The definite version is published as B. Steinwender, S. Einspieler, M. Glavanovics, and W. Elmenreich. Distributed power semiconductor stress test & measurement architecture. In Proceedings of the 11th International Conference on Industrial Informatics, pages 129–134. IEEE, 2013. (doi:10.1109/INDIN.2013.6622870)

Distributed power semiconductor stress test & measurement architecture

Benjamin Steinwender*†, Sascha Einspieler*†, Michael Glavanovics†, Wilfried Elmenreich*‡
*Mobile Systems Group/Lakeside Labs, Alpen-Adria-Universität Klagenfurt, Austria
†KAI Kompetenzzentrum Automobil- und Industrie-Elektronik GmbH, Villach, Austria
‡Complex Systems Engineering, University of Passau, Germany
Email: benjamin.steinwender@k-ai.at

***Abstract*—Conventional reliability testing of microelectronic power devices requires dedicated test systems. In order to test a statistically meaningful set of devices, only simplified stress pattern generation through a centralized controller is performed due to cost restrictions. Knowledge about device performance and failure time is commonly obtained by periodically removing the device from the test setup and performing a measurement on a different test hardware.**

In this paper, we propose a distributed power semiconductor stress test and measurement architecture to overcome limitations of existing test systems. We show that a local smart controller close to the tested device reduces the centralized system complexity by dividing the reliability testing problem into smaller tasks.

Keywords: Reliability testing, Distributed measurement, Smart Transducer, Automotive electronics, CAN

I. INTRODUCTION

Increasing system integration and miniaturization of power semiconductors lead to high electrical, thermal and mechanical stress during operation. As a result, the structure of the power transistor degrades over lifetime and its performance suffers. Therefore, stress testing of power devices prior to production release is required by customers and reliability standards (AEC Q100-012 [1]).

Reliability testing such as HTOL (high temperature operating life) or repetitive short circuit require several devices to be tested in order to obtain relevant results for statistical lifetime estimations. Conventional automated test equipment (ATE) is able to apply test patterns to the devices under test (DUT), but usually only a small number of devices can be tested simultaneously. Using a dedicated ATE for each single device is far too expensive. ATE systems are not cheap, thus climate chambers or racks are equipped with sockets to hold several DUTs [2]. Limited interface bandwidth allows basic control of the devices, typically without feedback. This poses a problem when we want to know about the actual performance of the devices or their time of failure.

We propose a paradigm shift towards a distributed power semiconductor stress test and measurement architecture which overcomes the limitations of existing test systems by employing smart controllers close to the DUT, bias, load and protection modules. The smart controller can interact locally with the DUT using an arbitrary number of parallel signals and dedicated interfaces while the communication channel through the outside storage can be kept physically small through a shared serial data bus (see Figure 1).

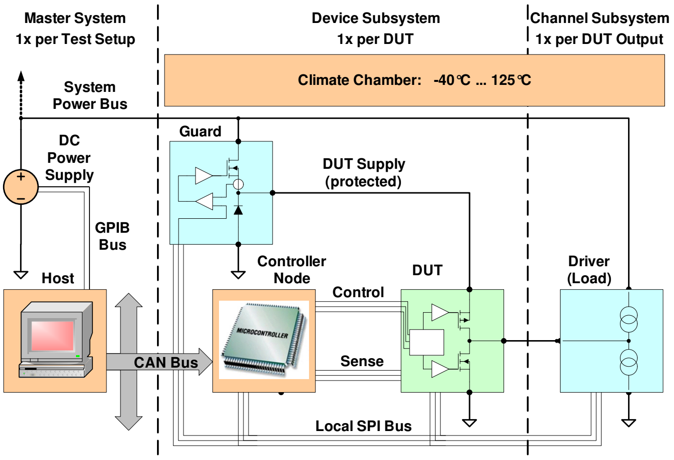


Figure 1. Modular Power Stress test –a controller located next to the DUT promises extended control and diagnosis in contrast to a centralized design

The purpose of this paper is to motivate the design decision towards a distributed architecture of smart controllers and to propose and discuss a system architecture for such a modular stress test system. The paper is organized in the following sections: first, we present the main motivation for the paradigm shift in section II. Then we discuss the state-of-the-art on smart controller architectures and existing reliability testing equipment in section III. In section IV we describe the proposed system architecture for a modular stress test system. The description and results of a prototype implementation can be found in section V. The paper concludes with a summary in section VI.

