Boosting robustness in CAN systems with new star topologies: CANcentrate and ReCANcentrate

Manuel Barranco[1], Julian Proenza[1], Luis Almeida[2]
{manuel.barranco, julian.proenza}@uib.es, lda@det.ua.pt

[1]DMI, Universitat de les Illes Baleares, Palma de Mallorca, Spain
[2]DET-IEETA, Universidade de Aveiro, Aveiro, Portugal

Abstract
Star topologies provide better error containment than bus topologies thus leading to more robust communication systems. This benefit, in several applications, compensates their typical higher cost. For example, the LAN domain has long moved to star topologies with Ethernet, a technology that is now progressing towards the industrial automation domain. Similar moves toward star topologies happened in the embedded systems domain, e.g., with TTP/C and FlexRay for in-vehicle systems. However, probably the most widely used network in distributed embedded systems, Controller Area Network (CAN), remained essentially a bus-only network. Recently, two star architectures were proposed for CAN, a simplex star (CANcentrate) and a replicated star (ReCANcentrate), the former aiming at increasing robustness for general distributed embedded systems and the latter aiming at safety critical applications. This paper discusses the benefits of star topologies in terms of error containment and then describes both architectures, showing how they achieve their aims and their performance.

Keywords
Error-checking, fault tolerance, network topology, centralized networks, buses, real-time and embedded systems, real-time distributed systems.

1 From field buses to (field) stars

The use of field buses in distributed control systems spread widely in all related application domains mainly due to their electrical robustness and low cost. One of the key causes of their low cost is the bus topology they rely on. However, the use of bus topologies implies some limitations regarding dependability. In a bus topology, components are attached to each other with scarce error-containment mechanisms. Thus, one single fault in any component (e.g., communication controller, transceiver, connector, wire, etc) of a network that relies on a bus may generate errors that can propagate throughout the communication subsystem leading to a generalized communication failure. Thus, a bus topology presents multiple single points of failure (Figure 1).

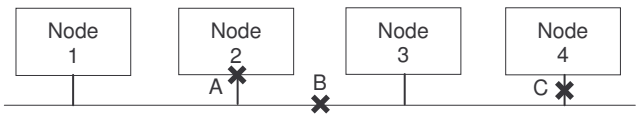

Figure 1. Example of errors that can propagate across a bus

 doi:10.1109/MC.2009.145

This characteristic is still observed when using replicated buses to provide fault tolerance or even bus-guardians[1] to increase error containment, or both together. The problem is the so-called *common-mode* failures, i.e., failures that simultaneously affect more than one component in the system, reducing or disabling its fault-tolerance or fault-detection mechanisms. For example, a failed node can send erroneous information to all the replicated buses it is attached to. Sometimes, the reason why a fault affects the replicated media is just their physical proximity. For example, if a mechanical action destroys a given node, it is likely that such action will also partition the replicated buses the node is connected to. These are known as *common-mode spatial proximity* failures. In case of bus-guardians, if they share components with the nodes they control, e.g., the clock oscillator or the power supply, one single fault in one of these components will affect equally the node and its bus-guardian and it is likely that certain faults will no longer be detected and isolated. Moreover, bus-guardians are ineffective with respect to faults generated by the transmission medium itself.

In contrast, star topologies may represent an effective solution to prevent the existence of multiple single points of failure. In a simplex star topology, each node is connected to a central element, the *hub*, by its own *link*. One advantage of a simplex star topology is that links only come into spatial proximity at the center of the star, thus the probability that different links suffer from common-mode failures is significantly reduced. But the most important advantage is that the center of the star, i.e., the hub, can be designed to have a privileged view of the system, knowing the transmissions from each node through the corresponding links. On the other hand, a star topology tends to generate more expensive cabling and still contains a single point of failure, i.e., the hub. Nevertheless, the cabling cost depends on the specific system layout and the probability of a single point failure can be attenuated using special techniques in the hub design and on its physical placement, or even eliminated using a replicated star topology. The extra cost must be weighted against the increase in robustness to determine whether the star topology is suited for a particular case.

