Proceedings of the 2006 IASME/WSEAS International Conference on Energy & Environmental Systems, Chalkida, Greece, May 8-10, 2006 (pp145-150)

Minimizing of the Communication Overhead in Distributed Control Computer of Power System Using Correct Partition and DMA

J. ZDENEK
Faculty of Electrical Engineering
Czech Technical University in Prague
Technicka 2, Prague 6, 166 27
CZECH REPUBLIC

Abstract: - Design problems of the distributed control computer of power systems are considered with a special emphasis on system, hardware and software architecture of the local computer network of a traction vehicle. The local computer network based on essential communication support of DMA transfer to unburden application host processor from network overhead is presented. The communication overhead on higher interrupt levels of CPU (which are important to be available to process control tasks) is minimized to several percent of CPU throughput. All network activities are easily processed by the host (application) processor, no additional communication coprocessor is required. Communication APIs are simple and user friendly.

Key-Words: - Power System, Distributed Control Computer, Local Computer Network, DMA Transfer

1 Introduction

Design of control computers of power systems is typically based on a distributed architecture [2],[3],[5],[8]. In such control system several (or many) complete local computers (i.e. with CPU, program and data memory and input/output channels) cooperate together to control dedicated power system application. To be common cooperation of separate computers possible and efficient it is necessary to organize communication channels with sufficient information capacity between them. Good design of a communication system of distributed control computer is important task for system, hardware and software designers and has basic influence on the overall system parameters [1],[4],[6]. Computer nodes together with communication paths create local area computer network (LAN). Distributed control computer (DCC) of an electric locomotive is a typical example of such LAN. We will present actual solution of the locomotive DCC LAN which utilizes powerful direct memory access (DMA) controllers to minimize the system overhead by a communication subsystem [7].

2 Channel Bandwidth and Overhead

The required communication channel bandwidth is dependent not only on the actual application but intensively on the correct partition of application tasks among nodes of DCC as well. DCC LANs utilize serial communication channels of many different protocols, but with a common problem. Serial channels impose additional burden on the application CPU (communication overhead) and its interrupt system. Some of the communication tasks can be placed successfully on the system background where CPU computational power is readily available (framing and decoding data). Coded data transfer in both sides (transmitter/receiver) is time critical and first of all in the receiver side it can not be postponed to the time when CPU has free time to process communication data [3], [7], [8].

3 DMA Based LAN Architecture

3.1 DCC System Partition

To define DCC system partition we have to select from pool of contractor system requirements group of all user tasks that are to be assigned to suitable DCC nodes. Further we have to collect together the user tasks by application of criterial (threshold) functions that can run together in one node of the DCC [8]. Finally for the defined groups of tasks we have to design suitable hardware and decide which system software is to be used for scheduling the user tasks. Selection of groups is iterative process which we have to continue until no user task is unassigned. Let us define following symbols:

$pjrq_i$ – project function requirements,

$ThUT_i$ – user task selection threshold function,
$pjSW$ – project software,
ss_i – system software support, RTOS etc.,
ut_i – user task,
utg_i – user task group,
$utgXX_i$ – user task group selected by *XX* criterion,
$ThFR_i$ – task function requirement threshold func.,
$ThCM_i$ – inter-task communication threshold func.,
$ThTP_i$ – CPU throughput threshold function,
$hwNode_i$ – hardware network node with CPU.