II. MOTIVATION

A shift towards a distributed architecture with smart controllers is expected to increase modularity, enable local, more fine-grained control and diagnosis of DUTs and to relax the global network communication requirements:

The particular advantages of increased modularity are:

- Enable DUT-specific tests and measurements

- Provide different load conditions for each DUT
- Reduce development time and cost, since tests can be added gradually to the system

Local, more fine-grained control and diagnosis of DUTs allows:

- to minimize number of global connections, since all local connections to the DUT are made via the smart controller
- to increase the number of local signals, since this is only limited by the smart controller implementation, not by the general network architecture
- to improve the real-time behavior and the data rate for DUT control and measurement
- to optimize analog signal quality due to short distances for analog lines

Furthermore, in a distributed architecture of smart controllers, the network communication requirements are relaxed:

- possibility to pre-analyze, reduce, and compress data locally
- removal of real-time load from host
- local storage of data until the network channel is available

Finally, we expect an overall cost reduction since the relaxed communication requirements can be fulfilled by a low cost communication system like standard CAN.

III. Related & Previous Work

The availability of small, compact, and affordable micro controllers has advanced the development of smart transducers [3].

These incorporate sensors or actuators, so the main concept at this stage still relies on measuring or actuating purposes which still require a real-time connection to a, possible centralized, controller. Therefore, many architectures emphasize the real-time capabilities of the communication system [4]. With increasing capabilities of a node, the smart devices become smart controllers being able to locally control a system part autonomously. An overview for control and management methods of such smart devices is given in [5].

In the past years, effort has been put into designing universal test equipment. As an example, the test system [6] is able to test the short circuit performance of protected smart power switches according to the AEC Q100-012 standard. The system features individual DUT control, integrated diagnosis, shutdown of failed devices and limited in-situ parameter measurement capabilities. The system [7], [8] allows emulating repetitive switching of inductive loads through a controlled current source, including failure detection, automatic shutdown and continuous voltage / current monitoring.

In these systems, multiple DUTs are subjected to stress test patterns under automotive environment conditions. Device status as well as parametric data are stored for further evaluation. However, both system concepts rely on a single central control instance, so a tremendous amount of hard-wired signal lines is required. In addition, these two systems are dedicated to a special test purpose and limited in additional capabilities.

In [2], a description of an HTOL reliability infrastructure for testing bipolar transistors is given. A small number of devices are tested on multiple systems. In-situ measurement configuration allows recording of several characteristics without removal of the DUTs from the test system. No details are given about the software architecture.

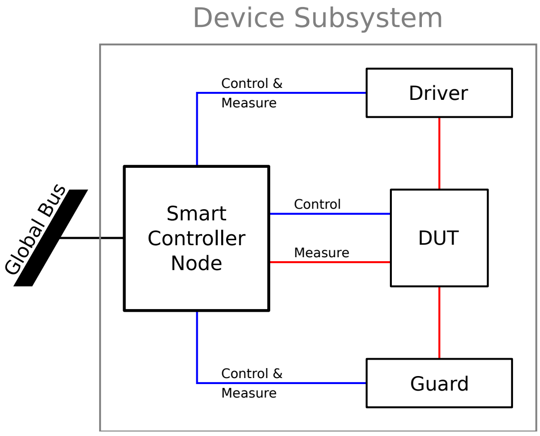


Figure 2. Device Sub-System –the Smart Controller Node steers Driver, Guard (Protection) and DUT

A modular, but still easily configurable test system has been described in [9], [10]. The test system and the test plan are described via XML configuration files. Upon loading the test definition, it is verified against the present hardware and modules and connections are configured through a hardware multiplexer according to the desired test setup. Such an approach simplifies the user interaction and errors due to erroneous user input are avoided. However, this system is only able to test one device at a time.

Therefore, a test system architecture is required that is not only able to stress a reasonable amount of devices simultaneously, but is also flexible so that different device types can be appropriately tested.