In the Local Area Networks (LAN) domain, the ruling technology, i.e., Ethernet, has long adopted the star topology and users have willingly paid the extra cost for higher robustness. On the other hand, in distributed computer control systems, field buses have ruled because of their reduced cost. However, the growing use of Ethernet in the industrial automation area is already pushing star topologies into the typical field buses domain. Moreover, in safety-critical embedded systems, other technologies have also evolved to star topologies such as TTP/C [1] and FlexRay [2] for vehicle networks. These two technologies offer replicated star couplers (hubs), for fault-tolerance, and a bus-guardian inside each hub that prevents a node from transmitting in the wrong instants.

2 The case of CAN

A network technology that has become extremely popular in distributed embedded systems and particularly for control applications is Controller Area Network (CAN). This technology was created in the early 80s by Bosch GmbH, Germany, and soon after became an ISO standard [3]. The success of CAN is mainly due to its low cost, simple configuration, electric robustness, prioritized medium access arbitration mechanism and error detection and containment features. These properties are essentially related to the technical options of using in-bit response together with dominant-recessive bit values. The former means that the transmission of a bit traverses all the network and electrically stabilizes and only then the next bit can be transmitted. The latter means that the dominant bit ”0” prevails over the recessive one ”1” so that the medium implements a wired-AND function of all nodes contributions.

[1] A bus-guardian is a device that controls the transmissions of a node and blocks them in case they are not considered legal both in the time domain, e.g., transmissions carried out in the wrong instant, and in the value domain, e.g., transmissions using wrong data lengths.

On one hand, these features provide a bit-wise deterministic collision resolution mechanism. All nodes willing to transmit synchronize on the start-of-frame bit and enter the *arbitration phase* transmitting a node-unique bit sequence called *identifier* while observing the actual bits on the bus. If sending a "1" a node observes a "0" it backs-off and retries after the current frame. At the end of this phase only the node with the lowest identifier continues transmitting. On the other hand, the in-bit response also supports in-bit detection of bit-stream errors [3] namely *bit, format*, *CRC, stuff*, and *ACK errors.* These names are self-explanatory apart from the *bit error*, which occurs when a recessive bit is observed even though a dominant bit is being transmitted. Each node detecting an error globalizes it in the following bit by transmitting an *error frame*, i.e., six consecutive dominant bits. This causes a *stuff error* (more than five similar consecutive bits) thus compelling all other nodes to detect and globalize an error too and, consequently, to abort the transmission/reception of the frame that is currently being transmitted. In order to isolate a faulty node that generates errors, each node includes a *Transmission Error Counter* (TEC) and a *Reception Error Counter* (REC), which are increased and decreased following some specific rules [3]. When any of these counters exceeds a given threshold, a node diagnoses itself as faulty and disconnects itself from the network (*bus-off state*). Nevertheless, these error handling and isolation mechanisms are still limited in efficiency due to the bus topology of CAN, as referred before, even with replicated buses [4, 5] and bus guardians [6].

Therefore, star topologies for CAN have also been proposed [7-11]. Some of them are *passive stars*, just addressing electrical signal transmission issues, such as impedance adaptation [7]. These stars present important disadvantages [12] concerning coupling losses, strong limitations on the star radius or in the bit rate, electrical problems, etc. Other stars are known as *active stars* [8-11], regenerating the incoming electrical signals and overcoming some of the technical problems of passive stars. Most of these stars rely on an active star coupler that receives the incoming signals from the nodes bit by bit, implements a logic AND, and retransmits the result to all nodes.

Unfortunately, these passive and active stars either do not address fault confinement, or only deal with a small set of possible faults. Moreover, some of them are not even fully compatible with the CAN protocol [12]. However, the hub of a CAN-compatible star can be designed to exploit its privileged view of the system, since it can easily know the contribution from each node, bit by bit, through the corresponding link. Thus, such a hub can enforce a prompt confinement of faulty transmission media and faulty nodes by disconnecting the respective hub ports, bringing the desired increase in global system robustness that this topology may offer.

