temporarily or permanent malfunction of the system is observed.

A special case of interference is Electrostatic Discharge (ESD). The according noise pulse is in this case typically provided through the chip pins and caused by the handling of the chip during production or system assembly or by many kinds of overstress conditions during chip operation. An ESD exposure can also happen during end-user operation in case of a plug-in of cables and change of batteries or maintenance in an unprotected area. A special kind of electrostatic discharge during the chip manufacturing process might occur due to resist charging during implantation or due to plasma processes which create non-uniformities of charge distribution across a wafer. The charge is coupled into the chip by antennas, simple traces of metal or diffusion. A corresponding discharge might lead to various parameter drifts of active devices, increased TDDB risk, increased gate leakage or immediate physical destruction of diodes, transistor channels or dielectrics.

Above, the circuit impact of the described reliability effects was considered from the perspective of a digital circuit design. Complex SoCs may also contain analog and mixed signal circuits. Reliability effects that lead to a permanent physical destruction, do not distinguish between digital or analog circuits. However, the operating conditions of the devices are different, leading to different acceleration of the related reliability effects. In digital circuits MOS transistors are usually operated at full supply voltage levels and current flow is limited to the switching transients whereas in analog circuits the supply voltage of a single MOS transistor is usually smaller due to voltage drop at serially connected devices, but the current flow spans a wide range of different pulse shapes, up to a constant current over time. Reliability effects, which lead to a parameter degradation of a device, are more difficult to consider for analog circuits. As above the operating conditions are different compared to digital circuits and therefore also the acceleration factors. Additionally not only V_{th} and I_{on} drifts impact the analog circuit behavior, also parameters like g_m and g_{ds} might suffer from aging and change the circuit response over time. And finally the wide variety of analog circuit classes shows a big difference in sensitivity to certain parameter changes. This is mainly due to design techniques utilizing relative parameters instead of absolute ones relying on a good matching of these parameters to cope with variations.

3. RELIABILITY-AWARE DESIGN TECHNIQUES

This section describes reliability-aware design techniques that are considered to be "state of the art" for advanced IC designs, and also gives an outlook. For the purposes of this paper, we define „state of the art" as techniques that are used for designing automotive ICs in 65nm technology. In the rather conservative automotive field, major vendors are currently qualifying this process technology. Design activities have started in 40nm, but are still in early stages. The focus of the descriptions in this section is on broadly applicable techniques, not on approaches specific to automotive ICs.

3.1 Device and Circuit Level Techniques

Manufacturing variations have traditionally been addressed by corner-based (best case / worst case) design techniques [12]. Increasingly, such techniques are running out of steam, however, due to a number of reasons:

- Variations are increasing. This results in ever more performance being lost to generous corner guard banding.
- More parameters are being considered, resulting in an exponential increase in the number of corners. E.g. some manufacturers consider variations in wire thickness on different metal levels in addition to transistor speed variation.
- Corner-based design methodologies can not cope with purely random within-die variations. These variations can only be addressed by safety-guardbands – and by the hope that they will typically average out if a path is long enough. Unfortunately, hoping for the best is not a very reliable design technique. To date, the industry has managed to get by with tweaking existing design techniques. E.g. Static Timing Analysis is being enhanced with various forms of "On-Chip-Variation (OCV)" techniques to address purely random within-die variations [13]. These work for the time being, but are really just band-aids to postpone the demise of established design techniques a bit further into the future.

Statistical design techniques for digital circuits have seen a lot of attention in the past 10 years [4]. But most of them have not yet gained any significant traction in typical industrial design flows, with the exception of in-house flows at a few of the world's largest IDMs. As the effort required to introduce such techniques into design flows (new tools, significantly more detailed library characterization required, very high computational effort required for some approaches) is very substantial, it remains to be seen to which degree statistical design techniques will see industrial adoption in the future.

In analog circuit design Monte-Carlo simulations result in more realistic behavior than corner-based methods, with the disadvantage of highly increased design time. Worst-Case-Distance methods [14] are much faster but also very complex and are therefore not yet established in the analog design community. With increasing complexity and sensitivity of the circuits towards variations this situation has to change in future to enable an efficient and accurate design process.

Extrinsic faults can't be addressed directly during the design process. A critical area analysis and layout post-processing like wire spread and fattening techniques can reduce the number of faults. Current strategy is to activate these latent faults with a burn-in stress and screen them out with a respective test program before final shipment. This procedure is going to be more and more challenging in the future because burn-in is time consuming and therefore quite expensive and on the other hand due to increased system complexity it is very challenging to find realistic burn-in stress scenarios which activate all possible latent faults.