IV. Modular Test System Architecture

We introduce a novel concept for power reliability testing: the modular test system architecture consists of three main parts as shown in Figure 1. A variable number of smart controller nodes are located close to the DUTs. The nodes are connected via a global communication interface. One bus participant, namely the host, acts as a graphic user interface (GUI) for loading test definitions and displaying device status information.

In subsection IV-A, the controller responsible for testing a single DUT is described. The communication interface and the central host software are described in subsection IV-B and subsection IV-C respectively.

A. Controller node

The smart controller node is the main component for testing the DUT. It is placed as the central instance of a so-called *device sub-system* (Figure 2) which also contains a *driver* and a *guard* module.

The driver provides application specific physical or emulated load conditions, whereas the guard ensures safe operation of

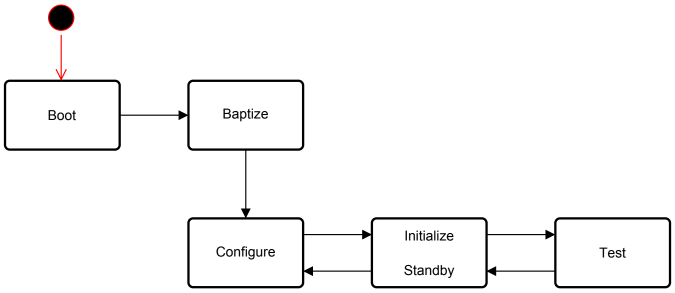


Figure 3. Controller State Diagram

the sub-system and may shutdown the DUT if it fails. The smart controller node communicates with the driver and guard modules via a local SPI (serial peripheral interface) bus in order to transfer parametric configuration and measurement data. The idea behind the modular approach is that driver and guard may be selected appropriately for the tested device and stress definitions while parts of the system such as the controller may be re-used for different device types.

The node provides an interface to apply test patterns to the DUT via serial and parallel signals. In addition, physical quantities such as voltage, current and temperature may be measured to observe and record the behavior of the DUT.

The smart controller node uses the global communication interface (subsection IV-B) to report to the host. By sending the node's capabilities to the host, the host is able to verify the loaded test plan against the physically present test setup. Further, test configuration data is sent via this global bus and the node also reports all recorded behavior and measurements from the tested device. Device measurements can be preprocessed directly on the controller node to reduce the load on the bus and stored until the communication interface is ready for transfer.

Figure 3 gives a basic overview about the proposed controller firmware state machine structure.

Upon booting, the node performs a self test and identifies the connected driver and guard as well as the DUT if possible. In addition, the communication interface is initialized to enable further remote configuration.

In the *baptize* state, the node transmits its unique serial number to the host and receives a short address. From this point on, the node listens only to messages sent to its assigned short address.

In the *configuration* state, the node may receive requests from the host in order to report its capabilities. Further, test settings are loaded onto the node in this state.

When the node enters the initialization state, it configures its internal and peripheral modules and becomes ready for performing the reliability test procedure.

The *test* state features a simple sequencer that performs periodic tasks to stress the device and evaluate its feedback.

Each single arrow transition occurs automatically within the internal state machine. Once the node enters the configuration state, an operator may start and stop the test through the host. Therefore, transitions between the states *Configure*, *Initialize* / *Standby* and *Test* are requested by the host through the communication interface.

B. Communication interface

A serial data interface is used to connect the nodes and let them receive stress test patterns as well as report device status to a central data storage. The communication interface is selected according to several requirements. It should be possible to send single messages for configuring the controller nodes as well as receive parameter measurements at relaxed time intervals (about $1\,\mathrm{kB/s}$ to $10\,\mathrm{kB/s}$ per DUT). Optionally, device waveforms are requested at large time intervals (e.g. on device failure). Further, broadcast messages are desired in order to notify all bus members at the same time instant. In addition, the communication interface ideally provides synchronization features in order to sequence multiple controller nodes if they test high power DUTs.

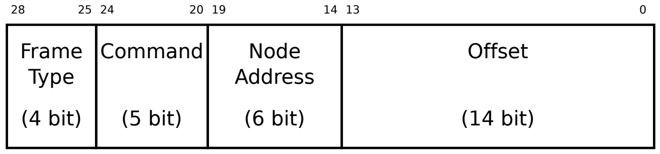
The interface also needs to support a harsh environment in the test system area while based on standard micro controller modules in order to use commercially available hardware. Further, hot plug capability of the nodes is desired, so the test operation needs not be interrupted when a single node is replaced.