This has been the motivation for designing two recent CAN-compliant star topologies, CANcentrate [12] and ReCANcentrate [13], that were specifically devised for error-containment and fault tolerance. These stars, described in the following sections, and referred to as *(Re)CANcentrate* for short, provide dependability features in CAN that are similar to the ones offered by protocols such as TTP/C and FlexRay. Moreover, both stars are fully compatible with commercial off-the-shelf (COTS) CAN components, with CAN applications and CAN-based protocols, e.g., CANopen, DeviceNet, or FTT-CAN. This compatibility also allows (Re)CANcentrate to keep all the good dependability properties already provided by CAN, e.g., in-bit response and the error signaling mechanisms.

3 Rationale behind CANcentrate and ReCANcentrate

The main objective of (Re)CANcentrate is to boost robustness and reliability in CAN networks by means of fault treatment (fault diagnosis and fault passivation) and fault tolerance.

To better understand the objective of these stars in terms of fault treatment, the following concepts were introduced [12]: *severe failure of communication*, i.e., when more than one node cannot communicate; and *point of severe failure*, i.e., a point whose failure is *severe*, which comprises the common concept of single point of failure. When analyzing CAN buses, the following faults may lead to severe failures:

- Stuck-at-dominant and stuck-at-recessive faults, either in the nodes or medium, arising from e.g., short circuits to ground or battery, or malfunctioning or isolated controllers.
- Medium partition faults that occur whenever the network is physically broken into several sub-networks called *network partitions*.
- Bit-flipping faults that occur whenever a network component, either node or medium, exhibits a *fail uncontrolled* behavior, sending random erroneous bits with no restrictions in value or time.
- Babbling idiot faults that occur whenever a node sends syntactically correct frames that are erroneous in the time domain, causing undesired interference.

(Re)CANcentrate deals with Physical Layer faults that are independent of the application, i.e., the first three types of faults outlined above. No assumptions are made concerning the location, frequency and duration of errors that may occur as a consequence of such faults. Additionally, a guardian could easily be included in the hubs of (Re)CANcentrate to confine babbling-idiot faults, but this is not currently considered.

The hub of CANcentrate isolates any faulty network component, e.g., cable, transceiver, etc, at the corresponding hub port, thereby preventing error propagation and thus the occurrence of a severe failure. Therefore, CANcentrate improves fault treatment in CAN by reducing the multiple points of severe failure exhibited by any other network based on a CAN bus to a unique single point of failure, i.e., the hub.

In some applications, the degree of dependability achieved by CANcentrate could be not enough and the presence of a single point of failure unacceptable. In these cases, spatial redundancy at the hub level is required so as to tolerate permanent hub faults. ReCANcentrate provides this redundancy by using a replicated star topology that includes two or more hubs. Besides providing the same capacity of error containment as CANcentrate, ReCANcentrate further tolerates hub and link faults.

4 CANcentrate and ReCANcentrate basics

In CANcentrate, each node is connected to the hub by means of a dedicated link that contains an uplink and a downlink (Figure 2). The hub receives each node contribution through the corresponding uplink, couples all the contributions and broadcasts the result through the downlinks.

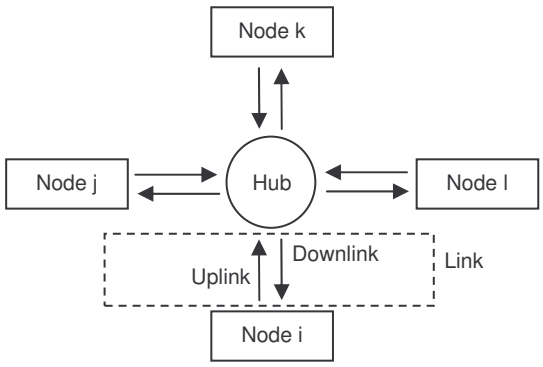

Figure 2. Connection schema of CANcentrate

Figure 3 shows the internal hub hardware architecture [12]. It contains three main modules: the Coupler Module, the Input/Output Module and the Fault Treatment Module. The Coupler Module takes into account each port contribution ($B_{1..n}$), couples them with a logic AND generating the

resulting signal B_0, which is broadcast back to the nodes. The AND gate replaces the wired-AND performed by a CAN bus, thus B_0 is similar to what would be observed on a fault-free bus. Hereafter we will call *resultant frames* to those obtained from B_0.