Aging can provide an exemplary perspective on the evolution of design techniques to address reliability challenges (DfR, Design for Reliability). In the past, many of these reliability challenges were not addressed specifically during design at all. Overall guard bands were being used in chip design, and it was assumed that these guard bands would cover aging effects, if any consideration was given to them at all.

As a next step up in sophistication, the size of these guard bands was verified by measurements on individual components (usually transistors) under stress conditions. These measurements typically result in conservative worst-case values which are used to determine the size of guard bands. However, they usually are quite conservative, resulting in unnecessary design effort and/or increased IC area and delay. On the other hand, they might not be sufficient to cover extreme cases – and a single extremely slow path can be sufficient to render an entire IC useless.

A major enhancement in DfR is then to perform a detailed, circuit-specific timing analysis incorporating aging. As transistor level analysis can only handle rather small circuit sizes due to the required computational effort, a prerequisite for such analysis is a modeling of aging effects at gate level and higher abstraction levels. Efficient and versatile techniques for such modeling have only recently been proposed [16]. Such specific analysis techniques are also a prerequisite to optimize the circuit against aging effects. The literature reports primarily on variations of optimization techniques that have been used e.g. in timing and power optimization before, such as pin reordering and gate restructuring [15].

Such analyses and optimizations performed during design will in the future need to be enhanced by run-time monitors, which

observe an IC during operation and enable an immediate reaction to monitoring results. Such reactions can take various forms:

- Issue a warning that an IC is reaching a specification limit, such that the corresponding ECU can be replaced. While this ensures safety, replacing an ECU is very costly.
- Increase the supply voltage so that the original frequency can be maintained despite the aging-related degradation. This results in increased power consumption (and increased future aging as well).
- Perform a graceful degradation of the performance of the IC (e.g. by reducing the workload so that the IC can perform the remaining functionality with a lower frequency). E.g. in a multi-core system, reallocate some tasks among cores so that an aged core can still be used despite offering lower performance. This requires that some spare capacity be built into the system. Also, reallocating task among cores, processors or ECUs might pose special challenges in the automotive domain due to certification issues.

An even more advanced technique of addressing aging is to take recovery effects into consideration as well. For NBTI and PBTI, recovery takes place if the stress condition is removed. This can be utilized in conjunction with online aging monitors. For example, in a multi-core system, if strong aging is sensed in a specific core, tasks can preferentially be moved to other cores to give the strongly aged core a chance to recover. However, such techniques require that specific causes of aging can be identified, rather than just the effects of the aging be noticed.

Regarding **TDDB**, the modeling of this aging effect has become a topic of much interest recently, e.g. because in ultrathin oxide transistors the time between SBD and HBD is very long [22], and the transistors can undergo even multiple SBDs [23]. Most often the breakdown is modeled as a voltage-dependent resistance between gate and drain/source to consider the worst-case scenario. Based on this, an analytical model has been developed to simulate the TDDB-based timing degradation in combinational cells [28]. However the model has potential for improvement, as it does not model the resistance change over time.

Via manufacturing risks are addressed by inserting redundant vias into the circuit. In situations where a circuit contains sufficient space for additional vias, this has no drawbacks. When wires need to be enlarged to allow the insertion of additional vias, the tradeoff between increased manufacturing cost and improved reliability needs to be considered. Therefore intelligent rule based methods to add redundant vias only where they are necessary and to verify a design by identifying only really critical single vias are needed in the future.

Soft Errors are addressed by techniques specific to different circuit components.

Memories are typically protected by parity codes or error-correcting codes (ECC), a special type of redundancy. Alternatively, transistors can be upsized, or an 8T architecture instead of the standard 6T SRAM cell can be introduced.

For flipflops in the circuit logic, ECC is not easily possible. The standard solution for such flipflops is to harden them, e.g. by upsizing the transistors. Also, double or even triple modular redundancy (DMR, TMR) or even more sophisticated techniques are possible. They can be applied to individual flipflops, or to entire larger circuit modules.

All of these techniques for protecting flipflops in logic against SEUs result in (usually significant) area and delay penalties. An important current research area is therefore to determine which flipflops are most critical to circuit operation and definitely need to be protected, and which other flipflops could be left without protection without catastrophic consequences for a circuit. The concept of the AVF is relevant here. Research has been reported using both simulation and formal techniques to determine the importance of a flipflop. The formal techniques are much less mature than simulation-based techniques today, however [29][30].

