Enhancing the efficiency of Controller Area Networks

G. Cena (†) A. Valenzano (‡)

Fieldbus networks should be able to support several kinds of data exchanges, characterised by very different requirements. The most popular solutions available today on the market are designed bearing in mind some sets of specific needs and usually are not always satisfactory for every kind of communications which can be found in an industrial environment.
In this paper a modification of the basic Controller Area Network (CAN) medium access technique is described which increases significantly the communication efficiency for the periodic exchanges of process data and for the messages devoted to high level functions, without affecting the very good responsiveness and flexibility of the conventional CAN protocol.

1 Introduction

Fieldbus networks should support several kinds of traffic, as a consequence of today's more and more sophisticated control applications. In an automated factory environment, in particular, there are basically two kinds of communication needs: process data, which directly affect the controlled system and messages, devoted to high level functions and parameterisation. Process data are small-sized (8 to 32 bit-long words are commonly adopted) and characterised by real-time requirements, while messages can be arbitrarily long and the timings involved in their transmission usually are not a critical issue. Process data exchanges can be further classified as *predictable* and *unpredictable*. Predictable data exchanges (also known as cyclic, periodic or synchronous) take place at well defined times, which are known *a priori* by the control application, while it is not possible to know in advance the exchange times for unpredictable data (also known as sporadic, asynchronous, acyclic or aperiodic).
On the one hand, predictable data are usually involved in those systems in which a controller device (PLC, CNC, etc.) inquiries the different I/O devices for the measured input data and status information and sends them back the related commands and set-points according to a fixed schedule. In this case, it is very important that cyclic data be exchanged at a very precise rate and that the jitters be kept low, in order for the physical system to be accurately controlled.
On the other hand, unpredictable data are important in event-driven systems, where sensor devices have to send the measured values to the intended destination(s) as soon as they are sampled from the physical system. Moreover, sporadic transmissions are useful for implementing urgent interactions such as, for example, alarm notifications or trigger signals. In this case, the communication system should be able to order the concurrent transmission requests according to some precedence criteria, so as to grant a suitable schedule for the related exchanges.
High level functions usually have not particular timing constraints and the related messages can be sent at a lower priority than the real-time data. However, they may require the transmission of a large amount of information. Hence, it is required that the available bandwidth of the communication system be shared out efficiently and fairly among the different nodes.
Each fieldbus network has basically to face two main physical limitations: first, the available bandwidth is a limited resource and, second, at any time only one node can be enabled to transmit a frame on the

† Dipartimento di Automatica e Informatica
‡ Centro di Studi per l'Elaborazione Numerale dei Segnali
Politecnico di Torino
Corso Duca degli Abruzzi, 24
10129 Torino - Italy
Phone: +39 011 564 7078 / 7061
Fax: +39 011 564 7099
E-mail: {cena,valenzano}@polito.it
Homepage: http://www.polito.it/~cena
http://www.polito.it/~valenzano

shared communication medium. Bearing in mind these constraints, an ideal communication system for industrial environments should behave as follows:

- the transmission of cyclic process data (which requires a percentage of the system bandwidth known in advance) have to be carried out with the highest precedence at the required rate, and the frequency jitters must be kept as low as possible;
- the part of network bandwidth which is not assigned to cyclic exchanges is used for non-critical real-time sporadic process data; as long as an overload condition does not occur on the network, bounded transmission delays must be ensured: to this extent, priority tags can be assigned to the different data in order to set the correct precedence among concurrent exchanges;
- the amount of network bandwidth not used for real-time data exchanges (both cyclic and sporadic) is shared out fairly and efficiently among the different nodes needing to transmit messages for supporting high level functions, so as to maximise the throughput for each node; in this case temporary network overloads are tolerated and must be managed correctly by the network without affecting the real-time exchanges;
- timings of very urgent sporadic data (critical alarms, whose occurrences are unpredictable) must be strictly respected; to this extent, their transmissions can pre-empt any non-critical transmission (either process data or messages) which is currently taking place on the network.

In the following, a new access technique and new transmission services are proposed for the Controller Area Network protocol whose behaviour is close to the ideal solution mentioned above. The paper is structured as follows: section 2 analyses the properties of several existing medium access control (MAC) mechanisms, while section 3 introduces the relevant characteristics of the new protocol. Section 4, finally, contains considerations on how the new technique can be applied.