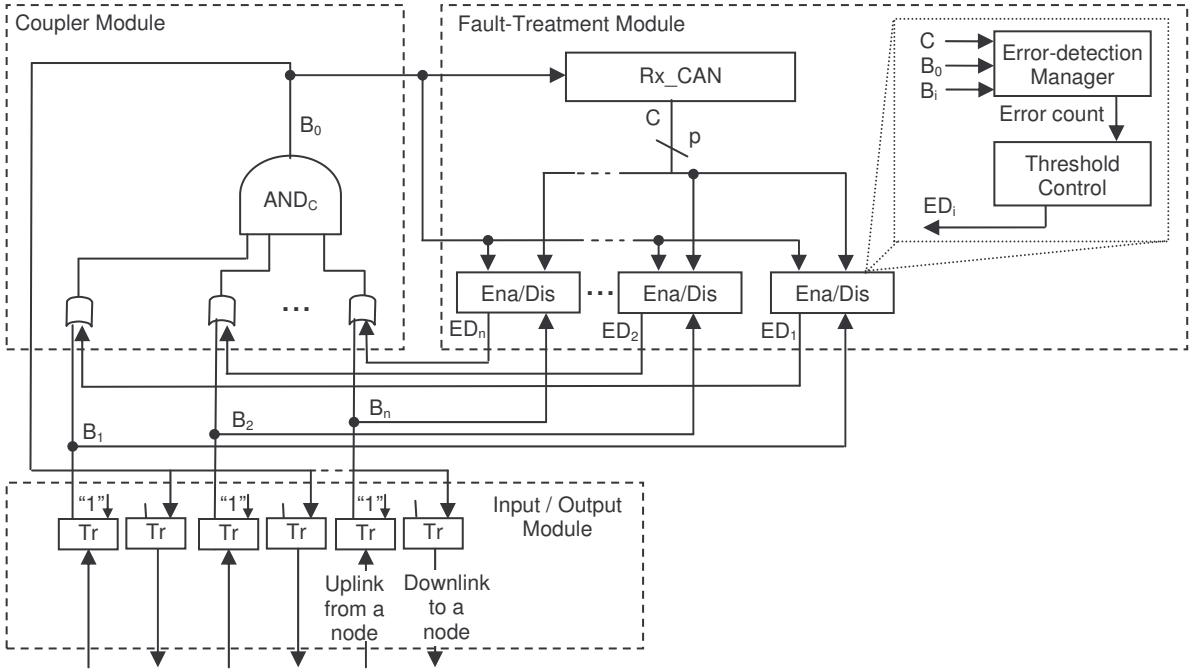

Figure 3. Internal structure of the hub of CANcentrate

The Input/Output Module contains the transceivers that translate the physical signals received from the uplinks into logic values required by the other hub modules and, in the other way, the logic value B_0 into a physical signal that is sent through the downlinks.

The Fault Treatment Module contains the Rx_CAN Module and a set of Enabling/Disabling Units that monitor each port contribution and decide on its enabling or disabling. The Rx_CAN Module observes the coupled bit-stream B_0 to synchronize with the *resultant frame* generating information on the *current state* of each hub transmission, i.e., the meaning of each outgoing bit such as the frame field every bit belongs to, whether or not a bit is a *stuff bit*, etc. Each Enabling/Disabling Unit uses the *current state* of the *resultant frame*, *C* in Figure 3, and the current role of the node attached to the port it supervises, i.e., transmitter or receiver, to estimate the next bit contribution from that port. If a mismatch is observed between the expected bit and the actual node contribution, an adequate error condition is raised and an associated error counter is incremented. These counters are also decremented after predefined periods of error-free operation for each error condition separately. When a given port has accumulated too many errors, the respective Enabling/Disabling Unit isolates the port contribution by driving a logic "1" through the Enabling/Disabling signal ($ED_{1..n}$) into an OR gate inside the Coupler Module. This will replace the actual port contribution with a "1" (recessive bit) at the respective input of the AND gate in the Coupler Module, which means disconnecting that port.