General techniques that are used to address many reliability issues are:

- Redundancy (e.g. DMR, TMR) – typically very costly
- Parity and error-correcting codes – cost-efficient implementations are typically limited to regular structures such as memories
- Circuit architectures such as RAZOR [31] or Pre-Error Adaptive Voltage Scaling [32]

3.2 System-Level Techniques

Software and system-level design techniques in the automotive domain usually follow a cross-layer design approach. Here, high-level models are used to specify (often various control) applications. Such a model-based design approach, in contrast to using, for example, handwritten code, allows formal verification and certification of the safety-critical functionality. These models are used to generate software code, which is then partitioned and mapped onto a distributed architecture consisting of various ECUs connected by communication buses.

One of the major challenges in this design flow stems from the fact that the high-level controller models more often than not ignore many platform architecture details (i.e., details of the platform on which the synthesized code is to be implemented). In other words, the models make a number of idealistic assumptions – like control functions are computed in zero/negligible time, there is no delay between sensing and actuation, etc. – which are increasingly not true in modern distributed automotive architectures. Hence, control performance properties that are proven at the model level do not hold true in the actual implementation, thereby requiring considerable integration and debugging efforts and raising questions on the safety/reliability of the resulting system.

In order to address this issue, the abovementioned design flow has to be suitably modified to take into account relevant platform architecture level details during the design of the high-level controller models. Similarly, the architecture design also should be aware of control performance and delay constraints. In other words, techniques from different layers of design abstraction should be combined together rather than designing these layers independently of each other. Such *cross-layer* design approaches are also discussed later in the following section.

So far, our discussion was based on the assumption that all the components in the platform architecture function *correctly*. From our previous reliability-related discussions, we know that this assumption is increasingly not true. To cope with an unreliable implementation platform, the cross-layer design approach outlined

above has to be extended to make the reliability information from the architecture level visible at the software and system levels. This way, critical parts of the computation/communication may be appropriately replicated or protected, and certain software or system-level functions may be designed to be more robust to architecture or device level failures/errors.

3.3 Interaction Between Design Layers

Protecting an IC against reliability challenges purely on individual layers of the design hierarchy is increasingly considered to be very costly and not the most efficient solution. E.g. hardening an IC against SEUs could be done by hardening each memory cell (larger transistors, 8T cell), or by providing redundancy and/or ECC or by computing checksums in SW. However, a growing consensus is evolving that the most cost-efficient techniques for analyzing and optimizing against reliability challenges involve cooperation between multiple design layers [21]. Major research projects are under way in Asia, Europe and the US to evaluate cross-layer techniques in addressing reliability (e.g. [25], [26] for an effort in Germany).

"Cross-layer" refers to the idea that efforts on different levels (layers) of the design hierarchy are combined to achieve an overall optimal tradeoff between required resources and resulting improvement in design quality. Cross-layer approaches can be employed both during IC design and during IC operation.

An example for cross-layer optimization during the design phase of an IC is protection against SEUs in logic. This can be achieved by hardening all flipflops. Possibly a more efficient cost-benefit tradeoff can be achieved by analyzing the design, identifying the most mission-critical flipflops (however this term might be defined) and then hardening only those. RAZOR or similar techniques might be implemented. Alternatively, entire critical modules could be replicated (DMR, TMR).

During IC operation, the operating system could employ various techniques for error detection (e.g. computation of checksum) and error recovery (e.g. checkpointing and roll-back). There are obvious tradeoffs here for the resources and the time required for both error detection and recovery. To which degree real-time requirements need to be fulfilled plays an important role in deciding which techniques to apply.

A specific example for cross-layer considerations is to consider how the execution of different instructions of a processor influences the results of circuit timing analysis [27]. This knowledge, gained during the design phase, can be used in multiple ways:

- change the design such that the most critical instructions become less critical (likely at the expense of IC area)
- supply the compiler with information about criticality of instructions such that this information can be considered when deciding which processor instructions to use in compiling a program
- finally, consider making changes to a program during its execution if e.g. aging of an IC is sensed and a risk of a certain instruction failing soon appears.

