Ethernet as a Real-time Network in a High Performance Distributed Remote Terminal Unit

Rogério Paulo, António Carrapatoso, Fernando Gomes, Fernando Xavier
{rdpaulo, amc, fgomes, fxavier}@se.efacec.pt
EFACEC Sistemas de Electrónica, S.A.
Rua Engº Frederico Ulrich, Apt. 3078, 4471-907 Maia, Portugal

Abstract - **The purpose of this presentation is to describe the ongoing work of evaluation and implementation of Ethernet as a real-time network. Ethernet would be integrated into a high performance distributed Remote Terminal Unit (RTU), the RTU500. Results of shared Ethernet performance tests are presented and an upper-layer proprietary protocol architecture is proposed.**

I. INTRODUCTION

The RTU500 is a highly flexible and scalable remote terminal unit targeted for electrical, water or gas utility SCADA systems.

A large RTU can only be sustained by a distributed networked architecture. Considering the several hundred scattered acquisition points that must be monitored in large plants, a centralised architecture would involve significant cabling costs and difficulties as well as a high concentration of processing power.

Tight time constraints, mainly in electrical utilities, mean that a large bandwidth real-time network must be used. Ethernet offers cost-effective bandwidths of 10 Mbps and 100 Mbps. It is largely accepted in the market, being a *de facto* standard in non real-time applications with a wide array of network controllers, boards and other equipment. As more and more industrial automation manufacturers provide Ethernet interfaces at the control and field levels, it becomes more interoperable than any conventional fieldbus or real-time network. However, Ethernet provides only a MAC sublevel interface and no determinism.

These are the main reasons why we have proposed to evaluate Ethernet as a high performance network.

II. RTU500 REMOTE TERMINAL UNIT

The primary functions of an RTU are data acquisition, data processing, data reporting and control execution. Some of the key features of the RTU500 are: (1) multiple protocol support; (2) up to 60x256 I/O points; (3) Intelligent Electronic Device (IED) interface; (4) centralised and decentralised automation functions, and (5) local Human Machine Interface (HMI) with data/event logging, alarms handling and data processing capabilities. These features allow the RTU500 to be used as a typical RTU, a data concentrator or a low-cost SCADA control centre.

A. Architecture

The RTU, as shown in figure 1, is composed of three types of subsystems: the central unit, the acquisition units and the synchronisation unit.

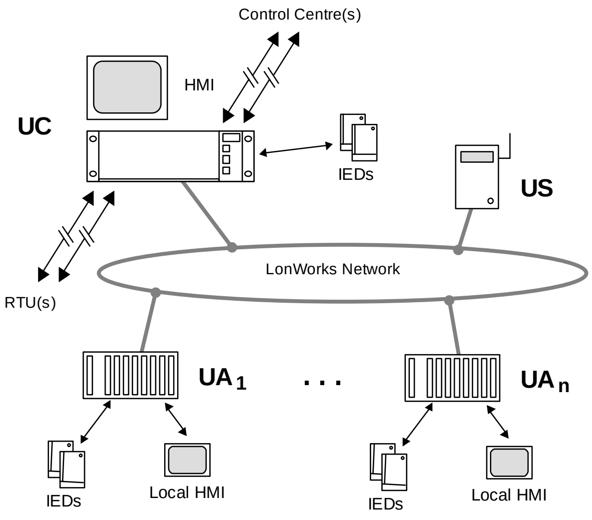

Fig. 1 - RTU500 distributed architecture.

The central unit (UC), in single or redundant configuration, is an industrial PC running Windows NT and is responsible for: (1) maintaining a SCADA data point current state database and SCADA log/archive database, (2) downstream RTU communication, (3) upstream control centre (CC) communication, (4) IED communication and (5) HMI management.

The acquisition unit (UA) is an Intel based embedded PC running a real-time kernel. Up to eight plant interface boards can be connected to the CPU via a proprietary bus allowing up to 256 digital input points (1, 2, 3 or 4 bits), 64 analogue input points or 128 digital output points. The maximum number of UAs connected to the same UC is 60.

