Line Redundancy in MVB-TCN Devices: A Control Unit Design

Juan Carlos Moreno, Eduardo Jesús Laloya and Jesús Navarro, *Member, IEEE*
Department of Electronic Engineering and Communications
University of Zaragoza
Zaragoza, Spain
jcmoreno@unizar.es

***Abstract*— TCN (Train Communication Network) standard was approved in 1999 by the IEC (IEC 61375-1) and IEEE (IEEE 1473-T) organizations to warrant a reliable train and equipment interoperability. TCN defines two serial buses: WTB (Wire Train Bus) and MVB (Multifunction Vehicle Bus), permitting double line attachments in both of them. Each attached device must take one of the lines as its trusted line and commute to the other, its observed line, if necessary. Its line redundancy control unit is responsible for these tasks.**

The MVB line redundancy control unit design here presented is, to our knowledge, the first one to be published. It is but one bit of the top-down full compliant MVB device design (class 1 to class 5) being developed by the authors to get a true reconfigurable system on chip.

Our proposal, based on concurrent and parametrical techniques, owns both a high speed performance and an easy reconfigurability.

Moreover, the performance of our design is only limited by the standard constraints themselves, being not restricted by our design criteria or our proposed implementation.

I. INTRODUCTION

This paper presents a synthesis oriented design for the line redundancy control unit of a TCN (Train Communication Network) MVB (Multifunction Vehicle Bus) device, according to the IEC 61375-1 [1], [2], [3] standard, also named IEEE 1473-T [4].

The TCN is a data communication network intended to connect programmable electronic equipment on-board rail vehicles for the support of traction and vehicle control, remote diagnosis and maintenance, and passenger information and comfort.

The TCN encompasses two serial master-slave buses: the Multifunction Vehicle Bus (MVB), which interconnects devices within a vehicle (or equipment within an inseparable group of vehicles), and the Wire Train Bus (WTB), which interconnects the vehicles in trains of variable composition, being capable of self-configuration.

The TCN is based on a two-level conceptual hierarchy including a WTB, interconnecting nodes in the different vehicles, and as many MVB buses as vehicles present in the train, interconnecting each of them the equipment within each different vehicle. The WTB and each MVB will be connected over a node acting as gateway (MVB class 5 device) (Fig. 1).

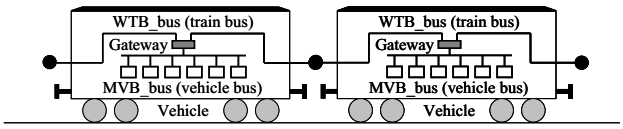

Figure. 1. WTB and MVB buses.

To support applications demanding a high reliability, the standard defines a redundancy scheme in which the bus may be double-line implemented. A double-line shall consist of two lines operating in parallel. Every device connected to a double-line bus shall identify the same line as "Line_A" and the other one as "Line_B" (Fig. 2).

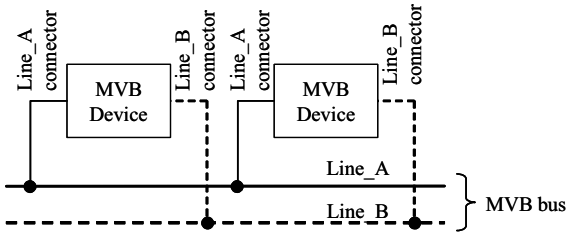

Figure 2. Single line and double line attachments.

The line redundancy principle assumes that a device transmits simultaneously the same data over both Line_A and Line_B, accepting only the data from one line, called

the Trusted_Line, while monitors the other line, called the Observed_Line. Each device chooses its Trusted_Line and Observed_Line independently from other devices.

TCN states six device types, or device classes, for the MVB bus: from class 0 to class 5. A class 0 device is a special device: the repeater. Slave devices are from class 1 to class 3. A class 4 device is a bus administrator which can become a bus master. Class 5 device is a gateway interconnecting the MVB to another bus, the WTB for example. Each device type has its specific capabilities.

TCN defines different device layers as ISO does. TCN also defines Real Time Protocols (RTP) covering both the data transfers between layers and the supervision of all of them, stating several interfaces related to the link layer: the LPI (Link Process data Interface), the LMI (Link Message data Interface) and the LSI (Link Supervision Interface) among them.

Each RTP contains some objects and procedures: objects are normally data structures storing useful data or procedure parameters, while procedures define actions.