In general, the more information about an application is available, the more specific (and therefore cost-efficient) approaches can be chosen to improve reliability. Typically, in an embedded system such as most automotive ECUs are, the environment is more constrained than in a general purpose CPU. If higher layers of the design hierarchy have information on (i) usage frequencies, and (ii) reliability requirements, then this information may be used for design and analysis at the lower layers of design.

On the other hand, e.g. reliability requirements at the device layer will imply thermal constraints (e.g., changes in thermal profile). These thermal constraints will in turn imply constraints on higher layers, e.g. on task mapping, task migration, and in the future dynamic frequency scaling.

4. RELIABILITY IN THE AUTOMOTIVE CONTEXT

Aging is becoming a major reliability concern especially for automotive electronics as the automotive environment poses specific challenges. In many consumer markets, electronics-based products are used only a few years before they are discarded – often before aging degradation becomes relevant. In automotive, IC manufacturers need to guarantee specified functionality for 2-5 years operating time depending on application and temperature range and up to 15 years in standby mode, and desire them for even longer time to avoid reputational risks. At the same time, their ICs are sometimes used in very harsh conditions (e.g. temperatures up to 150°C and for special purposes also up to 175°C at reduced life times), and almost continuous (e.g. taxis being used in multiple shifts; battery management electronics in electrical vehicles) which amplifies the aging.

Another major concern are power saving methodologies. Up to now power was not a very big issue for automotive electronics due to the existence of a strong battery in the car, which was mainly used for the ignition process of the combustion engine. Early car electronics therefore did not utilize any low power design techniques. As the number of ECUs in a modern vehicle is now approaching 100, with a further increasing tendency, power meanwhile has become an important issue. Low power techniques as dynamic voltage scaling, several kinds of sleep and power down modes or even module switch-off techniques will be used more frequently with the drawback of increased sensitivity to reliability effects as described in Section 2 (see also Figure 2).

On top of that we still have to deal with different voltage domains in a car. This increases the design effort for the communication between these domains due to low power requirements on the one hand and reliability requirements like EMI and aging on the other hand. The biggest challenge in future will be to handle the complexity of this problem. EMI and ESD up to now have been optimized for each chip separately. This methodology will not be sufficient for above described systems due to the strong interaction of the components. A modeling of each component on an abstract level and according high level system simulations are mandatory to be successful.

As discussed previously, there exist different communication buses to connect the ECUs. Among these, in particular, the FlexRay protocol has gained wide acceptance for safety-critical domains as it provides the infrastructure to design reliable communication networks. We give some examples [34] of such reliability-related features in the following.

Each FlexRay controller provides two communication channels for redundant data transmissions and a hardware bus guardian for schedule monitoring. In terms of network topologies the protocol offers flexible solutions such as bus, star, and hybrid structures to design fault-tolerant and redundant backbone architectures. The FlexRay frame contains two CRC fields; an 11-bit header CRC,

and a separate 24-bit CRC which is calculated for the entire frame and able to detect up to five arbitrary bit errors during any frame transmissions. Further, eight samples per bit are available and a majority voting mechanism enables a filter for suppressing any glitches.

In addition to the above described protocol features, switched FlexRay networks and frame packing techniques have been recently studied in [37] and [40] with the goal to isolate *babbling idiots* and short circuits to single branches and to adopt frame retransmissions techniques for faulty frames, respectively.

One broad category of design approaches to cope with reliability issues could explore tradeoffs between the *accuracy* of computations and the associated *computation time*. Since most of the applications are various controllers (implementing, e.g., safety-critical, driver assistance or comfort related functionality), control performance depends on (i) the sampling period (which in turn depends on the computation time or the time taken to compute the control law for each sample), (ii) and the accuracy of the computed control signal. Depending on the chosen tradeoff between these two at the control design stage, the architecture could be suitably designed. For example, during the compilation phase, instructions could be chosen whose accuracy/reliability is higher but result in a less efficient code that runs slower. Alternatively, certain other choices of instructions may result in more efficient code (i.e., smaller running time) but the outcome of certain instruction execution is unreliable because of the possibly one or more reasons that were discussed previously. Such reliability-aware compilation techniques have been studied recently [35], but they have not been combined with higher levels of design abstraction such as the levels at which controller models are designed and analyzed.

Similarly, the above mentioned tradeoff analysis between sampling periods of control algorithms and the accuracy of the feedback control signal may be used to harden certain instructions whose reliable execution is essential for meeting control performance objectives. Again, such selective hardening of instructions along with custom instruction set design has been explored in the past from a reliability perspective [36] but has not been combined with model-based (control) algorithm design.