Last, the communication interface should be able to be configured and programmed in a simple way and the used protocol stack should have little performance impact on the controller node.

Among some possible candidates, CAN (controller area network) [11] and switched Ethernet [12] fit well to our requirements. Ethernet provides high data rates but requires advanced micro controllers with built-in medium access control unit to fully utilize the increased bandwidth. The advantages of CAN are that many automotive controllers feature built-in CAN modules and our expected small message sizes keep the bus traffic at a reasonable low level.

CAN networks can be built in a bus topology using a carrier sense multiple access / collision resolution (CSMA/CR) mechanism. A CAN message basically consists of the message identifier, a control field, a variably sized data field (up to $8\,\mathrm{byte}$), the message check sum and an acknowledgment slot. Thus, it provides an identifier tag giving a meaning to the payload data as well as transmission consistency and immediate acknowledgment notification to the sender. With a bus speed of $1\,\mathrm{Mb/s}$ and a theoretical data throughput up to $56\,\mathrm{kB/s}$, it is well suited for configuring the controller nodes as well as transmitting measured data and device status. Since CAN is an event-triggered communication mechanism, several bus members may want to transmit data at the same time. Therefore, the identifier tag is used for priority based arbitration, where lower priority senders automatically back off the bus and retry again later.

Data transmission is limited to $8\,\mathrm{bytes}$ per message. Sending larger amount of data, like test configuration and DUT stimuli, requires higher layer protocols. CANopen [13] and DeviceNet [14] implement such protocols aimed for motion-oriented machine control and factory automation respectively. However,



28–25	24–20	19–14	13–0
Frame Type (4 bit)	Command (5 bit)	Node Address (6 bit)	Offset (14 bit)

Figure 4. CAN message identifier split up into four custom fields

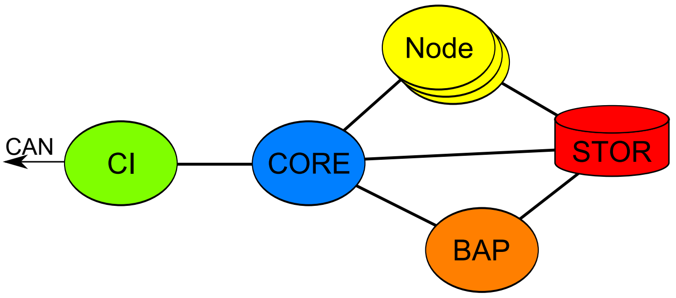


Figure 5. Software architecture

both protocols target small measurement nodes or simple actors, but not complex control and measurement systems like our proposed device sub-system. Therefore, we have implemented a protocol similar to CANopen, modified to suit our specific needs.

Extended CAN message types are used, where the identification part is further divided into four fields (see Figure 4): the four most significant bits are used to describe the frame type, which provides the possibility to increase priority for global system messages. Further, it is used to distinguish between node-to-host and host-to-node messages. The 5 bit command part is used to describe the intention of the message. Then, the 6 bit assigned address of the bus participant follows. Thus, up to 63 controller nodes can be connected, using the special address “0” for the host. Finally, a 14 bit file system offset is used to provide the address for writing and reading data to either configuration or measurement storage of the smart controller node. Specifying the address offset in the message identifier enables us to use all eight data bytes in the message payload area. Not all possible bit combinations of the frame type and command fields are used, so further extensions can be applied without changing the presented arrangement.