The line redundancy control unit belongs to the link layer. Its operation depends on some contents of the MVB_Status and MVB_Control objects pertaining to the LSI (see next section).

The MVB line redundancy control unit design here presented is, to our knowledge, the first one to be published. It is but one bit of the top-down full compliant MVB device design (class 1 to class 5) being developed by the authors to get a true reconfigurable system on chip [5].

Being quite complex the behavioural conditions specified by the standard for double line attachments, we have widely exploited concurrent techniques to warrant simpler ways of design implementation and verification.

Our line redundancy control unit design takes also advantage of both top-down methodologies and parametrical techniques, owning a great speed performance and being greatly modifiable and reconfigurable to easily comply with any future standard changes.

Finally, it should be also noted that our proposal includes several author's solutions to complete some different options left open by the standard specifications.

II. LINE REDUNDANCY CONTROL UNIT

The Line Redundancy Control (LRC) unit is at charge of bus traffic control and trusted line selection. It receives signals from the encoder, the decoders and the link layer main controller to do these tasks.

According to those signals, the line redundancy control unit properly discriminates the Trusted_Line from the Observed_Line and transfers the data received on the Trusted_Line to some other units of the link layer (Fig. 3).

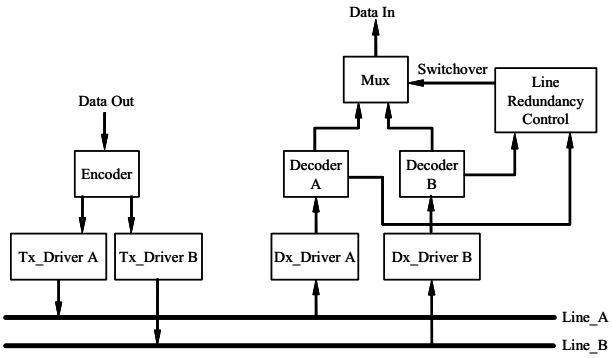

Figure 3. Encoder, decoder and line redundancy control architecture for a double line system.

The right operation of the LRC unit requires it accesses to four bits of the MVB_Control object ('sla', 'slb', 'cla' and 'clb') and to two bits ('LAT' and 'RLD') and three fields ("LineA_errors", "LineB_errors" and "Device_class") of the MVB_Status object (Fig. 4).

On the one hand, 'sla' and 'slb' values jointly determine the device line mode, i.e. single line attachment (the device is only connected to one line, Line_A) or double line attachment (the device is connected both to Line_A and to Line_B), and its management is application layer dependent. In double line attachments, 'sla' and 'slb' initial values fix the Trusted_Line at system initialization.

On the other hand, 'LAT' bit (Line_A Trusted) is set when Line_A is the Trusted_Line and 'RLD' bit (Redundant Line Disturbed) is set when the Observed_Line is disturbed. Their initial values depend on the current device line mode and they are updated by the link layer, namely by its LRC unit.

The fields "LineA_errors" and "LineB_errors" are memory mapped registers storing the number of transmission errors on, respectively, Line_A and Line_B. And obviously, "Device_class" field stores the device class parameters.

Finally, 'cla' and 'clb' bits, which are managed by the application layer, control the reset of LineA_error counter (each time 'cla' is set) and LineB_error counter (each time 'clb' is set).

According to all of its input signals, the LRC unit covers its different tasks, which can be structured in four main functions: device line mode selection, 'LAT' and 'RLD' management, switchover, and error counters management. They are described in the next four sections, being also detailed along them both the significance of the previously presented bits and fields in each LRC unit function and the input and output signals appearing in our design.

III. DEVICE LINE MODE SELECTION

The 'sla' and 'slb' bit values determine the device line mode according to the following criteria:

1) Condition 1.- 'sla' = 'slb' = '0'. The device is connected to a single line (Line_A, if nothing else specified). In a single-line attachment, a device shall consider the unused line as permanently disturbed, so 'RLD' shall be always set.

2) Condition 2.- 'sla' ='1' and 'slb' = '0'. The device is connected both to Line_A and to Line_B, but Line_A is always trusted, i. e. the line redundancy control unit must not switchover (see Section V) the lines, being always received the information present on the bus through Line_A.

3) Condition 3.- 'sla' ='0' and 'slb' = '1'. The device is connected both to Line_A and to Line_B, but Line_B is always trusted, i. e. the line redundancy control unit must not switchover the lines, being always received the information present on the bus through Line_B.

