Real-Time Communication Protocol for Broadcast Based Networks

Carlos Franco[1], Luis Gutierrez[1], Raul Jacinto[2]

[1] Universidad de Guadalajara, CUCEA, Periférico Norte 799, C.P.44140,
Zapopan, Jalisco, Mexico
[2] DAVTI, Francisco Rojas Gonzalez 232, C.P.44650,
Guadalajara, Jalisco, Mexico
carlos.franco@cucea.udg.mx, luis.gutierrez@redudg.udg.mx, raul.jacinto@gmail.com

Abstract. One of the main challenges in the design of real-time distributed systems is the definition of the communications scheme that is required to perform all system tasks in the network in such a way that all time constraints are met. In this work is presented a real-time communications scheme based in broadcast networks. This proposal is the result of the study of other schemes such as Token bus, FIP, Profibus and CAN. It is presented a mechanism for media access control based on arbitrated message contention according to priority, which is given by a master plan. The proposed scheme considers periodic and aperiodic message delivery with static schedule and is based on common characteristics found in hard real-time systems (HRTS), includes a closed task set with time constraints, where critical tasks are defined as periodic tasks and it takes advantage of the broadcast nature of most networks found in real-time distributed systems. This work includes also a simulation scheme of the proposal.

Keywords: Real-time systems, real-time communication, broadcast networks, arbitrated message contention, message scheduling.

1 Introduction

Broadcast networks are present in almost all today's network environments, and bus topology [1] is widely used because of its low cost and ease of administration.
In the literature [7][9][11] are discussed several factors that contribute on the delay of message delivery in the communication process: queuing, packeting, switching and propagation. These factors are present in different stages of message transmission. There are in particular, six delay moments, which are represented in figure 1. These delays, which are identified as d1, d2, … d6, are presented in the different OSI model layers.

Advances in Computer Science and Electronic Systems
Research in Computing Science 52, 2011, pp. 265-275

Received 23/01/11
Accepted 20/02/11
Final version 07/03/11

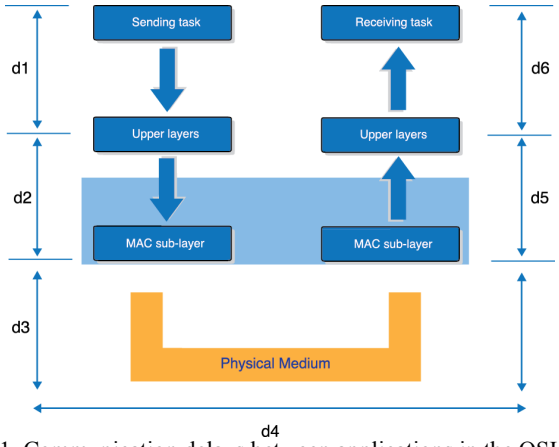

Figure 1. Communication delays between applications in the OSI model.

Delays d1 and d6 are presented when communication process is taking place in the upper layers of OSI model (layers 3 to 7). This is when the sending task and the receiving tasks are executed. Delays d2 and d5 are generated within layer 2 and generally are due to physical medium access control; however, d5 is significantly shorter than d2 because in the receiving task, there is no contention actually for physical medium. Delay d3 emerges when data is put into the physical medium and it is considered a queuing delay. Delay d4 is due message propagating in physical medium. As we can see, d2 is the hardest delay to deal with (it occurs in the medium access MAC sub-layer) because it is required to develop admission control mechanisms and packet scheduling schemes. Additionally, some techniques and communication models allow to shape traffic and to evaluate quality of service (QoS) requirements [5] for a particular application.

2 Problem Definition

As it is common in several hard real-time systems, execution plan is known in advance. In centralized or single-processor environments, the execution depends on a single entity -a dispatcher or a network referee- that defines which task is executed next. In other environments, such as Profibus [2], the execution control is distributed, where a token grants access to the network to its possessor. Both approaches have their own advantages but also disadvantages [6].

On one side, for the approach that uses a token, real-time execution can only be guaranteed to the node that holds token. On the other hand, the FIB approach, based in a bus referee turns out to be a very rigid scheme. CAN [3] is also based in the node priority, not in the priority of the task, situation that can lead to a problem in the real-time execution of the system. In this paper, an execution scheme based on distribution of control in the whole network is presented.