C. *Host*

The host is an independent bus member with additional features. Implementing service routines for such a test system requires a scalable, modular, extensible and simple software architecture. Thus, the central software has been created using the Actor Framework of NI LabVIEW [15] that is based on the actor model [16]. Five basic actors are used (see Figure 5):

- The communication interface (CI) provides access to the CAN bus by sending and receiving messages. These are forwarded to the central actor (CORE). Implementing the communication interface as a dedicated actor allows easier switching to a different network bus if necessary.
- The CORE actor utilizes a dispatcher to analyze the CAN message ID and relay the message to the dedicated actor instance. Messages from a controller node are relayed to the corresponding node actor. If a baptize request is received, the dispatcher relays the message to the baptize actor. Additionally, the core provides a GUI to the test system operator to load test definitions and start / stop the test procedure.
- The baptize actor (BAP) takes care of providing an unassigned short address to the newly inserted controller node. Therefore, all known unique identification numbers and the assigned addresses are stored in a database. Once a node has received its short address, this address is reserved for further use. To re-utilize occupied addresses of malfunctioning or replaced nodes, a least-recently-used strategy is employed. In addition, the system operator may delete the address reservation of an unconnected controller node.
- The node actor implements a virtual representative of the physical bus participant in order to handle the higher layer protocol through CAN messages. The actor stays in contact with the controller node by sending periodic status requests. Changes in either controller node status or DUT status are displayed and stored. DUT characteristics can be displayed dynamically depending on the test definition file.
- The storage actor (STOR) handles file and database read / write operations. The node, core and baptize actors may log events for offline evaluation.

Extensions to this software framework can be easily made by creating a new actor for a given task or by deriving it from an already existing instance and enhancing its capabilities.

V. HARDWARE PROTOTYPE

In order to verify the proposed system architecture, a prototype controller node has been designed (Figure 6) and manufactured (Figure 7). It features a 16 bit automotive micro controller as well as the guard and driver modules on the same PCB. The first device to be subjected to our new reliability testing architecture is a high complexity automotive system IC having on-chip processors, power converters and multiple analog (e.g. PSI5) and digital interfaces (SPI). Given the physical restrictions of the existing HTOL test system setup, eight DUTs are placed on the stress board. The connection to the node board is made via the signal interface. These signals include DUT supply voltages, digital test patterns and digital and analog sense lines to monitor the DUTs. The node board uses an additional connector (supply interface) for connecting to the power supplies and to the host system via the CAN bus. We used a data rate of $1\,\mathrm{Mb/s}$ which allows network lengths of up to $40\,\mathrm{m}$.

The guard module measures the DUT supply voltages and currents. The power supplies are controlled via the host system and shared by all node boards. Thus, the prototype only reports to the host if irregularities are detected. A future version of the

