TTCAN over mixed CAN/Switched Ethernet architecture

Jean-Luc Scharbarg, Marc Boyer, Jérôme Ermont, Christian Fraboul
IRIT - ENSEEIHT
2, rue Camichel
31000 Toulouse - France
Jean-Luc.Scharbarg@enseeiht.fr

Abstract

Embedded systems have specific real-time requirements that led to the development of dedicated communication protocols. Such systems often face increasing communication needs and the integration of switched Ethernet architecture. But moving from existing dedicated fieldbusses architectures to new Ethernet based architectures is not always easily feasible, due to industrial constraints.

In this paper, we propose a solution for integrating existing data busses (such as CAN, which is an important standard in automotive context) on a global architecture that respects increasing bandwidth requirements. We consider a time triggered CAN application and propose CAN/Ethernet bridging strategies that respect the time-triggered real time behaviour of CAN End System when communicating through a switched Ethernet network that can also be shared by (non CAN) applications.

1. Introduction

Fieldbusses , e.g., CAN [12], WorldFIP [17], Profibus [17] have been developed in the context of real-time applications (distributed computer control systems) that have specific communication requirements such as:

- bounded end-to-end transmission delays in order to guarantee respects of deadlines,
- the bounded and small jitter for periodic traffic.

However, the amount of information that are nowadays exchanged in such systems have been increasing steadily and is now reaching the limits of traditional fieldbusses, especially in terms of bandwidth [8].

Switching from dedicated fieldbusses to Ethernet is a classical trend in embedded systems due to the wide acceptance of the Ethernet standard and its evolution toward a more predictable switched architecture.

However, successful experience with introduction of a switched Ethernet in avionic systems (AFDX, [2, 11]) is mainly due to the preservation of the applications communication model (periodic schemes) and the respect of the expected real time properties (bounded delay).

The goal of the study presented in this paper is to build an heterogeneous architecture obtained by interconnecting existing CAN data busses on a switched Ethernet. More precisely, we aim at obtaining a time-triggered behaviour (TTCAN like [13]) on the whole system.

We have built a twofold study: on the one hand, we have developed a prototype with some Linux-PCs with CAN or Ethernet cards inside, on the other hand, we realised a more theoretical study. First results of the theoretical study are shortly presented in this paper.

Section 2 presents briefly TTCAN and switched Ethernet technologies. Section 3 proposes a solution to obtain a time-triggered behaviour on a CAN / switched Ethernet architecture. Section 4 concludes the paper and presents points we are presently studying and some ideas for future work.

2. Communication technologies

In this section, we present the two communication technologies that we intend to use, i.e., the Time-Triggered Controller Area Network (TTCAN) and Switched Ethernet.

2.1. TTCAN

Time-Triggered CAN (TTCAN) [10, 13] is an enhancement of the Controller Area Network (CAN). CAN [12] is a serial communication protocol suited for networking sensors, actuators and other nodes in real-time systems. The CAN specification defines several versions of the protocol for the physical and the data link layer. In this paper, we shortly present CAN 2.0 A. Several application layer protocols have been proposed (CANOpen, CAN Kingdom, . . .).

The CAN addressing system is based on message identifiers : a frame does not have a destination nor a source address. Frames are broadcasted on the bus. Stations get the frames they are interested in by a filtering process of the identifiers.

The frame format is depicted in Figure 1. The details of each field will not be presented. The relevant fields for the remainder of the paper are the following :

0-7803-9402-X/05/$20.00 © 2005 IEEE

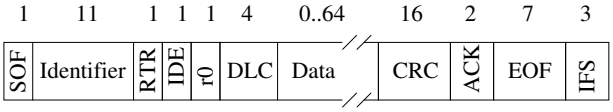


Figure 1. CAN frame (sizes in bits)

- the identifier field, which as mentioned earlier identifies the data contained in the frame,
- the DLC field which gives the length (in bytes) of the data field,
- the data field which is the payload of the frame.

Bit-stuffing is used to avoid the transmission of long sequences of bits with identical value [15]. As soon as 5 bits of identical value are transmitted, a bit of opposite value is automatically inserted. This mechanism is valid for the whole frame, except IFS, EOF, ACK and the last bit of CRC.

The medium access method (MAC) is CSMA/CR (Carrier Sense Multiple Access / Collision Resolution) : the starting of frame transmissions on the bus are synchronous. When two or more stations start a transmission simultaneously, the one with the highest priority identifier (lowest value) wins and the others stop their transmission. This is implemented by a collision detection on a bit by bit basis. When a station transmits 1 (recessive bit) and detects 0 (dominant bit), it knows that a frame with a higher priority is being transmitted and, consequently, it immediately stops transmission. This mechanism guaranties strict priority order on identifiers, provided identifiers are unique. It implies limitations of the bandwidth and the maximal length of the bus (e.g., 1 Mbs for 40 meters).