2 Medium access techniques characteristics

When the timings of all the data exchanges are known in advance, such as in the case of systems based only on cyclic exchanges, a time division multiple access (TDMA) technique is surely the most efficient solution. TDMA combines all the data produced or consumed by different nodes in a single (summation) frame and requires that the protocol control information be added only once per frame in order to achieve bit and frame synchronisation and appropriate error controls.

TDMA ensures the highest efficiency among all the different MAC techniques. In Interbus [1] (which is based on a MAC technique whose behaviour is similar to TDMA), for example, the efficiency for a sample controlled system can be as high as 0.6 (i.e. 60%). On the contrary, TDMA is very unsuited for sporadic data exchanges and high level communications, in that it requires the system bandwidth to be allocated in advance to the different nodes thus leading to poor flexibility. In this case, the bandwidth which is not used by a node cannot be reallocated to other stations, so that it is effectively wasted.

When sporadic process data need to be exchanged, a carrier sense multiple access (CSMA) technique with deterministic collision resolution, such as the one adopted in the controller area network (CAN) protocol [2][3], appears to be one of the most suited solutions. In this case, unlike pure CSMA, each different piece of information is assigned a priority tag (identifier). When a node needs to send a message, it starts the transmission as soon as the medium is free and, if a collision takes place, the contention is solved by means of an arbitration phase based on the value of the identifiers. The access technique of CAN, together with its limited payload size (up to 8 data bytes per frame are allowed), ensures very short response times and enables a scheduling policy which is truly distributed and dynamic.

Exchanging small pieces of information in a message-based network like CAN, however, leads to very poor efficiency. In the case of remotely requested 8 bit cyclic process data, the efficiency drops down to about 0.08, which means that the neat bandwidth effectively available is less than

Medium Access Technique	Kind of traffic		
	Predictable	Unpredictable	Messages
TDMA (Interbus-S)	High efficiency (combined message, one master / many slaves)	Medium responsiveness (can only be implemented with periodic exchanges)	Medium efficiency / medium flexibility (high fragmentation, static allocation)
CSMA (CAN)	Medium efficiency (message-based, each node is a master)	High responsiveness (truly dynamic and fully distributed)	Medium efficiency / high flexibility (high fragmentation, dynamic allocation)
Token Passing (PROFIBUS)	Medium efficiency (message-based, more than one master / many slaves)	Medium responsiveness (can only be initiated by masters at token reception)	High efficiency / high flexibility (low fragmentation, dynamic allocation)
CAN+	High efficiency (combined message, many masters / many slaves)	High responsiveness (truly dynamic and fully distributed, pre-emptive)	High efficiency / high flexibility (low fragmentation, fair and dynamic allocation)

Tab. 1: Characteristics and performances of some popular fieldbus networks.

one tenth of the network bit rate. In the same way, the small payload of CAN frames leads to a reduced efficiency for large messages, because of the relevant effect of the fragmentation [4].

Finally, when message transmissions are considered, a token-based access technique such as that adopted in Profibus [5] (where each single frame can include up to 246 user data bytes) seems to offer more satisfactory performances with respect to Interbus and CAN, at least from the point of view of the bandwidth share-out (in terms of both flexibility and efficiency). In this case, however, a reduced responsiveness and increased jitters are obtained for real-time process data.

As shown in Tab. 1, each kind of medium access technique provides optimal performances for a particular type of data exchanges, while it is usually less suited to other kinds of communications required in the factory automation systems. The basic idea of this paper is to find a satisfactory trade-off between the CSMA and TDMA access techniques for getting the most out of a fieldbus network. In this paper we propose a modification to the CAN protocol [2] [3] (called CAN+) which introduces TDMA-like data exchanges and also improves the transmission efficiency of low priority messages without reducing the responsiveness and the flexibility of conventional CAN. As will be shown, the resulting protocol combines the best features of a number of fieldbus protocols and ensures a very high efficiency and responsiveness in all the operating conditions.