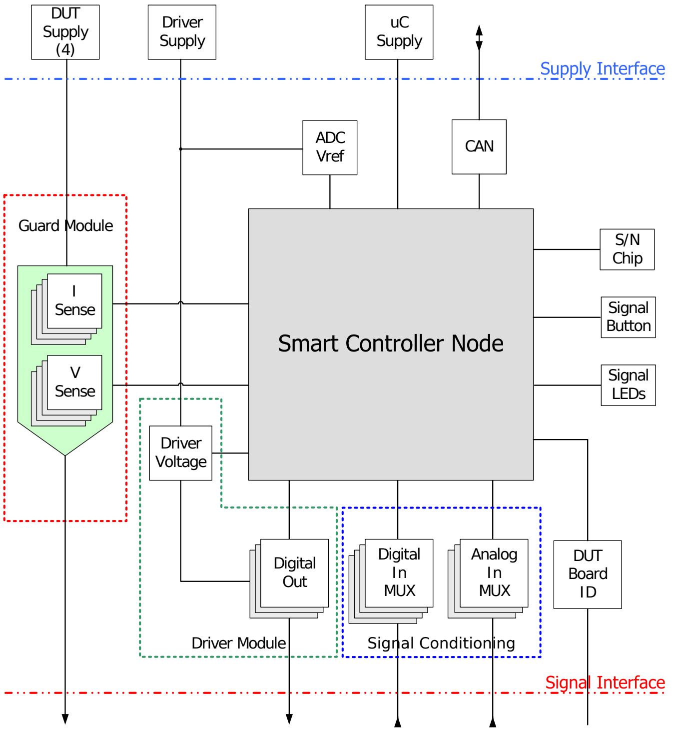


Figure 6. Fabricated prototype structure

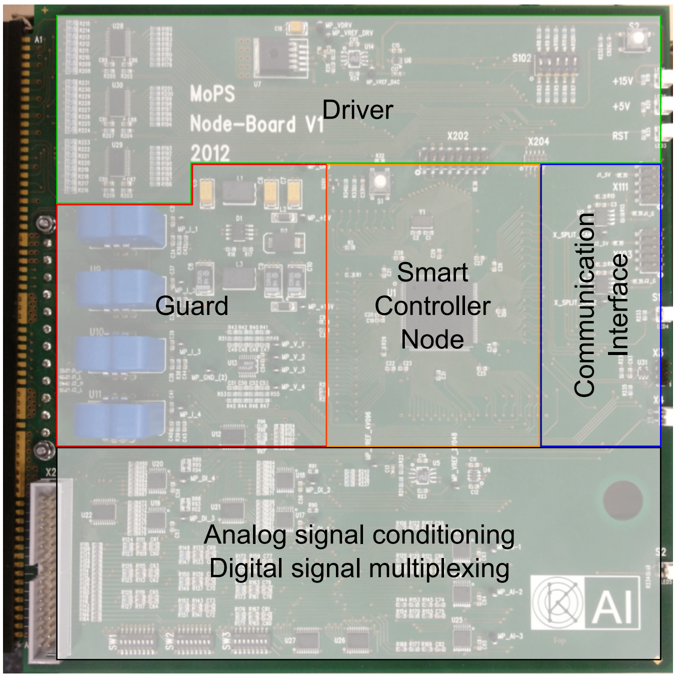


Figure 7. Smart controller node board

guard module may check threshold values itself and disable the supply for erroneous devices.

The driver module consists of the digital output drivers and a digitally controlled voltage regulator to adjust pin output voltages. This allows testing DUTs at their maximum rated voltage or interfacing low-voltage devices.

The analog and digital signal conditioning blocks contain multiplexers to reduce the vast amount of signals (7 channels x 8 DUTs) to a smaller number that can be handled by the smart controller. In addition, the analog signals can be scaled to the proper measurement range of the analog-to-digital converter.

Further serial interfaces to identify the currently applied DUT stress board and controller node are available as well as signal LEDs and a signal button for simple location of the node upon request by the operator or through the host software.

VI. Discussion & Conclusion

First laboratory tests with 5 prototype nodes showed sufficiently fast configuration time. The message structure for the test settings is 3776 bytes and can be transferred with 5 × 472 CAN messages. Making use of transmit buffers on the host, these messages are sent in less than 380 ms.

The maximum scenario would be having 63 nodes (according to the maximum available address space). Downloading the configuration on a single node takes about 75 ms, so all nodes are configured in less than 5 s. Due to the controllers' autonomy, downloading test definitions to a node is not time-critical and usually done only once per test procedure.

Node and device status is requested by the host once per second per node, so a total of 4 messages (2 × request + 2 × response) are transmitted per second per connected node. Again, worst case numbers for 63 nodes show less than 5 % bus load for diagnostic and status information.

Acquired measurement data however cannot be transferred that frequently, since the data structure size is 8.6 kB. Loading the data takes in average 6 s for one node due to limitations of the CAN driver on the host. Thus, measurements of 63 nodes can be read by the host software every 6.5 min. The large time intervals are not a problem, since we store measurements only every 15 min to 30 min in a life test setup running for 1000 h. In addition, a future version the controller node software may notify the host if a device has just failed and thereby trigger the transfer of the measurement data. Another possibility would be to move from CAN bus to switched Ethernet to increase the data rate.

In this paper, we have depicted limitations of existing reliability test systems and proposed a distributed test and measurement architecture. We have presented our design where the smart controller node is located in the device sub-system close to the DUT. Thus, the number of global connections is minimized and allows monitoring more local signals, immediate pre-processing and storage until transfer.

Test pattern definitions and measurement results are transmitted via a global bus connection using the controller area network. The smart controller nodes work autonomously based on their test definitions and report observed device abnormalities.

Our experiments with prototype implementations have shown that the instrumentation of the smart controllers with a low-cost communication system is feasible due to the relaxation of real-time constraints.

ACKNOWLEDGMENT

The authors would like to thank the reliability and product testing department of Infineon Technologies Austria AG for providing an interesting project topic. Fruitful discussions with KAI and Infineon colleagues are also greatly acknowledged.