The use of separate uplinks and downlinks for each node allows separating the contribution of each node from the coupled signal, so that the Enabling/Disabling Units can monitor each node contribution separately and detect faulty transmissions. This feature allows the hub to diagnose the location of faults with more precision than the typical error counters of CAN [3]. Permanent, or sufficiently long,

faulty contributions are disabled, thus not propagated to the coupled signal. Additionally, for the sake of survivability, the Enabling/Disabling Units continue monitoring the contribution from isolated ports and perform their reintegration after an error-free predefined time interval.

Moreover, since the hub carries out the coupling within a fraction of the bit time, its operation is transparent for the nodes. This makes CANcentrate fully compliant with CAN as referred before. However, a minor adaptation is still needed when connecting an ordinary CAN node to a CANcentrate port because of the separation between uplink and downlink [12]. Using COTS transceivers, this connection requires two of them at both ends of each link as shown for the hub end in Figure 3, in the Input/Output Module.

Although CANcentrate provides CAN with error-containment features that cannot be achieved by means of bus topologies, it does not provide fault tolerance. This must be handled by the application. However, ReCANcentrate [13] can provide such increased reliability by using a replicated star topology. Although ReCANcentrate does not limit the number of hubs, for the sake of simplicity, we consider only two hubs in the remainder of this paper.

The connection strategy is similar to that of CANcentrate, with each node connected to each hub via an uplink and a downlink (Figure 4). The replication strategy is such that nodes receive the same data, bit by bit, through all the stars in parallel and this is enforced transparently by a special coupling of the hubs using two or more dedicated links called *interlinks*, each containing two independent *sublinks*, one for each direction (Figure 4). Using more than one interlink allows tolerating interlink faults. Since the traffic in both stars is mirrored by the hubs coupling, the nodes can transmit to only one of the hubs at a time, no matter which.

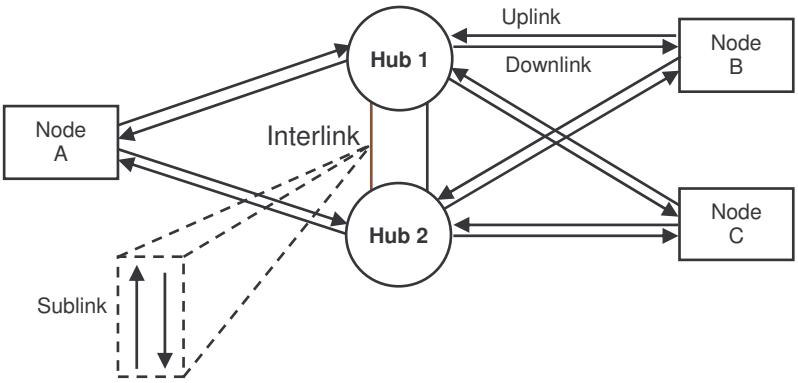

Figure 4. Connection schema of ReCANcentrate

Internally, a ReCANcentrate hub is very similar to a CANcentrate one (Figure 5). The main difference is that the Coupler Module in a ReCANcentrate hub performs the AND coupling in two stages. In a first stage each hub couples the contributions from its own nodes (i.e., the nodes that are directly connected to it), obtaining B_0, which now is called the *contribution of that hub*. The hub generates two (or more) replicas of this contribution, B_{00} and B_{01}, and sends them to the other hub via one sublink of each interlink. In a second stage each hub couples the replicas of the contribution of the other hub, i.e., B'_{00} and B'_{01}, with its own contribution, B_0, and then broadcasts the resulting signal, B_T, to the nodes that are directly attached to it.