3 CAN+ basics

The original CAN protocol exhibits very good performance for exchanging sporadic real-time data, while no special service is provided for cyclic data exchanges or for the transmission of large messages. This section describes the additional services of CAN+ which enhance the performance of CAN for the data exchanges mentioned above.

CAN+ introduces two new services, which are used respectively to collect the process input data and to distribute the process output data to a number of slave devices at the same time, thus providing a mechanism similar to the summation frame of Interbus [1]. Since the summation frame is usually larger than the conventional frames, the overall responsiveness is somehow reduced. To avoid excessive transmission delays, a pre-emption mechanism has been provided for extremely urgent notifications. Moreover, a service explicitly conceived for high efficiency message transmissions has also been introduced.

The new services make use of a MAC technique and frame formats very similar to conventional CAN, hence a certain degree of backward compatibility is ensured, at least from the applications' point of view.

3.1 Cyclic exchanges

Since the timings of cyclic data exchanges are known in advance, a technique similar to TDMA is used in CAN+ which relies on a master-slave interaction model to increase the communication efficiency. In particular, a solution similar to the summation frame of Interbus has been adopted where all the

output process data are transmitted in a single (large) OUT frame, while a special IN frame is used to collect all the input process data. The resulting efficiency increases because in general a smaller number of control bits are used by the protocol.

Unlike the single summation frame of Interbus, two kind of data-link services are provided, that is L_IN and L_OUT, which are used to read/write the input/output data from/to the slave devices, respectively. This is mainly due to the simple bus structure of CAN networks, whose nodes have not separate incoming and outgoing links as in Interbus. In the case of cyclic exchanges, the initiator of the service (either L_OUT or L_IN) is the cycle master, while the other devices involved are said cycle slaves.

Unlike Interbus, CAN+ is able to support more than one master on the same network, and each master can support different kinds of cycles (each one characterised by a unique cycle identifier), to allow each group of slave devices to be sampled at a different rate (as occurs, for example, in the FIP [6] protocol). This provides a further improvement of the overall efficiency. At a first glance, the use of long frames could reduce the network responsiveness unacceptably. In practice, cyclic process data usually have a higher precedence than the other kinds of information exchanged on the network, and the strict respect of their timings is always considered a fundamental requirement. If urgent sporadic data need to be sent with very tight timing requirements a pre-emption mechanism is provided in CAN+ to stop the IN and OUT frame transmissions, as described in detail in section 3.3.

Fig. 1 shows the format of the OUT frame. In this case, the cycle master puts the output process data for the different devices (one output slot for each slave device) directly after the frame length field (which is 6 bits long). Optionally (as specified in the configuration phase) a 4-bits ACK field can be included in each output slot immediately after the data field. As in CAN, the OS ACK field is made up of an ACK slot, which is overwritten with a dominant value by the intended receiver of the process datum to confirm the correct reception, followed by a recessive ACK delimiter bit. The ACK field begins with a synchronisation sequence, consisting of a dominant bit followed by a recessive bit, which provides a synchronisation edge in the bit stream and is used to transfer temporarily the right to access the shared communication medium to the slave device. The last output slot is followed by a dummy field which pads the overall data field to the nearest byte boundary.

Fig. 2 depicts the format of the IN frame. In this case, each piece of input data is preceded by a pair of bits which respectively specify whether the responding device is operating (slot present bit) and if the returned data have been refreshed since the last read operation (slot valid bit, used for checking the temporal coherency [6]). If a slave device does not respond (as indicated by a recessive value in the “present” bit), the cycle master must fill the gap on the bus with a dummy pattern so that the overall IN frame does not violate the bit stuffing rules.

Each input slot is followed by a synchronisation sequence for transferring the ownership of the bus to the next slave. When the last input slot has been