This work was jointly funded by the Austrian Research Promotion Agency (FFG, Project No. 831163) and the Carinthian Economic Promotion Fund (KWF, contract KWF-1521|22741|34186).

REFERENCES

[1] Automotive Electronics Council, "AEC-Q100-012: Short Circuit Reliability Characterization of Smart Power Devices for 12 V Systems," 2006.

[2] K. Feng, L. Rushing, P. Canfield, and L. Flores, "Determination of reliability on MOCVD grown InGaP/GaAs HBT's under both thermal and current acceleration stresses," *2001 GaAs Reliability Workshop. Proceedings (IEEE Cat. No.01TH8602)*, pp. 159–180, 2001. [Online]. Available: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=995743

[3] W. Elmenreich and S. Pitzek, "Smart Transducers - Principles, Communications, and Configuration," in *Proceedings of the 7th IEEE International Conference on Intelligent Engineering Systems (INES)*, 2003, pp. 510–515. [Online]. Available: http://www.vmars.tuwien.ac.at/~wilfried/papers/2003/rr-10-2003.pdf

[4] W. Elmenreich, "Time-Triggered Smart Transducer Networks," *IEEE Transactions on Industrial Informatics*, vol. 2, no. 3, pp. 192–199, Aug. 2006. [Online]. Available: http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=1668078http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=1668078

[5] ——, "Configuration and Management of Networked Embedded Devices," in *Networked Embedded Systems*. Boca Raton, FL 33431, USA: CRC Press, 2009, pp. 21–22.

[6] M. Glavanovics, H.-P. Kreuter, R. Sleik, and C. Schreiber, "Cycle stress test equipment for automated short circuit testing of smart power switches according to the AEC Q100-012 standard," in *Proc. 13th European Conf. Power Electronics and Applications EPE '09*, Barcelona, 2009, pp. 1–7. [Online]. Available: http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=5278683

[7] M. Glavanovics, H. Köck, H. Eder, V. Kosel, and T. Smorodin, "A new cycle test system emulating inductive switching waveforms," in *2007 European Conference on Power Electronics and Applications*. IEEE, 2007, pp. 1–9. [Online]. Available: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=4417742

[8] M. Glavanovics, H. Köck, V. Košel, and T. Smorodin, "Flexible active cycle stress testing of smart power switches," *Microelectronics Reliability*, vol. 47, no. 9-11, pp. 1790–1794, Sep. 2007. [Online]. Available: http://linkinghub.elsevier.com/retrieve/pii/S0026271407003216

[9] A. Pirker-Frühauf, W. Gallent, M. Kunze, and G. Pelz, "Acceleration of IC verification process using advanced flexible modular measurement systems and software architectures," in *2008 IEEE Instrumentation and Measurement Technology Conference*. IEEE, May 2008, pp. 1845–1847. [Online]. Available: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=4547345

[10] A. Pirker-Frühauf, "A knowledge-based test program following the ATML standard," in *2009 IEEE AUTOTESTCON*. IEEE, Sep. 2009, pp. 195–199. [Online]. Available: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=5314090

[11] Robert Bosch GmbH, *CAN Specification 2.0*. Robert Bosch GmbH, 1991.

[12] R. M. Metcalfe and D. R. Boggs, "Ethernet: distributed packet switching for local computer networks," *Communications of the ACM*, vol. 19, no. 7, pp. 395–404, Jul. 1976. [Online]. Available: http://portal.acm.org/citation.cfm?doid=360248.360253

[13] O. Pfeiffer, A. Ayre, and C. Keydel, *Embedded networking with CAN and CANopen*. Copperhill Media, 2008.

[14] S. Biegacki and D. VanGompel, "The application of DeviceNet in process control," *ISA Transactions*, vol. 35, no. 2, pp. 169–176, Jan. 1996. [Online]. Available: http://linkinghub.elsevier.com/retrieve/pii/0019057896000225

[15] National Instruments, "Actor Framework." [Online]. Available: http://ni.com/actorframework

[16] C. Hewitt, P. Bishop, and R. Steiger, "A universal modular actor formalism for artificial intelligence," in *Proceedings of the 3rd international joint conference on Artificial intelligence*. Morgan Kaufmann Publishers Inc., 1973, pp. 235–245.