4) Condition 4.- 'sla' = 'slb' = '1'. The device is connected both to Line_A and to Line_B. At the start Line_A is the Trusted_Line. Afterwards the Trusted_Line is switched from Line_A to Line_B and vice versa according to the switchover rules explained in section V.

IV. 'LAT' AND 'RLD' MANAGEMENT

Each time a decoder detects the reception of a valid frame (a frame having its start delimiter, data channel code and end delimiter, all of them, correct), reports it to the LRC unit through its "Valid_frame_A" or "Valid_frame_B" signal (Fig. 4).

Depending on both the detection of valid/invalid frames in both lines and the values of the 'sla' and 'slb' bits, the 'LAT' and 'RLD' bit values shall be actualized as commented in the next paragraphs.

1) Condition 5.- 'sla' = 'slb' = '0'. There is no Line_B. At the start 'LAT' = '1' and 'RLD' = '1', Line_A is always trusted and the information on Line_B is thoroughly discarded. Along system operation, 'LAT' shall be switched to '0' if a no valid frame reception is detected on Line_A. 'LAT' register shall be switched back to '1' once a valid frame reception has been detected on Line_A. And so on.

2) Condition 6.- 'sla' ='1' and 'slb' = '0'. The device is connected to two lines, but only Line_A is trusted. At the start 'LAT' = '1' and 'RLD' = '1'. 'LAT' bit must evolve same as stated in condition 5.

3) Condition 7.- 'sla' ='0' and 'slb' = '1'. The device is connected to two lines, but only Line_B is trusted. At the start, must be LAT = '0' and RLD = '1'. Along system operation, 'LAT' remains at '0' as long as the Line_B decoder receives valid frames and that bit is switched to '1' once the Line_B decoder detects a no valid frame. A new valid frame detection on Line_B resets 'LAT', and so on.

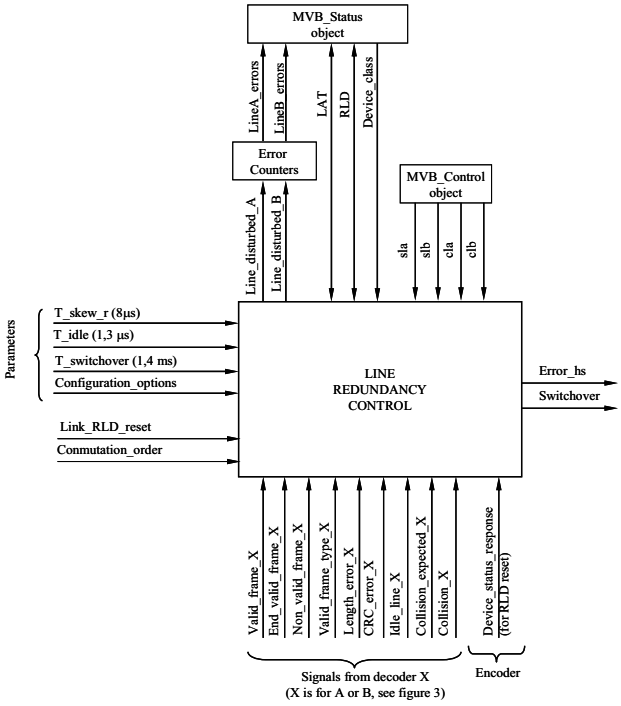

Figure 4. LRC unit: input and output signals.

4) Condition 8.- 'sla' = 'slb' = '1'. The device is connected both to Line_A and to Line_B. At the start, 'LAT' = '1' and 'RLD' = '0', being Line_A the Trusted_Line. Along system operation:

a) Condition 8a.- 'RLD' shall be set ('RLD' ='1') each time the LRC unit commands any switchover (see next section), so changing the Trusted_Line from Line_A to Line_B or vice versa. The 'LAT' bit value shall also be switched accordingly.

b) Condition 8b.- 'RLD' shall also be set ('RLD' ='1') each time the decoder of the Trusted_Line detects a valid frame and, since then, the decoder of the Observed_Line detects no valid frame along a time greater than T_skew_r, and no collision was expected on that frame. But, in this case, the 'LAT' bit value shall not be switched.

We must remark that T_skew_r is a system parameter (Fig. 4), 8 μs valued, stored in the link layer memory.