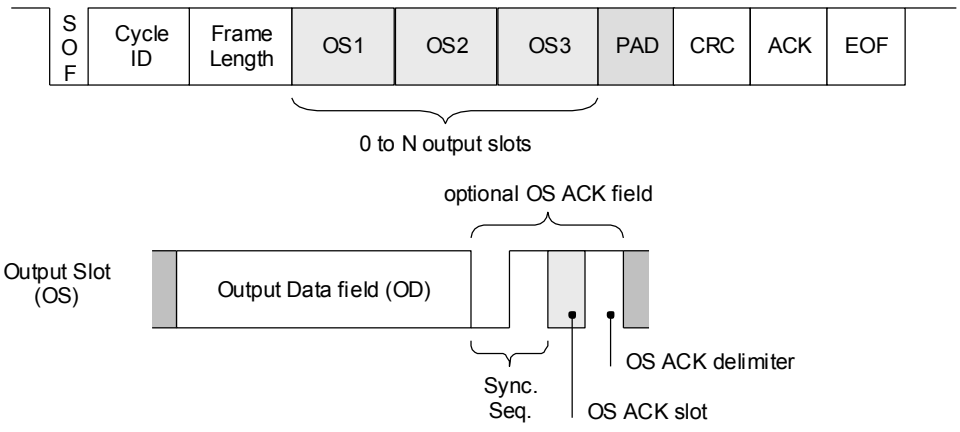

Fig. 1: Format of the OUT frame.

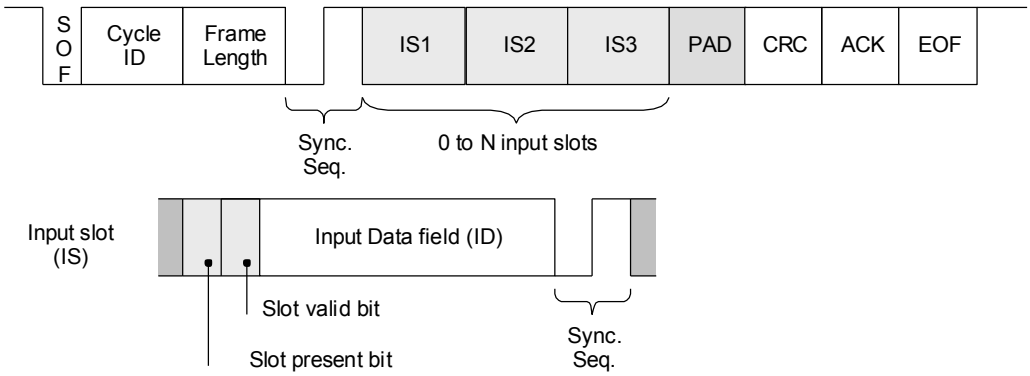

Fig. 2: Format of the IN frame.

exchanged, the cycle master sends a pad field, followed by the CRC field which is checked by all the nodes in the network. Each slave device computes the CRC and invalidates the whole frame in the case a mismatch is detected.

Input data are updated on the slave side by means of the L_UPDATE service, which writes the new value for the device into a local buffer. The data read from the different devices are then transferred effectively to the cycle master by means of the next IN frame circulated on the network.

Fig. 3 (which is related to the L_IN service) shows that the communication efficiency depends directly on the size of the combined data field, that is on the number of slave devices and the size of the process data. Since we do not wish to affect the responsiveness of the system heavily, the size of the data field in the IN and the OUT frames is limited to 63 bytes. As shown in Fig. 3, any further increase of this size does not lead to any real improvement of the network efficiency but the transmission delays are worsened accordingly. When cyclic process data exceed the 63 bytes limit, they can be split in several (different) cycles.

To make a comparison, in a conventional CAN network the efficiency for a cyclic exchange of 8 bit-sized asynchronous process data is only about 0.14, irrespective of the number of slave devices in the system, and that value drops down to 0.08 when slave devices are polled by the master by means of RTR frames. In a similar system where 10 different slave devices are grouped together by means of the L_IN service, the efficiency of CAN+ rises up to 0.46.

Cyclic services are not symmetric and hence two different roles are involved, that is master devices (controllers) and slave devices (sensors and actuators). The exact position of a particular slot in the IN and OUT frames is known only to the associated slave device and to the cycle master. This means that the combined frame is seen by the other nodes in the network as any conventional frame, since the mechanism ensures that the bit stuffing rules are always respected, even when some slave is in a non-operational state.

A suited configuration phase should be provided (before the normal operations are started) where the cycle master notifies each device the position of its input and output data slots in the combined frame. Such a configuration phase can be either static or dynamic; in the latter case, the conventional services of CAN can be used for the configuration operations.