The synchronisation unit (US) is GPS based and delivers absolute time (UTC) information to all the other units.

This distributed architecture allows not only the distribution of processing power among the units, but also the separation of functionality types. Hence, real-time functions with tight time constraints are only assigned to the UAs. Other functions, like system configuration and RTU/CC communication, are assigned to the UC.

The currently used network, LonWorks [1], uses a priority enabled predictive p-persistent CSMA/CD medium access mechanism, running at a maximum speed of 1.25 Mbps. Despite being robust and deterministic, LonWorks performance is limited due to the complex nature of the network controller chips. The seven-layered protocol is essentially run by software in a triple 10 MHz microprocessor ASIC with shared memory. This ASIC and its parallel interface to the CPU introduce a bottleneck that prevents systems from meeting some bandwidth requirements.

B. Functional description and network requirements

The RTU real-time functions that rely heavily on network performance are: (1) data exchange between the UC and the UAs; (2) distributed automation functions among the UAs, and (3) time synchronisation.

Synchronisation is the most time critical function and must guarantee that all the units are synchronised with one millisecond precision.

Distributed automation requirements vary a great deal with the application, but the most time critical would fall in the range of 20 to 40 milliseconds from event occurrence in one UA and event notification in consumer UAs.

The UC-UA communication involves UA parameter loading, time stamped event reporting to the UC and control issuing from the UC. This is the least time critical function and the minimum network latency requirements are about one hundred milliseconds or more.

Other requirements, like fibre optics physical medium for harsh environments, redundancy for at least all active equipment and distances between nodes up to one kilometre, must also be met.

III. 10 MBPS SHARED ETHERNET PERFORMANCE

The Ethernet CSMA/CD 1-persistent medium access mechanism [2] together with the exponential backoff algorithm for collision resolution results in a low probability of frame loss due to collisions, but introduces an uncertainty in the maximum frame delay. In order to evaluate Ethernet response we have conducted a series of tests in a 10 Mbps shared network.

A. Test environment and characteristics

All the tests used the same 10BaseT shared network composed of one 24 port hub and 23 computers with CPU clock speeds ranging from 200 MHz to 400 MHz.

Only an unacknowledged link level protocol was used (802.2 LLC class I [3]). One of the 23 nodes is the test master and the remaining 22 the test slaves. The master first downloads the test parameters to the slaves, then synchronises each slave one by one and finally signals the slaves to start the test. The master itself does not send any frames.

The confirmed synchronisation mechanism, together with a continuous clock drift adjustment in every slave, ensures that all nodes are synchronised with 0.5 millisecond precision. The main goal of the tests was to simulate a worst case collision scenario. The tests consist in having the slaves send a number of frames (1 to 3) at the same time and measure the transmission time of each frame. Every slave sends broadcast time-stamped frames and calculates each received frame's transmission time based on the arrival time. Three thousand tests were performed for each configuration (table 1).

Number of slaves	Frame payload length (bytes)	Number of frames
10	50, 100 and 250	1
16		
22		
20		2 and 3

Tab. 1 - Ethernet test configurations.

B. Results

Figures 2 to 4 present the actual results, no frames were lost in any of the tests. These results indicate that shared 10 Mbps Ethernet can meet the typical time requirements in a RTU500 configuration with up to 20 UAs.

Note that, with a 250 byte payload, the network can sustain up to three frames per node every 50 milliseconds (99.9% best results). The longest transmission time was 80 milliseconds for one 250 byte frame.

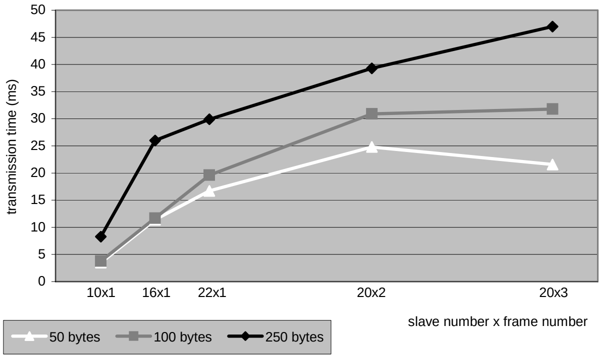

Fig. 2 - Maximum transmission times (99.9% best results).