We must also remark that collisions are possible each time the bus master asks for general event or group event frames and such possibilities are reported through the "Collision_expected_A" or "Collision_expected_B" signals (Fig. 4). Collisions must be detected by the decoders,

reporting about them to the LRC unit ("Collision_A" or "Collision_B" signals in Fig. 4).

c) Condition 8c.- 'RLD' shall be reset ('RLD'='0') either: Each time a device status response slave frame has been sent by the encoder ("Device_status_response" signal in Fig. 4), or: Each time the link layer main controller requests it ("Link_RLD_reset" signal in Fig. 4).

V. SWITCHOVER

A device shall exchange its Trusted_Line and Observed_Line if and only if:

1) Condition 9.- No valid frame has been received on the Trusted_Line for a time greater than T_switchover (T_switchover = 1,4 ms) since the end of the last received valid master frame.

The detection of a no valid frame is reported by the decoders to the LRC unit (Fig. 4) by means of their "Non_valid_frame_A" and "Non_valid_frame_B" signals.

We note that T_switchover is a system parameter (Fig. 4) stored in the link layer memory.

It must also be noted that the TCN standard states two different start delimiters for master and slave frames. The checking of the received start delimiter is made by the decoders, reporting about it through "Valid_frame_type_A" and "Valid_frame_type_B" signals (Fig. 4).

2) Condition 10.- No valid frame has been received on the Trusted_Line for a time greater than T_switchover since the last switchover.

3) Condition 11.- The decoder of the Observed_Line detects a valid frame and the decoder of the Trusted_Line detects no valid frame within a time equal to T_skew_r, being no collision expected on that frame, having been idle the Observed_Line for a time longer than 2.0 BT (BT = Bit Time; 2.0 BT = 1.33 μs) from the end of its last received valid frame, and provided that the 'RLD' bit is not set ('RLD' = '0').

The end of a valid frame is detected by each line decoder, reporting such an ending to the LRC unit by means of the "End_valid_frame_A" or "End_valid_frame_B" signals (Fig. 4).

The idle condition of each line is also detected by its respective decoder, being reported through "Idle_line_A" or "Idle_line_B" signals (Fig. 4).

It must be noted that T_idle is a system parameter (Fig. 4) stored in the link layer memory.

4) Condition 12.- A valid frame has been received on the Trusted_Line, but the frame length or its check sequence(s) is(are) incorrect, provided that the 'RLD' bit is not set ('RLD' = '0'). This is an optional condition, to be or not applied depending on the particular configuration of each device.

Frame length is checked by the decoders, reporting about it to the LRC unit through the "Length_error_A" and "Length_error_B" signals (Fig. 4).

The decoders must also verify the CRC correctness, reporting about their checks to the LRC unit through the "CRC_error_A" and "CRC_error_B" signals (Fig. 4).

5) Condition 13.- A device sends its device status response slave frame, provided that the 'RLD' bit is not set ('RLD' = '0') in advance to that response. This condition is mandatory for class 1 devices but optional for any other class devices. That is why the LRC unit must receive information about its device class configuration: the "Configuration_options" signal (Fig. 4) covers that need.

6) Condition 14.- A device receives a switchover request from its link layer through "Commutation_order" signal (Fig. 4). This condition is not applicable to Class 1 devices.

The "Switchover" signal (Fig. 4) shall be activated each time a switchover must take place.

VI. ERROR COUNTERS MANAGEMENT

The LRC unit shall source two signals, one for each line, "Line_disturbed_A" and "Line_disturbed_B", to reckon their respective transmission errors (Fig. 4), storing their counting in the mandatory error counters.

A copy of these countings must be stored in the "LineA_errors" and "LineB_errors" fields of the MVB_Status object.

"Line_disturbed_A" and "Line_disturbed_B" signals shall be set or reset according to these criteria:

1) Condition 15.- It must be set each time its line decoder detects a collision, being no collision expected.

2) Condition 16.- It must also be set if a valid frame is received on the Observed_Line and no valid frame appears on the Trusted_Line within a time greater than T_skew_r since the valid frame was detected on the Observed_Line, and no collision was expected.

3) Condition 17.- It must be reset if a valid frame has been received on the Trusted_Line.

VII. LINE REDUNDANCY CONTROL UNIT DESIGN

Once briefly presented the behavioral blocks of the LRC unit and the input and output signals directly related to them (Fig. 4), we must give a short comment about one more signal, the "Error_hs" one. It jointly represents several signals supporting a handshake protocol, for error code

management purposes, between the LRC unit and the link layer main controller.