3 General Description of the Communications Scheme

It is assumed that there exist a closed number of participant nodes and there is already a master execution plan, where all tasks have been assigned to their corresponding processing entity. This plan is feasible and all deadlines are met. Each node has an instance of the global execution plan and it is assumed that each node has available all resources required to perform all the assigned tasks.
Assigned tasks to each node are not necessarily communication tasks in all cases, so it is possible to schedule tasks that do not require delivery of messages. It is very important to identify communication tasks from the others. Trough broadcast we can guarantee a minimum synchronization level between all participant nodes, because each sent message would be known and listened by the entire network. Global plan allows each node to know when a message is going to be sent, the order of sent messages and therefore, identify when and which message send each time.
Regarding to the transmitting node, when it sends a message, this is received by all network members (broadcast) but only the node that the message is directed to will process it. In that moment, as the received message has the address of the next node allowed for transmitting, it is assured an accurate synchronization in messages delivery. The authorized node sends its message according to the described procedure, which is repeated until the execution plan is completed. Figure 2 presents a general view of the communications network.

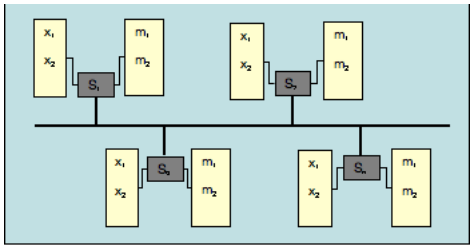

Figure 2. General view of the communications network

In the figure can be seen: the instance of the global plan, represented by the set X, the set of participant nodes $S = \{s_1, s_2, ...s_n\}$ and the schedulable set of messages assigned to each node, denoted by $M_{(S_i)} = \{m_1, m_2, ...m_n\}$. The proposed communications scheme allows delivery of periodic and aperiodic messages [8] with static schedule [10].

4 Protocol Description

4.1 Network topology

It is considered a bus network topology, where medium access is controlled with the implementation of a "logical ring" within the network to avoid collisions. The ring will work with a circulating token that will become the transmitting permission, so each node will send all required messages. Only one node will have the token at a time, so only one node can transmit at the same time. With this strategy, collisions are avoided and it is guaranteed that messages are delivered according to the schedule previously established.

4.2 Medium access control

Access to medium is serialized because the message delivery is developed according to global plan. When network is initializing, one node will randomly be designated as the master node and will verify the following:

- All nodes must have been assigned with all necessary tasks for the operation of the system
- Identify the sequence of messages to be delivered among all nodes
- Circulate a start message for the network initialization, in such a way that all nodes have an instance of the schedule, the messages and tasks assignment within the system
- Create the initial token for the network

Once the initialization token is circulated, each and every node will know which tasks must perform, what messages is going to receive, what messages are required to be sent and at what time these actions are going to take place. Therefore, the network will have a pre-established token circulation, and there will be guarantees that system feasibility can be accomplished.
At this point, all nodes have a copy of the schedule that is going to be performed and they must identify if there is aperiodic traffic to transmit.
Then, the master node sends the first token to initiate the normal execution of the system. This token will consider that in this moment, the highest priority for sending aperiodic traffic is for the master node. This is only for the initialization of the network operation.

4.3 Periodic and Aperiodic Traffic

There exist a set of periodic messages, a set of aperiodic messages (represented both by communication tasks) and other tasks that are not communication tasks. Periodic messages have very strict time constraints because they represent critical communication messages (real-time) whereas aperiodic messages have more relaxed time constraints because they do not represent critical communication tasks.

All messages have fixed length and the same structure: message ID, data field and next station to transmit node ID (token).

4.4 Message Scheduling

Periodic messages will be sent according to system's global plan and aperiodic messages will be locally sorted in every node according to its deadline. A message with closest deadline will have higher priority compared to a message with a later deadline. With this, messages will be delivered in the right order and system requirements will be satisfied. Provided the fact that broadcast based technology [4] (Ethernet, for example) is well known in their data propagation times and that now it already has a medium access control mechanism that avoids collisions, it is possible to accurately calculate if the delivery of a set of messages is schedulable in the network according to its deadline.

5 Formalization and Evaluation of the Protocol

The proposed protocol requires certain initial conditions for its operation:
C1: It is assumed a closed set of sites or participant nodes S
C2: Each node is assigned a set M of nodes and tasks that is schedulable
C3: Each node has a copy of the global message schedule, known as execution plan X.
C4: Messages have fixed length and a three-field structure: message identifier, data field, and next transmitting station or node identifier.
C5: Message transmission time δ is negligible
C6: All nodes receive the same message at the same time, including the origin node.