Then we have:

$$ut_j = ThUT_j(\sum_i pjrq_i) \quad (1)$$

$$pjSW = \sum_i ut_i + \sum_j ss_j \quad (2)$$

Further we define application function requirements (3), inter-task data flow rate (4) and required CPU throughput (5).

$$utgFR_j = \sum_i ThFR_j(ut_i) \quad (3)$$

$$utgCM_j = \sum_i ThCM_j(ut_i) \quad (4)$$

$$utgTP_j = \sum_i ThTP_j(ut_i) \quad (5)$$

Finally we get group of tasks utg_j (6) to that proper hardware $hwNode_j$ (7) will be assigned.

$$utg_j = utgFR_j \cap utgCM_j \cap utgTP_j \quad (6)$$

$$hwNode_j \leftarrow utg_j \quad (7)$$

3.2 DCC LAN System Architecture

DCC of the electric locomotive (type 93E) (Fig. 1) is three level hierarchical system with serial communication channel between nodes. The nodes have local parallel busses to expand functionality of nodes easily. Levels one and two communicate through the Q-bus. Backbone communication channel of Q-bus is Q1 NBP protocol 1 Mbps (Unformatted) serial bus with substantial transfer support of DMA controllers in both ends of transmission path. Q2 bus is spare and is in service when Q1 faults. Q3 bus supports mutual precise

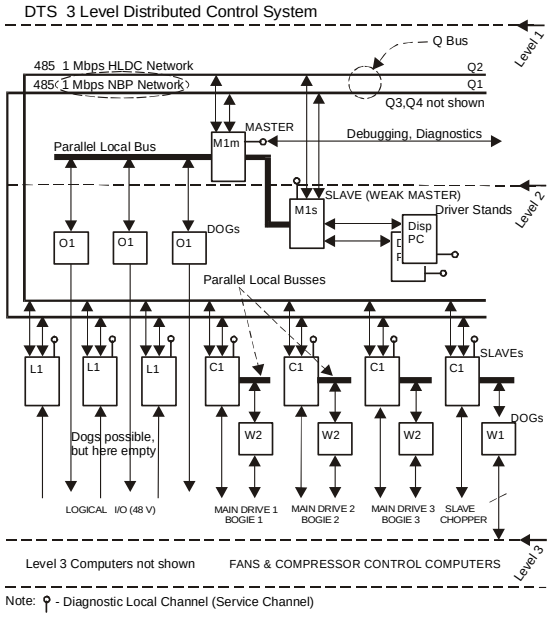


Fig. 1. Distributed control computer structural view

synchronization of control computer of the bogie converters.

3.3 DCC Node Core Architecture

Each DCC node type has uniform host computer core (Fig. 2) [7] including network communication section and different specific hardware unit based on the target node function. Fast multi-channel DMA controller, vectored interrupt controller and timers are integrated inside core. Q1 bus utilizes fast (1 Mbps) NBP UARTs on both communication sides. UARTs have no hardware frame address detection. DMA controller organizes virtual dual-port RAM area between CPU and UART [7]. DMA controller read/write cycle is not shadow type (no CPU bus cycle delay) it uses cycle stealing.

3.4 DMA vs. Interrupt Data Transfer

Let us define following symbols:

tIRtst - interrupt service transfer time,
tIRxxx - interrupt service xxx source overhead,
tIRhpd - higher priority service max delay,
tIRrql - interrupt request max time latency,
tIRsct - save context time,
tIRmdt - process and move data time,
tIRrsct - restore context time,
tIRirt - interrupt return time,
tDMAxxx - DMA service source overhead,
tDMArql - DMA request max time latency,
tDMArdt - read source data time,
tDMAwr - write data to destination time,
tDMAtst - DMA transfer time,
Then length of interrupt service to move data is:

$$tIRtst = \sum_i tIRxxx_i = tIRhpd + tIRrql + tIRsct + tIRmdt + tIRrct + tIRirt \quad (8)$$

and length of DMA service to move data is:

$$tDMAtst = \sum_i tDMAxxx_i = tDMArql + tDMArdt + tDMAwrt \quad (9)$$

For used mid-range core CPU, relation between *tIRtst* (8) time and *tDMAtst (9)* time is:

$$tDMAtst \leq 70 \cdot tIRtst \quad (10)$$

In the presented application and the communication speed 1 Mbps the received data byte have to be read every 10 μs. Time *tIRtst* ≈ 40 μs i.e. the interrupt system can not be used to process the incoming data. Time *tDMAtst* ≈ 600 ns and the received data are processed easily (10).