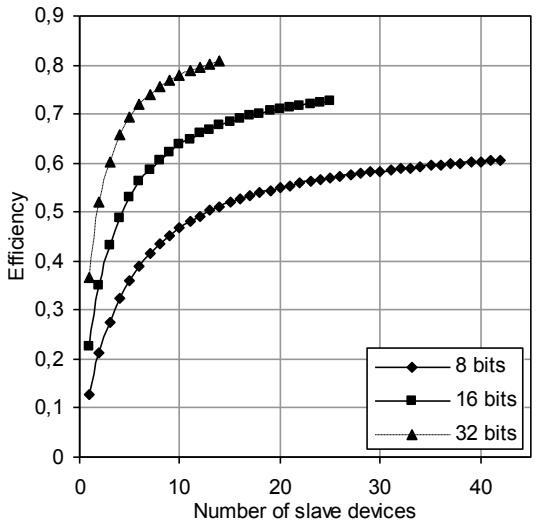

Fig. 3: Transmission efficiency of the cyclic exchanges vs. number of slave devices and size of data.

3.2 Block transfer

To ensure a satisfactory responsiveness for high priority frames, CAN frames have a data field which is very small with respect to other popular fieldbus protocols. This implies that long messages have to be split into a sequence of smaller chunks (which are referred to as fragments) according to some fragmentation protocol, which is usually placed in the application layer [7, 8]. Fragmentation protocols perform all the operations needed to split the message, send each fragment into a separate frame and reassemble all the fragments at the destination node to rebuild the original message.

Whichever the kind of the fragmentation protocol used, the resulting efficiency for exchanging messages is quite low. In the case of CAL/CANopen, for example, each fragment causes a pair of frames to be exchanged. This leads to an efficiency which is theoretically not better than 0.25. The major drawback in performing fragmentation at the application level, however, is that it has to be carried out in software and hence it consumes CPU time. Instead, a hardware implementation at the data-link level (communication controller), can lead to noticeably higher performances.

An improvement of the transmission efficiency for the low priority messages (without penalties for the network responsiveness) can not be obtained by simply increasing the payload of the frame. A possible solution is to split the message into a sequence of 8 byte fragments, which are then sent sequentially by the transmitting node. Each fragment is followed by a stop field, which consists of a stop bit preceded by a synchronisation sequence and followed by a stop delimiter. By overwriting the stop bit with a dominant value, each node in the network which has to transmit a higher priority frame can temporarily (and gracefully) stop the current message transmission, without discarding those fragments that have already been sent.

A special BLOCK frame has been introduced in CAN+ to support the transmission technique described above whose data field is organised as a sequence of adjacent message fragments, as shown in Fig. 4. When the transmission of the block frame is stopped, the node which is currently transmitting the message adds (immediately after the stop field) the CRC, ACK and EOF fields to the bits already transmitted, as in the conventional CAN frames. Then, the transmission of the message will be resumed from the point it was abandoned (whole fragment boundary) when the bus becomes free again. In this way, a long message is sent as a sequence of block frames, each one consisting of a number of fragments. If the message transmission is never interrupted, only one block frame is effectively transmitted.

If an error is detected, the error management mechanism of CAN stops the current frame transmission and ensures that both the transmitter and all the receivers are notified of the error. In this case all the fragments in the current block frame have to be re-transmitted.

In CAN+ two kinds of fragment are used, that is intermediate (IF) and final fragments (FF). All the intermediate fragments contain exactly 8 data bytes, so that it is unnecessary to specify their length explicitly. The final fragment, instead, can contain 0 to 7 user data bytes and hence includes a 3-bit fragment length (FL) field, while the stop field is not present. In block frames, the conventional length field of CAN is replaced by two 4 bit-wide fields: the fragments number (FN) field contains the total number of intermediate fragments in the whole message, while the starting fragment (SF) field specifies the sequence number of the first fragment in the block and is used to resume an interrupted transmission. A value of SF equal to 0 means the transmission of a new message, while any other value reminds that a message transmission previously interrupted is being resumed. In the latter case, the SF value also shows the number of fragments of that message which have already been exchanged successfully in the previous transmission(s).