Let $S = \{s_1, s_2, ...s_n\}$ be the set of sites or participant nodes, $M_{(S_i)} = \{m_1, m_2, ...m_n\}$ the set of schedulable messages assigned to node S_i, such as $\forall m_j \in M$, we have that $m_j = (i, data, token)$, where i is the index or identifier of the message in the execution plan X, [redacted] represents the information that is going to be transmitted and $token$ represents the permission for the next node can actually transmit a message.
$X = \{x_1, x_2, ...x_n\}$ is the set of messages that conform the execution plan, where $\forall x_i \in X$, we have that $x_i = (s_{origen}, m_j, s_{destino})$ where s_{origen} is the node that sends the message m_j to node $s_{destino}$.
We have that $\forall s_n \in S, \exists p_n$, such as p_n is a local administrator that manages communication activities in each node and is responsible to send, receive and deliver a message m_j from node S_{origen} to node $S_{destino}$ by using the functions $create_i(m_j)$, $send_i(m_j)$, $receive_i(m_j)$ and $deliver_i(m_j)$.
Functions $create_i(m_j)$ and $deliver_i(m_j)$, are higher lever functions than sending or receiving the message m_j at the nodes s_{origen} or $s_{destino}$. Reception of a message does not imply the immediate delivery of the message because delivery is conditioned by

timing parameters.

$send_i(m_j)$ is the function that assures the sending of the message m_j from node s_{origen} to node $S_{destino}$

$receive_i(m_j)$ is the function that assures the reception of the message m_j from transmission medium to node $S_{destino}$.

$create_i(m_j)$ is the function that communicates to the application level to generate messages that eventually are going to be send to the communication medium.

$deliver_i(m_j)$ is the function that assures the delivery of the message m_j to the node $S_{destino}$.

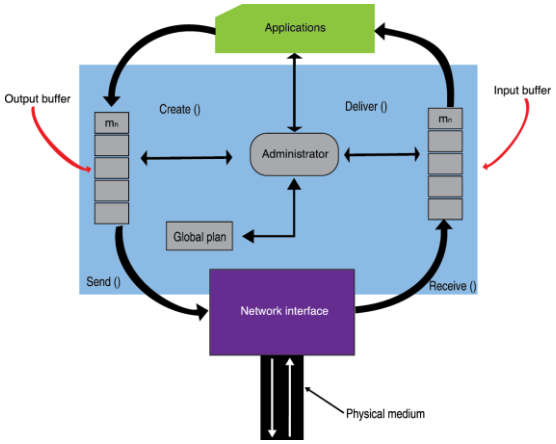

Figure 3. Architecture of a communication node within the network.

It is necessary to perform a simulation process in order to verify the average arrival time of messages in several executions of the global plan and also to verify that there is no contention for the physical medium.

Next, are presented the considerations taken into account for the simulation:

1. It is assumed that there is a closed set of participant nodes
2. There exists a previous fixed schedule for periodic messages
3. It is assumed that the periodic messages set that is going to be send is feasible
4. Every node that comprises the network has a copy of the execution plan (to avoid medium access contention) and therefore, knows which are the messages to send.
5. Nodes do not require an explicit synchronization, because having a copy of the global schedule is an intrinsic mechanism of synchronization.
6. Messages are sent according to the sequence established in the execution plan.
7. In each message, a token is transmitted. If the token is released, the next site or node in the list can send its message.
8. The message has a three field structure:
    a. Message identifier
    b. Data

c. Token

The token acts as an identifier for the next node to transmit.

Based on the simulation previously described, it is expected:

1. Know the average arrival time of messages in every node
2. Verify that no exists contention for the medium.

For the simulation of aperiodic tasks there are included two proposals, the first (proposal 1) consists in applying the criteria that the node that has the token, is the node that send the aperiodic traffic and only in case this node has no aperiodic messages to send, the token goes to the next node according to global plan.
The second (proposal 2) applies the criteria of circulating the token (to send aperiodic traffic) in a predefined order (which is defined for example at the moment of the network initialization). It is convenient to remark that aperiodic traffic will be sent only after sending all periodic traffic (real-time traffic) trying to take advantage of network idle times.

6 Results

For the simulation, it was performed a work that consisted in the evaluation and the comparison of a couple of simulation scenarios described in the previous section. In order to perform those comparisons it was required to design a simulation environment that could allow creating the network conditions to simulate. This implied the development of the following activities:

1. Determine the number of processors in the network
2. Generate a set of schedulable tasks for each node in the network, with the processing load in each task chosen for each node.
3. Determine the load of periodic messages present in each node.
4. Generate a global schedulable plan that includes the established communication tasks
5. Determine aperiodic messages load present in each node
6. Simulate the behavior of the communications network
7. Get the results
8. Interpret the results
9. Conclusions

6.1 Number of processors.

Simulation environments were developed with 5 and 10 processors. In this work are only presented simulation results with 5 processors because results obtained with 10 processors are very similar in both cases.