Our proposed design takes not only into account the single error condition specified by the standard, the skew error condition, i. e. the existence of a delay between the activation of the "Valid_frame_A" and "Valid_frame_B" signals higher than T_skew_r, but also checks for a lot of different error conditions: a collision is detected, being no collision expected; a valid start delimiter is detected, but a valid end delimiter is not received once elapsed a prefixed time; a valid CRC is received but its content is not correct; and so on. The implementation of these additional error conditions warrants a full covering of the options left open by the standard.

Similarly, the standard only states the general management of the 'sla' and 'slb' bits by the application layer, giving no precisions and recommendations about the procedures to do so.

In our design, the application layer not only initializes these bits but dynamically manages them in such a way that their evolution can be followed by the link layer, namely by the LRC unit, in its operation. This characteristic grants, if necessary, a real time control of the system by the user.

Moreover, in our proposal, the system parameters are not only stored in the link layer memory at system initialization, as it is mandatory in the standard, but dynamically managed by the application layer and/or the link layer main controller, as required, too.

This fact grants the dynamic reconfiguration of the LRC unit response in a consistent way, giving it much more flexibility and adaptability than the raw standard specifications imply.

We must also remark that our LRC unit synthesis design comprises some blocks added to the four functional ones presented in the previous sections, namely (Fig. 5): Memory Access Controllers 1, 2 and 3; Initialization Management; and Error Management. The presence of these additional subunits has eased and speeded up the design process and its verification.

Obviously, the four functional blocks are devoted to exhaustively implement the full set of conditions commented along the preceding sections. The implementation of each condition is normally based in a single state machine, but some of them require more than one state machine to be implemented.

The memory access controllers have been designed to warrant a concurrent, consistent and simultaneous access by the LRC unit, the link layer main controller and the application layer to the link layer memory and the MVB_Control and MVB_Status LSI objects.

This goal has been achieved using dual port RAM blocks to implement them (Fig. 5) and applying time multiplexing to some of their ports to speed up their access by our LRC unit.

The Initialization Management block is responsible for system parameters loading at system initialization, a task also speed up using direct memory access.

The Error Management block is responsible for the whole set of additional, not standardized, error conditions checking.

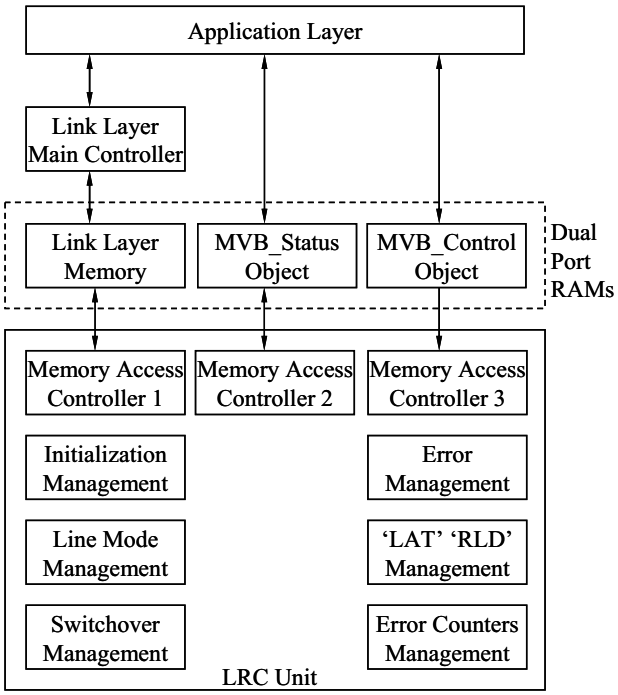

Figure 5. LRC unit: block synthesis design.

Finally, let us briefly comment about the state machines implementing the behavior conditions of our four functional blocks. We shall take the condition 11, being it a complex one, to do so.

Fig. 6 shows a simplified version of condition 11 main state machine (only its state names and main signals are shown).

Truly, three additional, not shown, state machines are involved in the exhaustive implementation of this condition, namely: the switchover state machine (it takes into account the current device line mode), the skew error condition state machine and the idle condition state machine.

The hierarchical and modular methodology supporting our design procedure has led us to a very fast design performance, being the speed of its response only limited by the standard specifications themselves, neither by the

architecture nor by the hardware technology implementing our LRC unit.

That is why our proposed design, retaining its actual architecture and implementing technology, shall correctly run on new, more restrictive, time specifications possibly included in future versions of the TCN standard.