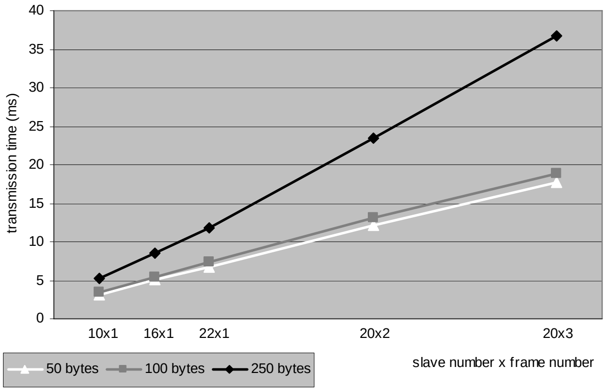

Fig. 3 - Maximum transmission times (99% best results).

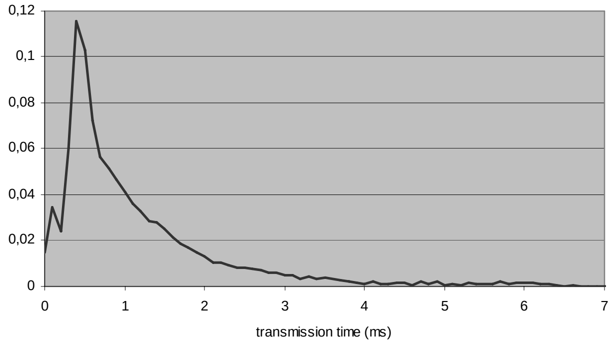

Fig. 4 - Empirical result distribution (22 slaves, 100 bytes, 1 frame).

C. Test conclusion

No tests were performed with switched or 100 Mbps Ethernet. Nevertheless, the maximum transmission times should be much lower than the ones observed, specially in collision-free switched networks. From these results and assumptions we conclude that Ethernet is a suitable technology for the distributed RTU.

IV. PROPOSED UPPER-LAYER PROTOCOL

The lack of suitable upper-layer protocols means that a proprietary real-time protocol architecture should be introduced. This architecture must ensure that (1) it is compatible with multiple protocols, (2) Ethernet frame lengths do not exceed a maximum number of bytes, and (3) nodes have a limited frame generation rate.

Since one of the typical bottlenecks in real-time networks is node performance and complexity and since there are no routing needs, we propose a collapsed stack, as shown in figure 5.

The IEEE 802.2 class I (unacknowledged connectionless service) [3] implementation should be extended to provide at least three priority levels and to allow integration with switch based traffic class queuing.

OSI reference		**Proposed stack**		
Application		*Sync Service*	*Distributed Database Service*	*Telecontrol Communication Service*
Link	LLC	IEEE 802.2 class I		
	MAC	IEEE 802.3		
Physical				

Fig. 5 - Protocol architecture stack.

The proposed application level protocol (figure 6) would provide service types to match the RTU real-time functions: (1) synchronisation, (2) distributed database for distributed automation functions and (3) telecontrol communication. These services would be mapped to the three priority classes at the link level.

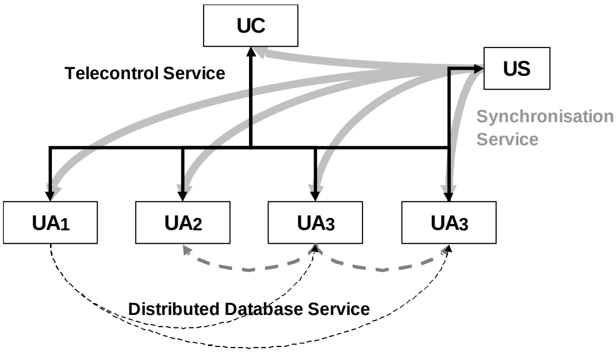

Fig. 6 - Proposed protocol architecture.