6.2 Generation of a set of schedulable tasks.

There were created random sets of tasks for each of the network nodes, which are completely schedulable in each one of the processors. The simulation presents two cases, when the processor load is 60% and when it is 80%. This means that in each node we had a maximum load of tasks for each processor equivalent to 60% or 80% and that set of tasks was schedulable.

6.3 Determination of the periodic load in each participant node.

From the total number of tasks assigned to each node, not all of them are communication tasks, that is, some tasks that do not require communicating with other nodes and do not require information exchange to perform correctly. This means that in the simulation scenarios it is required to define from the total number of tasks or messages assigned to each node, how many messages are periodic. For this simulation, it was considered that 10% of the tasks assigned to each node are communication tasks.

6.4 Generation of an execution plan in each node: Global plan.

As each set of tasks is schedulable in each node, now it is required that all sets of tasks are schedulable when they combine. This is, if the sets of tasks are schedulable in the local environments, there is no guarantee that all local plans can be schedulable when it comes about a global schedule. The simulation scenario then creates local plans that are also schedulable in the global level. The simulation performed did not detect any problem regarding to collisions or missed deadlines in the system messages or tasks.

6.5 Aperiodic messages load.

From the total of communication tasks present in each node, most are periodic tasks, but there were considered different levels of aperiodic tasks load, to analyze the behavior of the network to these variations.
For the task processing load of 60%, it was simulated a message load of 10% and an aperiodic messages load of 5%, 10%, 15% 20% and 25%.
For the task processing load of 80%, it was simulated a message load of 10% and an aperiodic messages load of 5%, 10%, 15% 20% and 25%.

6.6 Simulate behavior of the communications network.

To simulate the behavior of the communications network, it is required to clear up that it was considered a 2 time units delay for the minimal transmission and propagation time (communication time between two nodes, the closest) and a 5 time units delay for the maximum transmission and propagation time (communication time

between two nodes, the farthest). Tasks will have established periods between 100, 50 and 40 units. The execution units, the period and deadlines of each task or message are generated randomly. Precedence relationships are established randomly and after processes have been generated for each processor.

6.7 Obtained results.

Obtained results have to do with the communications network behavior and consist in the following:

- Response time for periodic messages
- Response time for aperiodic tasks

Results are shown in tables 1 and 2.

Task load: 60% of processor capacity					
Periodic messages load: 10%					
Aperiodic message load	**5%**	**10%**	**15%**	**20%**	**25%**
Proposal 1	2.96	3.1	3.52	3.67	3.98
Proposal 2	2.8	2.96	3.41	3.56	3.78

Table1. Simulation results for 60% of tasks load

Task load: 80% of processor capacity					
Periodic messages load: 10%					
Aperiodic message load	**5%**	**10%**	**15%**	**20%**	**25%**
Proposal 1	2.56	3.01	3.38	3.71	3.95
Proposal 2	2.52	2.91	3.31	3.66	3.92

Table2. Simulation results for 80% of tasks load

The response time for periodic messages is constant, because its attention is not in risk because it is assigned in fixed intervals. Results were obtained in the format in what the simulation tool delivers them, however it was necessary to treat them so they could be interpreted and plotted.

6.8 Interpretation of results.

As it can be seen in tables 1 and 2, there is a slight variation in the response times for the delivery of aperiodic traffic when the load percentage of processor capacity is increased in each node. This can also be noticed in figures 4 and 5.

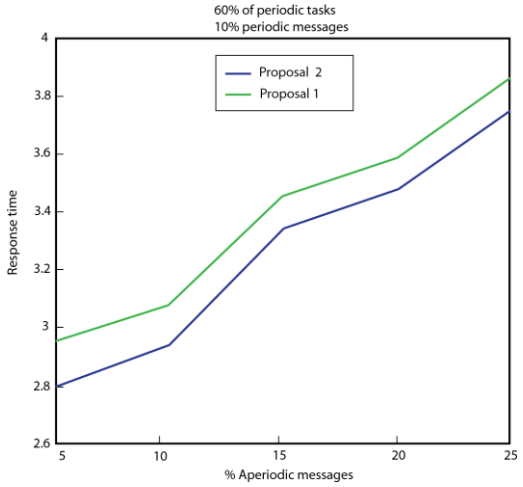

Figure 4. Comparing response time for aperiodic messages with 60% of processor load.