It must be noted that our LRC unit design, our whole link layer design indeed, has been verified upon a Xilinx XC200E-6 PQ208 C FPGA, and also that a synthesizable VHDL description of it is currently available, but here not offered to cope with this paper length restrictions.

VIII. CONCLUSIONS

The LRC unit design here proposed, to our knowledge the first one to be published, is part of a link layer MVB-TCN device design leading to a reconfigurable system on chip. It is fully compliant with the mandatory specifications of the TCN standard but also takes into account its open options, covering its whole functional possibilities.

The top-down design criteria followed by the authors and the concurrent and parametrical techniques used to hardware implement it have led to a LRC unit, a MVB-TCN link layer indeed, with a very fast and flexible performance and having also a remarkable adaptability to future changes in the standard specifications, being mainly constrained by them not by our design and implementation criteria.

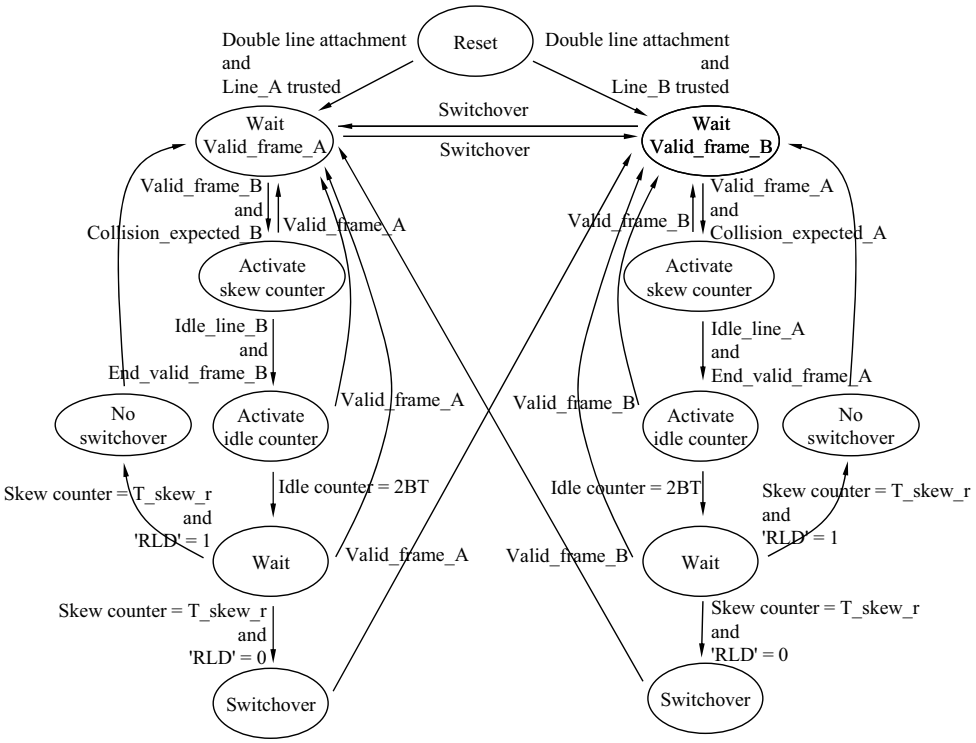

Figure 6. Condition 11: main state machine.

ACKNOWLEDGMENT

This design has been supported, from 2003 to 2005, by the project TIC2001-0062, funded by the 2000-2003 Scientific Research and Technological Development and Innovation Spanish Program.

REFERENCES

[1] International Electrotechnical Commission (IEC), "Electric railway equipment – Train bus – Part 1: Train Communication Network IEC 61375-1. Ed. 01," 1999.

[2] H. Kirrmann and P. A. Zuber, "The IEC/IEEE Train Communication Network," *IEEE Micro*, pp. 81-92, March-April 2001.

[3] C. Schafers, G. Hans, "IEC 61375-1 and UIC 556 – International Standards for Train Communication Network," in *Proceedings of the 51st IEEE Vehicular Technology Conf.*, Japan, May 2000, pp. 1581-1585.

[4] American National Standards Institute (ANSI), "Communication Protocol Abroad Trains – 1473-1999," 1999.

[5] J. C. Moreno, E. J. Laloya, and J. Navarro. "MVB Slave Devices: Link Layer and Real Time Protocols Functional Description and Design," Internal Technical Report. Group of Electronic Design in Digital Communications. University of Zaragoza, Jan. 2005.