Finally, such tradeoffs between reliability and computation time may also be explored in the context of cache memories. Caches occupy more than half of the chip area in today's microprocessors and their reliability is therefore a critical design issue. Since the charge stored in a memory cell (such as an SRAM cell) decreases with each process generation, the accuracy of the information stored in the memory cells is becoming increasingly vulnerable to soft errors. Execution correctness characterization (such as those based on the AVF) have recently been refined using a number of models such as PARMA [38] and MACAU [39]. These models capture the effects of soft-errors such as single-bit upsets and temporal multi-bit upsets on a cache memory and the impact of protection/correction codes on the error incurred by a given application.

It will be meaningful to utilize these models in conjunction with code reordering and cache locking techniques. By reordering the code of a control application, the lifetime of various variables in the cache memory can be modified. On one hand this influences the execution time of the code and on the other hand it will influence the accuracy of the computed control signals (because of the change in the vulnerability of the application to soft-errors). Similar tradeoffs may also be explored by *locking* parts of the cache. Cache locking has been explored in the past for improving the *predictability* of real-time applications. While caches result in improving the execution time of programs in the average case, they also introduce significant variability in the execution time, thereby introducing jitter in some cases and making the computation of worst-case execution times (WCET) more difficult. Cache locking makes the computation of WCET more straightforward, thereby increasing the predictability of the application at the cost of deteriorating its average case performance. However, cache locking has not been commonly used for increasing *reliability*. By suitably analyzing the impact of sampling delay and accuracy on the performance of a control application, appropriate cache locking mechanisms will be useful in the context of designing safety-critical automotive control software.

5. SUMMARY

Advances in automotive technology are increasingly driven by electronics and software. As the amount of electronics in cars increases, and automotive electronics moves to advanced process nodes of 40nm and below, reliability concerns become a major issue. These are amplified in the automotive domain, as it is both very cost-sensitive and safety-conscious at the same time. In this paper, we describe major reliability challenges and discuss both established and emerging techniques to handle them. We especially point out the need for cross-layer optimization from transistor level all the way to software to conquer reliability challenges.

Acknowledgements

This work was supported partially by the German Research Foundation (DFG) as part of the priority program "Dependable Embedded Systems" (SPP 1500 – http://spp1500.itec.kit.edu).

6. REFERENCES

[1] E. Frickenstein. Mikroelektronik fährt BMW ConnectedDrive. In *3. Symposium Mikroelektronik*, Berlin, Germany, September 25-25, 2012.

[2] S. R. Nassif. Modeling and forecasting of manufacturing variation (embedded tutorial). In *ASP-DAC*, 2001.

[3] D. Boning and S. R. Nassif. Models of process variations in device and interconnect. In *Design of High-Performance Microprocessor Circuits*, A. Chandrakasan, Ed. Piscataway, NJ: IEEE Press, 2000.

[4] D. Blaauw, et al. Statistical timing analysis: from basic principles to state of the art. IEEE Transactions on CAD, 27 (4): 589-607, Apr. 2008.

[5] K. Kuhn et al. Managing process variation in Intel's 45nm CMOS technology. In *Intel Technology Journal*, 12(2): 131-144, June 2008.

[6] S. R. Nassif, N. Mehta, and Y. Cao. A resilience roadmap. In *DATE*, 2010.

[7] S. R. Nassif, V. B. Kleeberger, and U. Schlichtmann. Goldilocks failures: not too soft, not too hard. In In *Proc. of Reliability Physics Symposium*, 2011.

[8] K. Bernstein et al. High-performance CMOS variability in the 65-nm regime and beyond. In IBM Journal of Research and Development, 50(4.5): 433-449, 2006.

[9] S. R. Nassif. Modeling and analysis of manufacturing variations. In *CICC*, 2001.

[10] D. Lorenz, M. Barke, and U. Schlichtmann. Aging analysis at gate and macro cell level. In *ICCAD*, 2010.

[11] K. N. Tu. Recent advances on electromigration in very-large-scale-integration of Interconnects. In. J. Appl. Phys, 94(9), September 2003.

[12] N. Weste and D. Harris, CMOS VLSI Design: A Circuits and Systems Perspective, Addison-Wesley, 2009.

[13] Synopsys PrimeTime Advanced OCV Technology

[14] H. Graeb. *Analog Design Centering and Sizing*. Springer, 2007.