Some drawbacks of the CAN native MAC have been identified. First, it is event-triggered : when a station has a frame to transmit, it tries to. It will succeed as soon as no frame with a higher priority is being transmitted. This mechanism can induce large jitter on periodic frames. Second, identifiers are associated with frames statically. This imposes a scheduling algorithm using static priorities, e.g., rate monotonic [14] when periodic traffic is considered. It is well known that higher utilisation of the medium is obtained with a scheduling algorithm using dynamic priorities, e.g., Earliest Deadline First [14].

Solutions have been proposed to solve those drawbacks. Most of them add a protocol over CAN native MAC (e.g., [7, 16]).

Time triggered CAN (TTCAN) [10, 13] is a well-known solution. It imposes a static scheduling on CAN. This scheduling is memorised in a table, the matrix cycle, which is known by all the stations. This scheduling comprises in particular exclusive arbitration and free windows. Each exclusive window is dedicated to exactly one frame identifier while an arbitration window is shared. Free windows allow some evolution of the application. The stations are resynchronised with a trigger message broadcasted periodically by the master station. Figure 2 gives an example of a matrix cycle. Each line of the table is called a micro-cycle and has a duration D_{uc}. The number of micro-cycles in the table is a power of 2. All the windows in a given column of the table have the same duration. The reference message can be easily recognised by its identifier. With TTCAN's level 1, the reference message only holds some control information of one byte. In extension level 2, the reference message holds additional control information (e.g., the global time information of the master) and covers four bytes.

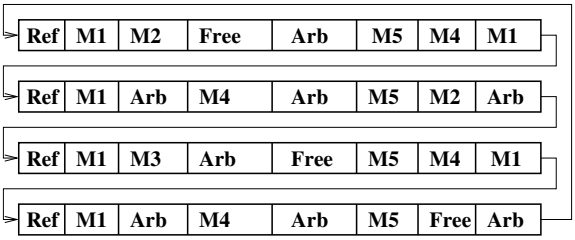


Figure 2. Example of a matrix cycle

2.2. Full Duplex Switched Ethernet

Full Duplex Switched Ethernet is an enhancement of Ethernet. The Ethernet link layer [9] is designed for computer local networks where high bandwidth and low cost hardware is more important than guaranteed deadlines and/or jitter.

The Ethernet addressing system is based on MAC addresses: each Ethernet entity has a unique MAC address. In each frame, the destination (unicast, broadcast or multicast) and source addresses are inserted. Frames are broadcasted on the physical layer. Entities get the frames there are interested in by a filtering process.

The Ethernet native medium access method is CSMA/CD (Carrier Sense Multiple Access / Collision Detection): the collision resolution mechanism is non deterministic and leads to unbounded transmission delays.

Full Duplex Switched Ethernet is a way to bypass the medium access strategy of Ethernet: each station is directly connected to an Ethernet switch with a full duplex link. Then, the medium is always free. Consequently guaranteed performances are strongly connected to policies of the switch. Several approaches have been developed. One generic approach is the Network Calculus ([5, 6]), that have been successfully applied in the AFDX network system for Airbus embedded networks [2, 11]. Another is the Response Time Analysis, [14].

3. Time-triggered principles over CAN / full duplex switched Ethernet

Our goal is to interconnect several CAN busses with a full duplex switched Ethernet network and to obtain a time triggered behaviour on the whole network. Such an architecture is depicted in figure 3. It includes four CAN busses and an Ethernet switch. There is a bridge station

between each CAN bus and the switch. The switch has four receive ports and four queued transmit ports. When a frame arrives at the switch, the control logic determines the transmit port and tries to transmit the frame immediately. If the port is busy because another frame is already being sent, the frame is stored in the first-in first-out transmit port queue. The memory to store pending frames is obtained from a shared memory pool. If no more memory is available, the received frame is dropped.