3.5 Communication Programmer Model

Programmer system model of the communication service (mail service) (Fig. 3) is the master/slave type and utilizes message queues on both sides. Main features are as follows (details in [6],[7]):

- message transfer method,
- up-to 512 kbps user data throughput i.e. 64 kBps,
- 1 Mbps communication speed (unformatted),
- low application processor overhead - 3 % approx.,
- configurable message length - 4, 8, 16, 32 bytes,
- master-slave control, deterministic bus access,
- up/down dual message cycle,
- message cycle time 1 ms,
- max 15 slave nodes,
- run-time user reconfigurable.

3.6 Mail Software Design

Network communication utilizes two types of data transfer cycles – Network Management Transfer (NMT) cycle and Process Data Transfer (PDT) cycle. The software design utilizes following resources of the node core: Master side – two DMA channels, NBP UART, timer, one interrupt level, and background. Slave side – two DMA channels, NBP UART, timer, four interrupt levels, background. Structure of NMT cycle is explained in Fig. 4 and Fig. 7. Details of PDT cycle see [7]. NMT and PDT cycles timing and mutual

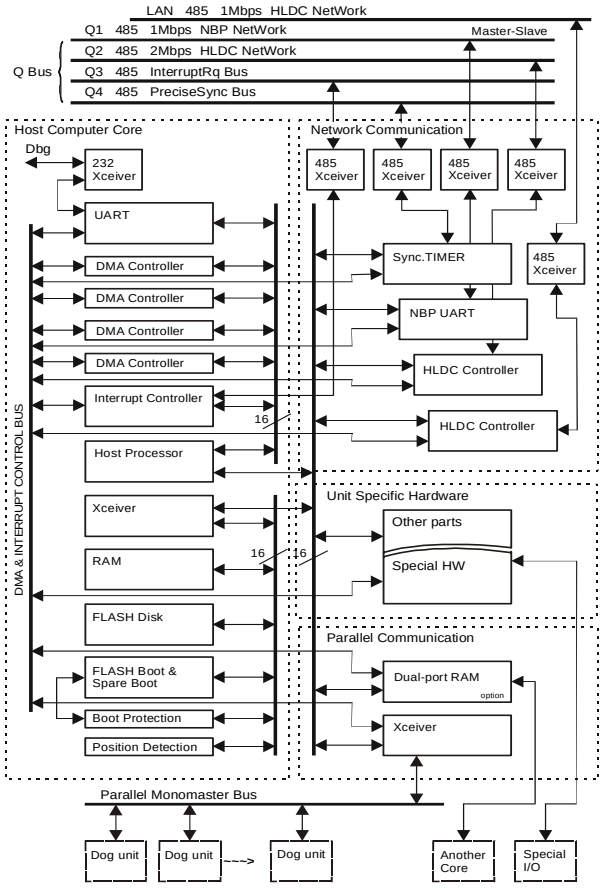


Fig. 2. Node uniform core structure

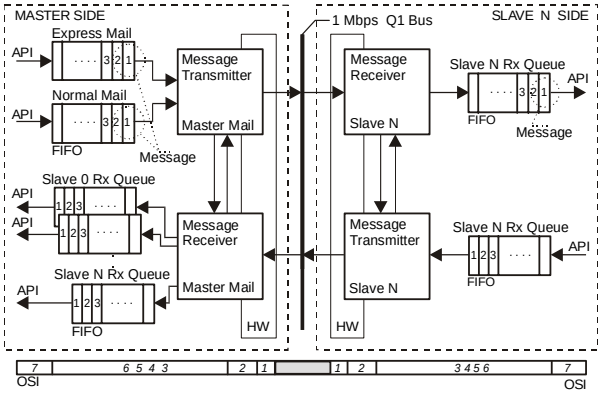


Fig. 3. Q1 bus 1 Mbps mail programmer model

coordination of interrupts, DMAs and DMAs interrupts are depicted in Fig. 4, Fig. 5 and Fig. 6. The records of Q1 bus traffic from the logic analyzer show frame header response time of the addressed slave (T_{rt} = approx. 30 μs) (Fig. 8) and one complete bus double message cycle (Fig. 9) (data rate v ≈ 64 kBytes per sec, overhead ≈ 49%).