A block frame can contain up to 15 intermediate frames and exactly one final frame. In this way the L_BLOCK service can be used to transmit messages whose total length is up to 127 bytes. The L_BLOCK service does not worsen too much the responsiveness of a CAN network: the sending node, in fact, can be stopped after each (whole) fragment.

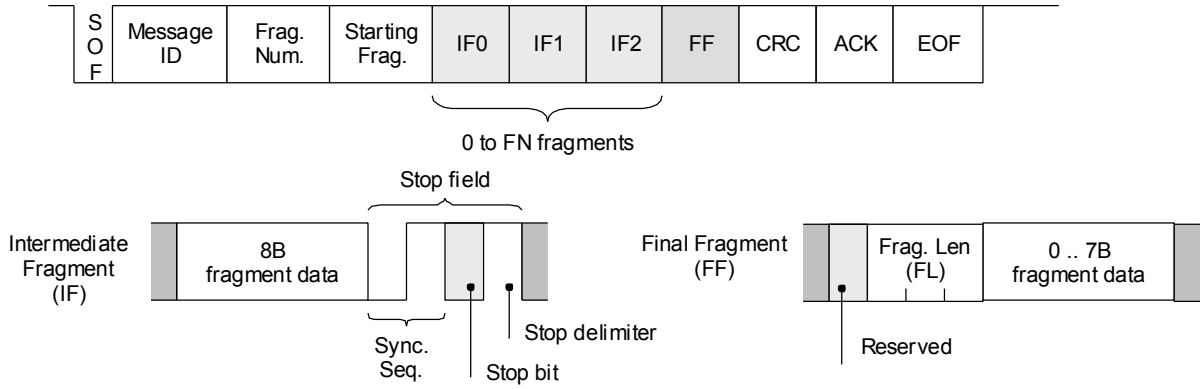

Fig. 4: Format of the BLOCK frame.

As shown in Fig. 5, the block transmission of a message consisting of 127 user bytes results in an efficiency which is about 0.90 in the best case (that is, when the block frame is never interrupted). In the worst case, that is when the transmission is repeatedly interrupted (and thus each block frame carries only one fragment) the efficiency decreases to 0.55.

In addition, to ensure a fair bandwidth share out among the different nodes, a technique similar to the priority promotion mechanism discussed in [10] can be adopted, which dynamically modifies the priority of the different nodes and enforces an overall behaviour which is similar to the token-based networks.

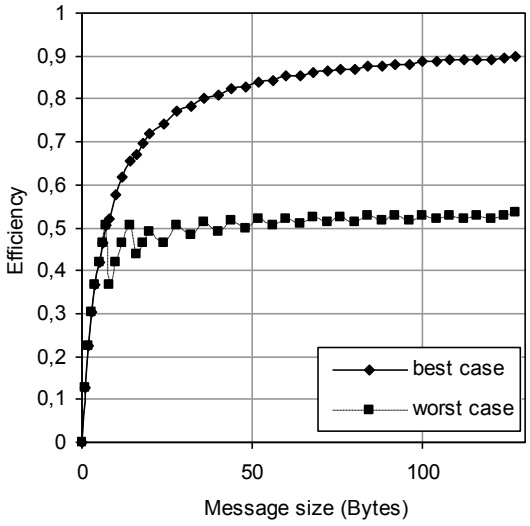

Fig. 5: Transmission efficiency of messages vs. message size.

3.3 High priority notifications

In fieldbus networks, a phenomenon which is known as priority inversion [9] could take place when the transmission of an urgent frame is delayed because of the current transmission of a lower priority message. This is due to the fact that the MAC mechanism is not pre-emptive and hence when a frame transmission is started it is always allowed to be taken to completion. It should be noted that the priority inversion problem is not relevant in the CAN networks because of the very limited frame size.

To provide an extremely high responsiveness for very urgent interactions, the CAN+ protocol can be modified to allow the communication media to be pre-empted. In particular, a new L_ERROR service is provided to abort the transmission of the current frame on the network. In practice, this service initiates the transmission of an ERROR frame which corrupts (and hence stops abruptly) the current transmission.