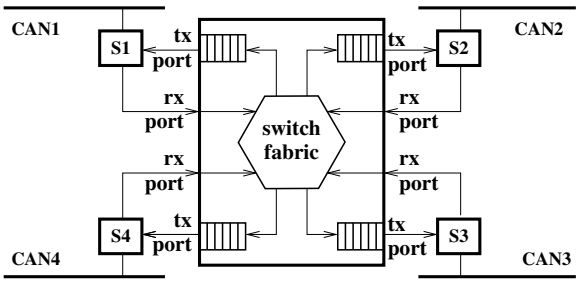


Figure 3. Network architecture

The solution we propose is based on the following principle. Each bridge station is the TTCAN master of its CAN bus. A CAN message can be either local (the producer and all the consumers of the message belong to the same CAN bus) or global (the producer and the consumers of the message are distributed on two or more CAN busses). The CAN bus where the producer of a global message lies is called its home bus, while the busses where the message has consumers are called its distant busses. A global message is not transferred on a CAN bus which is neither its home bus nor one of its distant busses. Global messages have to be transferred through the Ethernet switch. In order to do that, they are encapsulated in Ethernet frames and broadcasted via the switch. The encapsulation consists in putting the Identifier, DLC and Data fields of CAN frames in the Data field of the Ethernet frame (the other fields of CAN frames can be easily reconstructed). This means that a CAN frame occupies at most 10 bytes of the Data field of an Ethernet frame.

Different strategies can be used to implement this encapsulation:

- an event-triggered strategy: as soon as a global frame is ready on the bridge of the producer, it is encapsulated in an Ethernet frame and broadcasted via the switch,
- a time-triggered strategy: at predefined instants, all global frames ready on a bridge are encapsulated in a single Ethernet frame and broadcasted via the switch.

As we aim at obtaining a time-triggered behaviour on the whole network, we focus on the second strategy. The following problems have to be solved:

- the choice of transmission instants,
- the synchronisation between the different CAN busses, when initialising the system and to compensate clock drifts,
- the coordination of the different TTCAN tables.

An example solution to those problems is depicted in figure 4 with the network architecture of figure 3.

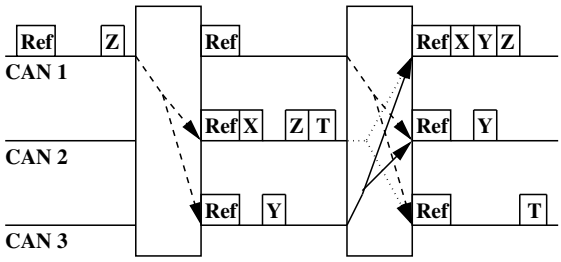


Figure 4. Time-triggered over switched Ethernet

Global messages imply three CAN busses (all the messages of CAN bus 4 are local and bus 4 will be ignored) : X is produced on CAN bus 2 and consumed on CAN bus 1, Y is produced on CAN bus 3 and consumed on CAN busses 1 and 2, Z is produced on CAN bus 1 and consumed on CAN bus 2, T is produced on CAN bus 2 and consumed on CAN bus 3. The period of messages X, Y, Z and T is 20 ms. TTCAN tables are given on figure 5. Windows where messages are produced are underlined. The duration of the micro-cycle D_{uc} is 10 ms for the three tables. Local messages are not considered in this example.

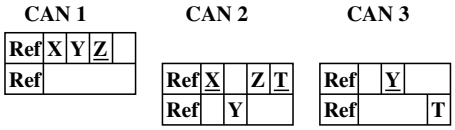


Figure 5. TTCAN tables

A master is defined among bridges stations - here, the bridge of CAN bus 1 is chosen. Its role is to synchronise the CAN busses at initialisation time. Following synchronisations do not require a master. At the beginning of the application, the first TTCAN micro-cycle executes on CAN bus 1. At the end of this cycle, the bridge of CAN bus 1 broadcasts to bridges of CAN busses 2 and 3 an Ethernet frame encapsulating global CAN frame Z. This Ethernet frame synchronises the three CAN busses. Then, the second TTCAN micro-cycle executes on CAN bus 1, while the first one executes on CAN busses 2 and 3. At the end of the TTCAN micro-cycle, each bridge broadcasts an Ethernet frame encapsulating ready global CAN frames (X and T for CAN bus 2, Y for CAN bus 3). Those Ethernet frames synchronise the three CAN busses, which then execute the next TTCAN micro-cycle.

The main characteristics of this solution are:

1. the transmission instants correspond to the ends of micro-cycles,

2. the synchronisation of different CAN busses is done at the end of each micro-cycle, via Ethernet frames exchanges,

3. the TTCAN tables are built so that, when a global frame is sent on its source bus during a micro-cycle, it is sent on its destination bus during the next micro-cycle (see for example frame Y in figure 4),

4. a global CAN frame stands in the same column in all TTCAN tables (e.g., frame Y is always in column 2 on figure 4).

The time triggered behaviour is obtained on each individual CAN bus by the construction of the TTCAN table.