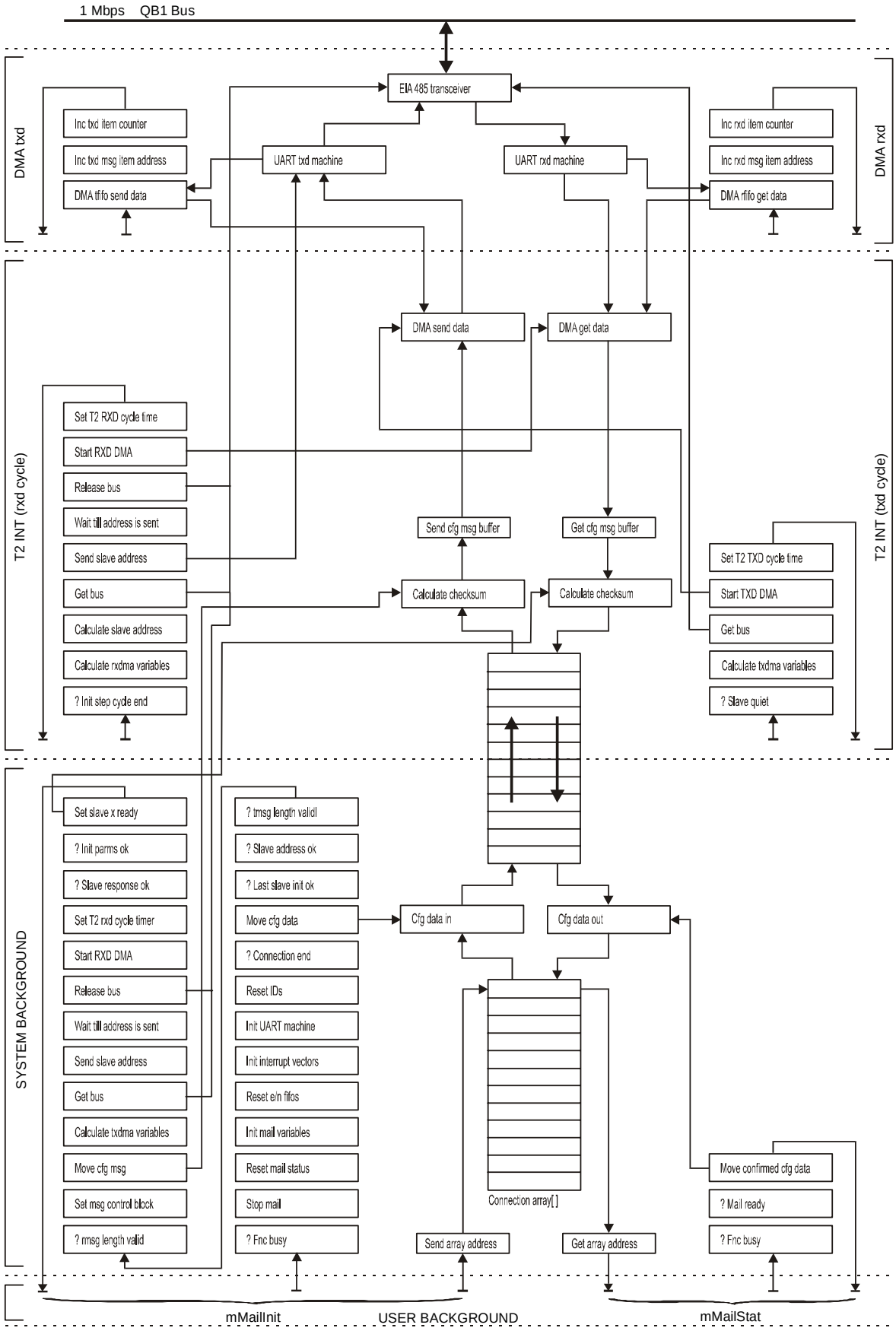


Fig. 4. Q1 bus 1 Mbps master mail – NMT architecture