This coupling scheme is necessary to allow each hub to monitor the other hub and isolate it in case of detecting errors. Such monitoring and isolation is carried out by specialized units, called Hub Enabling/Disabling Units, that differ from the ports Enabling/Disabling Units essentially in the thresholds that are applied to the respective error counters. In addition, when a hub fails it is also

isolated by the nodes directly connected to it, by the native error detection mechanisms of CAN controllers.

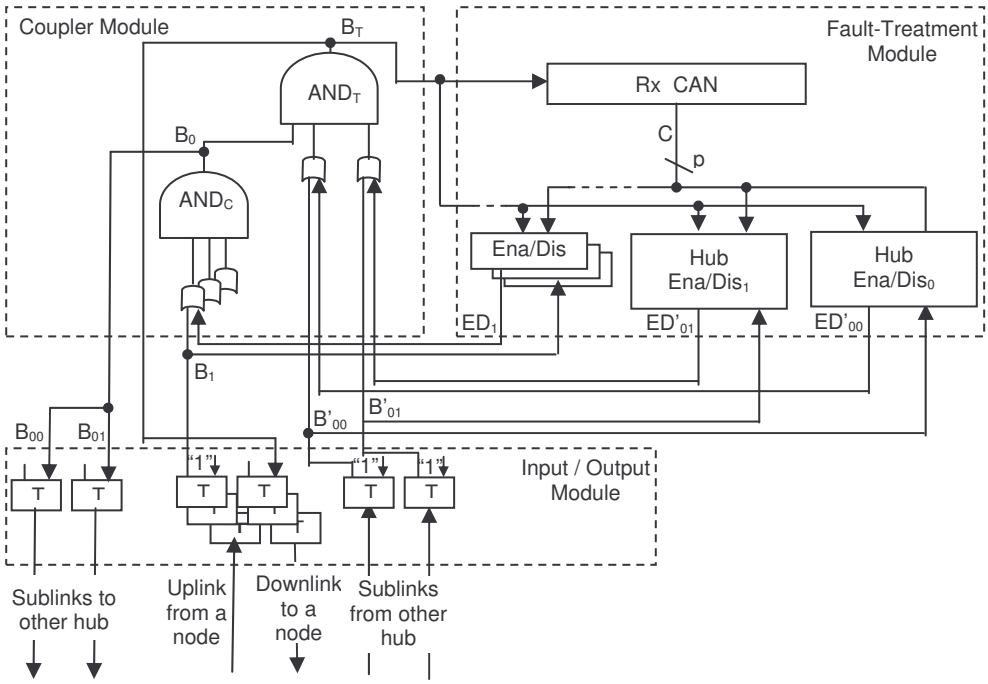

Figure 5. Internal structure of the hub of ReCANcentrate

The final coupled signal, which is broadcast to the nodes by all hubs, is unique and contains the contributions of all nodes in the system with at least one non-faulty connection to one hub, no matter which. This enforces a consistent view of the network, i.e., all connected nodes reach each other, even when some of them are connected to one hub, only. The possibility of connecting less critical nodes to only one hub is also very desirable to reduce the cabling cost. On the other hand, critical nodes must be connected to more than one hub. This can be accomplished using one independent CAN controller to connect to each hub or, as described in [4], a single CAN controller to connect to all hubs. In any case, the bit-level synchronization enforced by ReCANcentrate between the replicated stars also allows overcoming the difficult problem of replicated channel synchronization, even when using multiple CAN controllers, since the replicated stars behave like a single logic communication medium.