A. Synchronisation service

Given the large fluctuation of transmission times (far beyond one millisecond) a special mechanism must be devised to ensure the correct synchronisation. The proposed service will consist of three steps: (1) warning frame, (2) confirmed time broadcast, and (3) synchronisation confirmation. Upon receiving a broadcast warning frame from the US, a node will not produce additional traffic for a *Ts* period (with the exception of the frames already queued at the controller). This allows a significant reduction in the contention time for the next sent frames. The US then sends a synchronisation broadcast message with UTC time at the moment the frame leaves the controller. This frame signals one of the UAs to acknowledge the synchronisation with a reply containing the internal node delay. Upon reception of the acknowledgement frame, the US will be able to calculate the synchronisation error. Should this error be less than one millisecond a broadcast synchronisation confirmation message is sent to all nodes, otherwise the synchronisation process must be repeated.

B. Distributed database service

The distributed database service consists of periodic and aperiodic unacknowledged broadcasts of state information among the UAs. Each UA is assigned two time values. The first value (*Tp*) sets the periodic broadcast frequency. The UA will send the periodic broadcasts even if there is no change in its state. This allows some protection against lost frames and also a keep-alive mechanism for node failure detection. To ensure a fast response to events, the node may broadcast its state on-event. The second value (*Tap*) sets the maximum aperiodic broadcast frequency ($Ts < Tap < Tp$). To reduce unnecessary software frame processing at the receivers, each node sends its state to a specific Ethernet multicast address. The state image is strictly dependent on the particular automation function and may include both variable states and data processing results.

The major drawback is that information loss may occur, if, for example, a single digital point changes state twice in less then the *Tap* period. If such behaviour is undesirable the user must implement additional mechanisms. This loosely-coupled mechanism is, however, very simple to implement and allows fast removal and entrance of UAs into the distributed database.

C. Telecontrol communication service

The ultimate goal of the UA-UC communication is to achieve maximum consistency between the physical variables and the UC image of the plant. This requires a service that ensures no events are lost, and all events are reported in the correct order. Furthermore, it must guarantee the current state is set correctly upon UA or UC initialisation. These requirements enforce the use of a more complex protocol. This protocol must include acknowledgement and flow control, as well as, application data definitions and functions.

The proposed service is based on the IEC 870-5 family of telecontrol communication standards [4, 5, 6 and 7]. Additional link layer functions provide a master-master confirmed service with a window size of one and the possibility of automatic repetitions for error recovery. The application layer includes time-stamped event reporting, cyclic analogue point reporting, single or general interrogation and application confirmed control issuing.

V. CONCLUSIONS

Even though Ethernet response is in essence probabilistic, bit rates of 10 Mbps and more, switched collision-free networks and emerging switch-based priority standards will ensure, with a suitable upper-layer protocol architecture, a deterministic performance compatible with the RTU requirements.

Some issues, such as switch delay or switch frame dropping, remain untested. Therefore the next step will be to conduct further testing on an actual RTU with a prototype protocol implementation in both shared and switched networks.

REFERENCES

[1] ECHELON, "LonTalk Protocol Specification, Version 3.0", ECHELON, 1994

[2] ANSI/IEEE, "Carrier Sense Multiple Access with Collision Detection (CSMA/CD) IEEE 802.3", IEEE, 1985

[3] ANSI/IEEE, "Local Area Networks: 802.2 Logic Link Control", IEEE, 1985

[4] IEC, "IEC 870-5-1 - Part 5: Transmission protocols. Transmission frame formats", IEC, 1990

[5] IEC, "IEC 870-5-2 - Part 5: Transmission protocols. Link transmission procedures", IEC, 1992

[6] IEC, "IEC 870-5-3 - Part 5: Transmission protocols. General structure of application data", IEC, 1992

[7] IEC, "IEC 870-5-5 - Part 5: Transmission protocols. Basic application functions", IEC, 1995