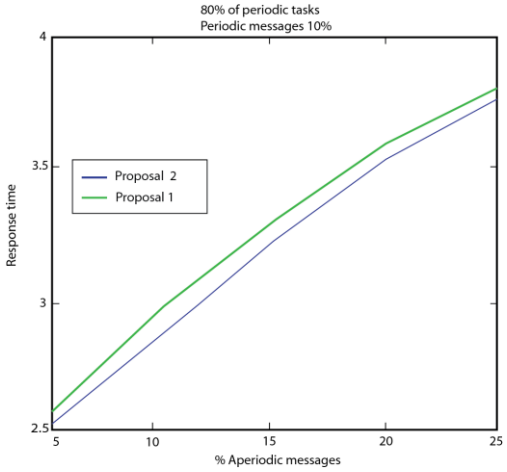

Figure 5. Comparing response time for aperiodic messages with 80% of processor load.

In figure 4, we can appreciate the behavior of the average propagation and transmission times when tasks load is 60% of the processors capacity. In figure 5, we can appreciate the behavior of the average propagation and transmission times when tasks load is 80% of the processors capacity. In both cases are considered the two proposals described previously for dealing with aperiodic traffic.

6.9 Conclusions

According to behavior presented in graphics and tables, we can conclude that the first proposal offers better response times. However, as the processor load is increasing, the second proposal tends to a more stable behavior and to offer similar response times, whereas the firs proposal as the processor load is increasing tends to increase also its response times. This behavior is consistent with the idea that the first scheme

is unfair under the perspective of how many nodes can send aperiodic traffic. That is the reason why it has better performance when there is not much load in the processor. Nevertheless, the second scheme, as it is more balanced (fair) it tends to improve the performance when processor load increases.

The first proposal is more efficient for cases when processor load is not so demanding and in cases where aperiodic messages is kept within the 2%, referred in the literature by Stankovic [12]. The second proposal is better for cases where aperiodic messages load is above average. Even this scenario is hard to find in practical situations, it is interesting for analysis. According to these results, it could be evaluated a third proposal that have a medium between both presented proposals regarding the amount of aperiodic messages it can handle and possibly it can be obtained a better scenario between response times and stability.

References

1. Weaver and C. Summers, “The IEEE Token Bus-A Performance Bound on GM MAP” IEEE transactions on Industrial Electronics, Volume 35, Issue 1, feb 1988.
2. E. Tovar and F. Vasques. “Real-time fieldbus communications using Profibus networks”. IEEE Transactions on Industrial Electronics. Volume 46, Issue 6, Dec 1999 Page(s): 1241 – 1251.
3. H. Kaschel, E. Pinto. “Análisis protocolar del bus de campo CAN” Reporte de trabajo. Facultad de Ingeniería, Depto. de Ingeniería Eléctrica. Universidad de Santiago de Chile. Chile, 2002.
4. G. LeLann and N. Rivierre, “Real-Time communications over broadcast networks: The CSMA-DCR and the DOD-CSMA-CD Protocols”. INRIA Report RR1863, 1993.
5. P.Ferguson and G. Huston. “Quality of Service: delivering QoS on the Internet and in corporate networks”. John Wiley & Sons press. USA, 2000.
6. C. Aras, J. Kurose, D. Reeves and H. Schulzrinne, "Real-Time Communication in Packet-Switched Networks", Proceedings of the IEEE, Vol. 82, No. 1, pp. 122-139. Enero, 1994.
7. Cottet, Francis, Joêlle Delacroix, Claude Kaiser, & Mammeri, Zoubir, “Scheduling in real-time systems”. England: John Wiley and sons press. USA, 2002.
8. Y. Atif and B. Hamidzadeh, “A Scalable Scheduling Algorithm for Real-Time Distributed Systems”, Proceedings of the 18th International Conference on Distributed Computing Systems, May 26-29 1998, pp. 352-359
9. D. Verma, H. Zhang and D. Ferrari. “Delay jitter control for Real-Time communication in a packet switching network”. In proceedings of TriComm. 1991.
10. Buttazzo, Giorgio C. “Hard real-time computing systems: predictable scheduling algorithms and applications”. Kluwer Academic Publisher. Boston, 1997.
11. Cheng, Albert M.K. “Real-time systems: Scheduling, analysis, and verification”. John Wiley and sons press. New Jersey, 2002.
12. Stankovic, John A., Spuri, Marco, Ramamritham, Krithi, & Buttazo, Giorgio. “Deadline scheduling for real-time systems: EDF and related algorithms”. Kluwer Academic Publisher Boston, 1998.