5 Cabling and bit rate considerations

The in-bit response property of CAN imposes an inverse relationship between the bus length and the maximum bit rate because of the time needed to propagate each bit through the network before transmitting the next one. Since this property is kept in (Re)CANcentrate, the same relationship holds. Fortunately, such compromise applies to the star diameter only, because the bit propagation progresses in parallel in all links. For CANcentrate the diameter is the sum of the two longest links, whereas in ReCANcentrate the diameter is the sum of the two longest links plus the longest interlink. However, the inclusion of the hub or hubs within the communication path between nodes introduces an extra delay that further restricts the bit rate. Therefore, the maximum bit rate allowed in a star (B) is strictly lower than in a CAN bus with a length equal to the star diameter (B'), given the delay introduced by the hub(s), t_h (1). Note that this delay must account for the time to cross the hub(s) twice, as the signals travel back and forth before they stabilize.

$$B = \frac{B'}{1 + B' t_h} < B' \quad (1)$$

Nevertheless, in a practical application two aspects must be considered. Firstly, the difference between B and B' is small. For example, in the prototype implementation described in [12], COTS CAN transceivers were used leading to t_h=310ns. For B'=250Kbps, $1/(1+B't_h)$=0.93, which has a minimal impact on the bit rate. The impact is further reduced for lower transmission rates. Moreover, using special purpose high-speed transceivers, the reduction in bit-rate becomes insignificant. But a more important aspect is that the star diameter is normally shorter than the bus length needed to connect all the nodes, which favors the star topology most likely out-weighting the limitation referred in (1).

Regarding the cost of the cabling, star topologies do not necessarily lead to longer total cabling and higher costs than bus topologies [14]. In fact, the cabling length is highly dependent on the network physical layout, e.g., if a bus has to connect nodes around a given perimeter, a hub placed near the center of such perimeter may, depending on the number of nodes, require less cabling to connect them. In any case, the benefits in terms of dependability yielded by star topologies when compared with bus topologies, should justify a higher cost when dependability is an issue.

6 Prototyping and fault injection tests

Two prototypes, one of CANcentrate and another of ReCANcentrate, were built to verify their functionality and measure their performance [12, 14]. Each hub was mainly implemented using the VHSIC Hardware Description Language (VHDL) and was synthesized in a Field Programmable Gate Array (FPGA). The physical interface of the FPGA with the media was implemented with COTS CAN transceivers and COTS connectors. Links and interlinks were built using UTP Cat 5 Ethernet cable because of containing multiple twisted-pairs.

CAN nodes were implemented using COTS components only, including a micro-controller with integrated CAN Controller plus a pair of CAN transceivers. For ReCANcentrate two pairs of transceivers were included in each node but with coupled downlinks as in [4] to allow using a single CAN controller. Despite limiting the fault tolerance properties of ReCANcentrate, this approach is simple to deploy and still allows verifying the fault treatment and fault tolerance capabilities of nodes and hubs.

The validation of the functionality was performed at the level of both the VHDL design of the hubs and the physical network [12, 14]. At the first level, the state machines that constitute the hubs were successfully checked under error-free conditions as well as under fault scenarios included in the (Re)CANcentrate fault models. At the physical network level, the network load was forced close to the maximum with an arbitration in every frame transmission. Then, faults were injected connecting and disconnecting nodes, and using a fault-injector.

Disconnecting nodes was always correctly diagnosed by the hubs as stuck-at-recessive ports. With ReCANcentrate it was also observed that nodes were able to communicate with each other as long as each of them was connected to at least one hub and the hubs were interconnected through at least one interlink. Inconsistencies in the output streams of the ReCANcentrate hubs were never observed, thanks to the hubs coupling.

The fault injector generated a square wave, corresponding to a periodic sequence of dominant and recessive bits, which was applied to links and interlinks. The frequency of this wave varied from 6.6Khz to 2.5Mhz. Using low frequencies compared to the bit rate allowed injecting alternating sequences of stuck-at-dominant and stuck-at-recessive bits. Using higher frequencies, i.e., close to the bit rate or higher, allowed injecting bit-flipping faults. All faults were correctly diagnosed and isolated by the affected hub(s). At the lower frequencies, the hub(s) were also able to reintegrate an isolated port during the recessive pulses. At 625kbit/sec, the measured link isolation latency was 73µs for stuck-at-dominant faults and ranged between 150µs and 690µs for bit-flipping faults. The

corresponding interlink isolation latencies were 216µs and ranged between 476µs and 2600µs, respectively.