The L_ERROR service has two parameters: maximum priority (MP) and maximum remaining bytes (MRB). The error frame is effectively transmitted only if a frame is currently being exchanged on the network, the priority of that frame is (numerically) strictly higher than MP and the number of remaining data bytes to be sent is strictly greater than MRB (these information are known by each node in the network), so as to ensure bounded transmission delays without reducing the efficiency too much. If MRB is set to 0, for example, the frame can be stopped provided that the data field has not been completely exchanged yet. For efficiency reasons, the current transmission cannot be stopped if the CRC field is already being sent.

In CAN+, the L_ERROR service is mainly conceived to interrupt the cyclic process data exchanges when small-sized very urgent notifications (critical alarms) have to be sent. Whenever possible (as specified by the MRB parameter), message transmis-

sions should be preferably interrupted (gracefully) by means of the block mechanism (that is, using the stop bit).
A node needing to transmit an urgent frame as soon as possible has first to invoke the L_ERROR service in order to stop any possible current transmission of a low priority frame and, immediately after, to transmit the higher priority frame.

4 Conclusions

In this paper a new access technique has been introduced, which is mainly based on CAN but also inherits a number of good features from other kinds of fieldbus networks such as Interbus. The resulting CAN+ protocol has the same optimum responsiveness of the original CAN protocol, but ensures a higher efficiency for periodic data exchanges, which is comparable to Interbus. Moreover, a particular message transfer technique has been conceived which ensures a transmission efficiency similar to Profibus for the long messages and enables a completely dynamic management of the available bandwidth. To give some figures, the CAN+ protocol has a throughput that, in the usual operating conditions found in a factory environment, outperforms CAN by a factor which is about 5 for cyclic data exchanges and 3 for message transmissions.
In the near future, it is very likely that the cost of developing and producing network controller chips will decrease more rapidly than the cost of the physical communication supports, at least from the point of view of the distributed factory applications. Hence, networks which ensure an optimal use of the available bandwidth (even at the cost of a slightly more complex protocol) will probably be the preferred solutions. The substantial increase of the network performance of CAN+ enables to use the communication support as efficiently as possible and hence lower bit rates can be adopted on the network.
CAN+ constitutes a proper superset of the CAN protocol and it is backward compatible with all the existing applications (not devices) developed for that protocol suite. Moreover, it should be noted that with a minimum effort (an intermediate compatibility software layer) most of the applications conceived for other fieldbus networks (such as Profibus, Interbus or WorldFIP) can also be adapted to rely on a CAN+ communication support.

References

[1] German Institute of Normalisation, “Sensor/Actuator Network for Industrial Control Systems”, Technical Report 46, DIN E 19 258, 1993.

[2] International Standard Organization, “Road vehicles - Interchange of digital information - Controller area network for high-speed communication”, ISO 11898, November 1993.

[3] International Standard Organization, “Road vehicles - Interchange of digital information - Controller area network for high-speed communication” Draft Amendment, ISO 11898:1993/DAM 1, February 1994.

[4] G. Cena, C. Demartini and A. Valenzano, “On the Performances of Two Popular Fieldbuses”, in *Proc. WFCS’97 IEEE Workshop on Factory Communication Systems,* Barcelona, Spain, October 1997, pp. 177-186.

[5] German Institute of Normalization, “PROFIBUS Standard Part 1, 2,3 and 4”, DIN 19 145, April 1991.

[6] French Association for Standardization, “FIP Bus for Exchange of Information between Transmitters, Actuators and Programmable Controllers”, NF C46 601-607, March 1990.

[7] CAN in AUTOMATION International Users and Manufacturers Group e.V. “CAN Application Layer (CAL)”, CiA/DS201- CiA/DS205, CiA/DS207.

[8] CAN in AUTOMATION International Users and Manufacturers Group e.V. “CANopen Communication Profile for Industrial Systems Based on CAL”, CiA Draft Standard 301, Revision 3.0, October 1996.

[9] K. Tindell, A. Burns and A. Wellings, “Analysis of Hard Real-time Communications”, technical report YCS 222, Real-Time Systems Research Group, Department of Computer Science, University of York, England.

[10] G. Cena and A. Valenzano, “An Improved CAN Fieldbus for Industrial Applications”, in *IEEE Transactions on Industrial Electronics*, Vol. 44, No. 4, August 1997, pp. 553-564.