[15] K.-C. Wu and D. Marculescu: Aging-aware timing analysis and optimization considering path sensitization. In *DATE*, 2011.

[16] D. Lorenz, M. Barke, and U. Schlichtmann. Efficiently analyzing the impact of aging effects on large integrated circuits. In *Microelectronics Reliability* 52(8), 1546-1552, August 2012.

[17] M. Alam, B. Weir, and P. Silverman. A study of soft and hard breakdown - Part II: Principles of area, thickness, and voltage scaling. In *IEEE Transactions on Electron Devices*,. 49(2): 239 –246, February 2002.

[18] R. Degraeve et al. Relation between breakdown mode and location in short-channel nMOSFETs and its impact on reliability specifications. In *Proc. of Reliability Physics Symposium,* 2001.

[19] R. Rodriguez et al. The impact of gate oxide breakdown on SRAM stability. In *IEEE Electron Device Letters*, 23(2), September 2002.

[20] B. Kaczer et al., Gate oxide breakdown in FET devices and circuits: from nanoscale physics to system-level reliability. In *Microelectronics Reliability*, 47(4-5), April-May 2007.

[21] N. P. Carter, H. Naeimi, and D. S. Gardner. Design techniques for cross-layer resilience. In *DATE*, 2010.

[22] Y.-H. Lee et al. Prediction of logic product failure due to thin-gate oxide breakdown. In *Proc. of Reliability Physics Symposium*, 2006.

[23] M. A. Alam et al. Statistically independent soft breakdowns redefine oxide reliability specifications. In *Proc. of Int. Elec. Dev. Meeting (IEDM),* 2002.

[24] H. Matsuyama et al.: Investigation of stress-induced voiding inside and under vias in copper interconnects with “wing” pattern. In *Proc. of Reliability Physics Symposium*, 2008.

[25] J. Henkel et al. Design and architectures for dependable embedded systems. *International Conference on Hardware/Software Co-design and System Synthesis (CODES+ISSS)*, 2011.

[26] A. Herkersdorf et al. Cross-layer dependability modeling and abstraction in systems on chip. In *Workshop on Silicon Errors in Logic - System Effects (SELSE),* 2013.

[27] V. B. Kleeberger et al. Program-aware circuit level timing analysis. In *International Symposium on Integrated Circuits (ISIC)*, 2011.

[28] M. Choudhury et al. Analytical model for TDDB-based performance degradation in combinational logic. In *IEEE Design, Automation, and Test in Europe (DATE),* 2010.

[29] R. Hartl et al. Architectural vulnerability factor estimation with backwards analysis. In *13th Euromicro Conference on Digital System Design*, 2010.

[30] S. A. Seshia, W. Li, and S. Mitra. Verification-guided soft error resilience. In *DATE* 2007.

[31] D. Ernst et al. RAZOR: A low-power pipeline based on circuit-level timing speculation. In *Micro-36*, 2003.

[32] M. Wirnshofer et al. On-line supply voltage scaling based on in-situ delay monitoring to adapt for PVTA variations. In *Journal of Circuits, Systems, and Computers*, 21(8), 2012.

[33] S. Mukerjee et al. Measuring architectural vulnerability factors. In *IEEE Micro*, 23(6): 70-75, 2003.

[34] Rausch, M.. FlexRay: Grundlagen, Funktionsweise, Anwendung. In Carl Hanser Verlag GmbH & CO. KG.

[35] S. Rehman et al. Reliable software for unreliable hardware: Embedded code generation aiming at reliability. In *CODES + ISSS*, 2011.

[36] U. D. Bordoloi et al. Reliability-aware instruction set customization for ASIPs with hardened logic. In *RTCSA*, 2012.

[37] B. Tanasa et al. Reliability-aware frame packing for the static segment of FlexRay. In *International Conference on Embedded Software (EMSOFT)*, 2011.

[38] J. Suh, et al. Soft error benchmarking of L2 caches with PARMA. In *SIGMETRICS*, 39(1), 2011.

[39] J. Suh, M. Annavaram, and M. Dubois. MACAU: A Markov model for reliability evaluations of caches under single-bit and multi-bit upsets. In *HPCA*, 2012.

[40] P. Milbredt et al. Switched FlexRay increasing the effective bandwidth and safety of FlexRay networks. In *15th International Conference on Emerging Technology and Factory Automation (EFTA), 2010.*