Concerning performance, the maximum star diameter achieved with CANcentrate at 690kbit/sec was of 70m [12], whereas for ReCANcentrate at 625kbit/sec it was of 25m [14]. These values were independent from the number of ports the hubs were provided with.

7 Conclusion

Similarly to other communication technologies that turned from bus to star topologies, e.g., Ethernet, TTP/C and FlexRay, Controller Area Network can also benefit from such move. Star topologies with adequate mechanisms can boost the error containment capabilities of the communication system and bring substantial gains in robustness and reliability. This paper proposed two star architectures for CAN, a simplex star CANcentrate and a replicated star ReCANcentrate, that provide two different levels of dependable communication, the former aiming at increasing robustness for general distributed embedded systems and the latter aiming at safety critical applications. Both architectures are fully compatible with CAN and allow benefiting from the good properties of this technology such as the electrical robustness, the easiness of deployment, the bounded latency with prioritized medium access control and low cost. This last aspect is particularly relevant because with the proposed architectures it is possible to reach robustness and reliability levels that were only possible with more expensive technologies such as TTP/C and FlexRay.

8 Acknowledgement

This work is partially supported by DPI 2005-09001-C03-02 and FEDER funding.

9 References

[1] G. Stoeger, A. Mueller, S. Kindleysides and L. Gagea, “Improving Availability of Time-Triggered Networks: The TTA StarCoupler”, *SAE 2003 World Congress*, Detroit, USA, 2003.

[2] FlexRay™, “FlexRay Communications System Protocol Specification Version 2.1”, 2005.

[3] ISO, “ISO11898. Road vehicles - Interchange of digital information - Controller area network (CAN) for high-speed communication”, 1993.

[4] J. Rufino, P. Veríssimo, and G. Arroz, “A Columbus' Egg Idea for CAN Media Redundancy” *FTCS-29. The 29th International Symposium on Fault-Tolerant Computing, Winconsin, USA*, June 1999.

[5] J. Rushby, “A Comparison of Bus Architectures for Safety-Critical Embedded Systems”, SRI International, Menlo Park, California. Contractor Report, 2003.

[6] J. Ferreira, L. Almeida and J.A. Fonseca, “Bus Guardians for CAN: a Taxonomy and a Comparative Study” Proc. of WDAS 2005, Workshop on Dependable Automation Systems, Salvador, Brazil, October 2005.

[7] CiA, “CAN physical layer”, CAN in Automation (CiA), Am Weichselgarten 26, Tech. Rep. [Online]. Available: headquarters@can-cia.de

[8] M. Rucks, “Optical layer for CAN”, *1st International CAN Conference*, November 1994.

[9] IXXAT, “Innovative products for industrial and automotive communication systems”, 2005. [Online]. Available: http://www.ixxat.de/index.php

[10] G. Cena, L. Durante, and A. Valenzano, “A new CAN-like field network based on a star topology”, Computer Standards & Interfaces, vol. 23, issue 3 , July 2001

[11] H. Saha, “Active High-Speed CAN HUB”, *Proceedings of the 11th international CAN Conference (iCC 2006)*, Stockholm, Sweden, 2006.

[12] M. Barranco, J. Proenza, G. Rodríguez-Navas and L. Almeida, "An active star topology for improving fault confinement in CAN networks", *IEEE Transactions on Industrial Informatics, vol. 2, num. 2*, USA, May 2006.

[13] M. Barranco, L. Almeida and J. Proenza, "ReCANcentrate: A replicated star topology for CAN networks", *Proceedings of the 10th IEEE International Conference on Emerging Technologies and Factory Automation (ETFA 2005)*, Catania, Italy, 2005.

[14] M. Barranco, L. Almeida and J. Proenza, "Experimental assessment of ReCANcentrate, a replicated star topology for CAN", *SAE 2006 World Congress*, Detroit, USA, 2006.