As mentioned earlier, a global CAN frame has consumers in different CAN busses including the one where it is produced. Characteristics 3 and 4 above imply there is a delay of one micro-cycle between the transmission of the message on its home bus and the transmission on its distant busses. This delay has to be taken in account when designing the application. It includes the synchronisation delay which depends on the transmission delays on Ethernet links, the delay induced by the switch and the delays induced by the drift between bridges clocks.

4. Conclusion and future works

In this paper, we mainly focused on two types of communication technologies :

- the Controller Area Network (CAN) and its time-triggered version (TTCAN), which is a good example of deterministic real-time communication system,

- Switched Ethernet, which is a popular non real-time communication system.

The aim of the paper was to study the use of switched Ethernet in conjunction with CAN for communications in a real-time system. More precisely, the challenge was to obtain a time-triggered behaviour on a mixed CAN / switched Ethernet architecture. We show a solution on an architecture that includes several CAN busses interconnected via bridges by an Ethernet switch. TTCAN is used on each CAN bus and exchanges between bridges take place at the end of each micro-cycles.

The solution we propose has to be validated. We are presently working on the modelling of our system, on one hand using timed automata (using UPPAAL), on the other hand using Petri nets (using TINA [4]). The goal is to verify properties such as: is the delay of global CAN frames always respected ? Studies around the verification of protocols with such tools have already been conducted (e.g., [3]).

We are also studying the behaviour of our solution when the Ethernet network is shared with non CAN applications. In this context, the respect of the real time behaviour of CAN end systems implies the use of a more sophisticated Ethernet switch, in order to be able to differentiate traffics.

Among other points that should be studied, there is the use of other time-triggered strategies on CAN, such as FTTCAN [1] and an architecture with a more global Ethernet including several switches.

References

[1] L. Almeida, P. Pedreiras, and J. A. G. Fonseca. The ftt-can protocol : why and how. *IEEE transactions on industrial electronics*, 49(6), dec 2002.

[2] ARINC Specification 664: Aircraft Data Network, Parts 1,2,3,4,5,8. Technical report, Aeronotical Radio Inc., 2002-2005.

[3] B. Bérard, P. Bouyer, and A. Petit. Analysing the PGM protocol with Uppaal. *International Journal of Production Research*, 42(14):2773–2791, July 2004.

[4] B. Berthomieu, P. Ribet, and F. Vernadat. The tool tina – construction of abstract state spaces for petri nets and time petri nets. *International Journal of Production Research*, 42(14), July 2004.

[5] R. Cruz. A calculus for network delay, part I. *IEEE Transactions on Information Theory*, 37(1):114–131, January 1991.

[6] R. Cruz. A calculus for network delay, part II. *IEEE Transactions on Information Theory*, 37(1):132–141, January 1991.

[7] M. Di Natale. Scheduling the can bus with earliest deadline techniques. In *Proceedings of the IEEE Real-Time Systems Symposium*, 2000.

[8] D. Dietrich and T. Sauter. Evolution potentials for fieldbus systems. In *IEEE Workshop on Factory Communication Systems*, Porto, September 2000.

[9] CSMA/CD access method. IEEE Standard 802.3, IEEE, 2002.

[10] T. Führer, B. Müller, W. Dieterle, F. Hartwich, R. Hugel, and M. Walther. Time triggered communication on can. In *International CAN Conference*, 2000.

[11] J. Grieu, F. Frances, and C. Fraboul. Preuve de déterminisme d'un réseau embarqué avionique. In *Actes du 10ème Colloque Francophone sur l'Ingenierie des Protocoles*, Paris, 7-10 Octobre 2003.

[12] ISO. *ISO International Standard 11898 - Road vehicles - Interchange of digital information - Controller Area Network for high-speed communication*, nov 1993.

[13] ISO. *ISO International Standard 11898-4 - Road vehicles - Controller Area Network - Part 4 : Time-Triggered Communication*, dec 2000.

[14] C. Liu and L. J.W. Scheduling algorithms for multiprogramming in hard real-time environment. *Journal of ACM*, 20(1):46–61, 1973.

[15] T. Nolte, H. Hansson, C. Norstrom, and S. Punnekat. Using bit-stuffing distributions in CAN analysis. In *IEEE/IEE Real-Time Embedded Systems Workshop (RTES)*, 2001.

[16] P. Pedreiras and L. Almeida. Edf message scheduling on controller area network. *Computing and control engineering journal*, pages 163–170, aug 2002.

[17] J.-P. Thomesse. Fieldbusses and interoperability. *Control Engineering Practice*, 7:81–94, 1999.