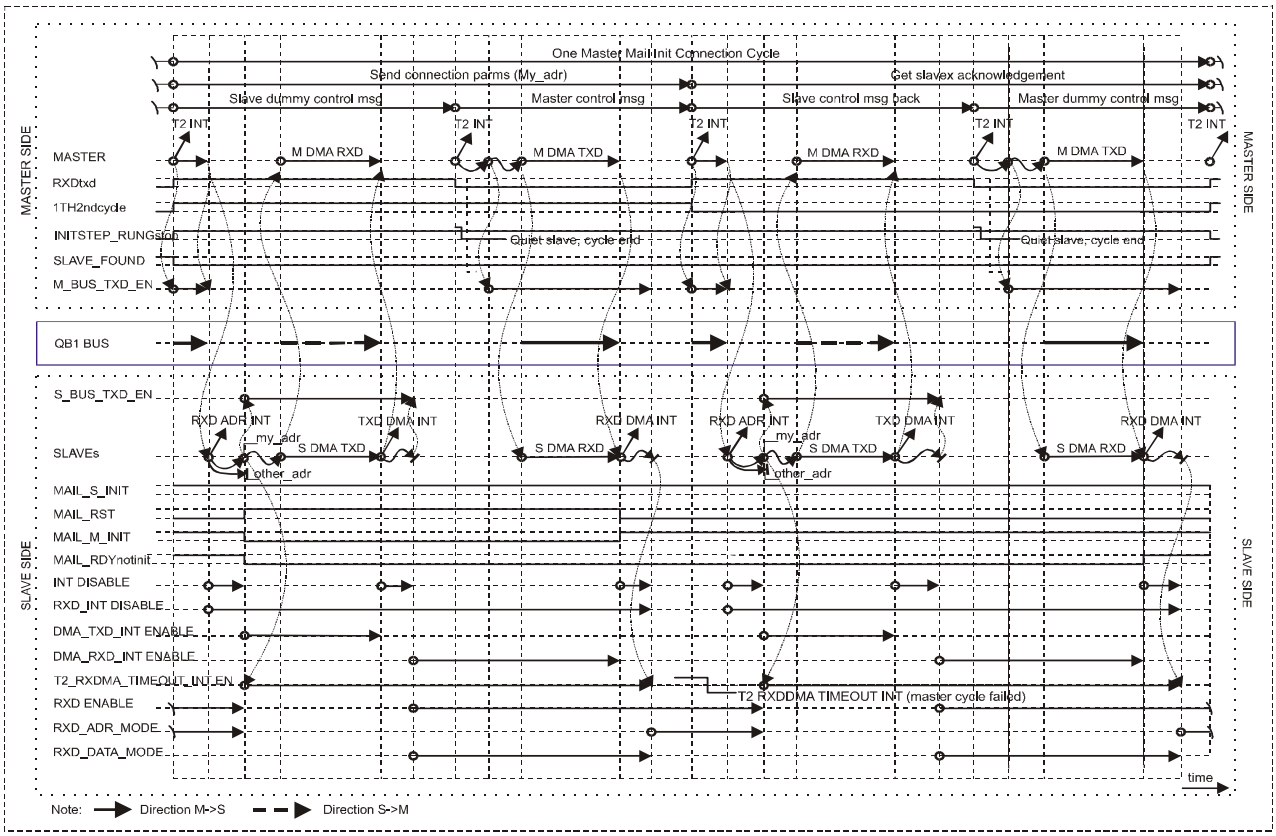


Fig. 5. Q1 bus 1 Mbps mail – NMT timing and master/slave cooperation

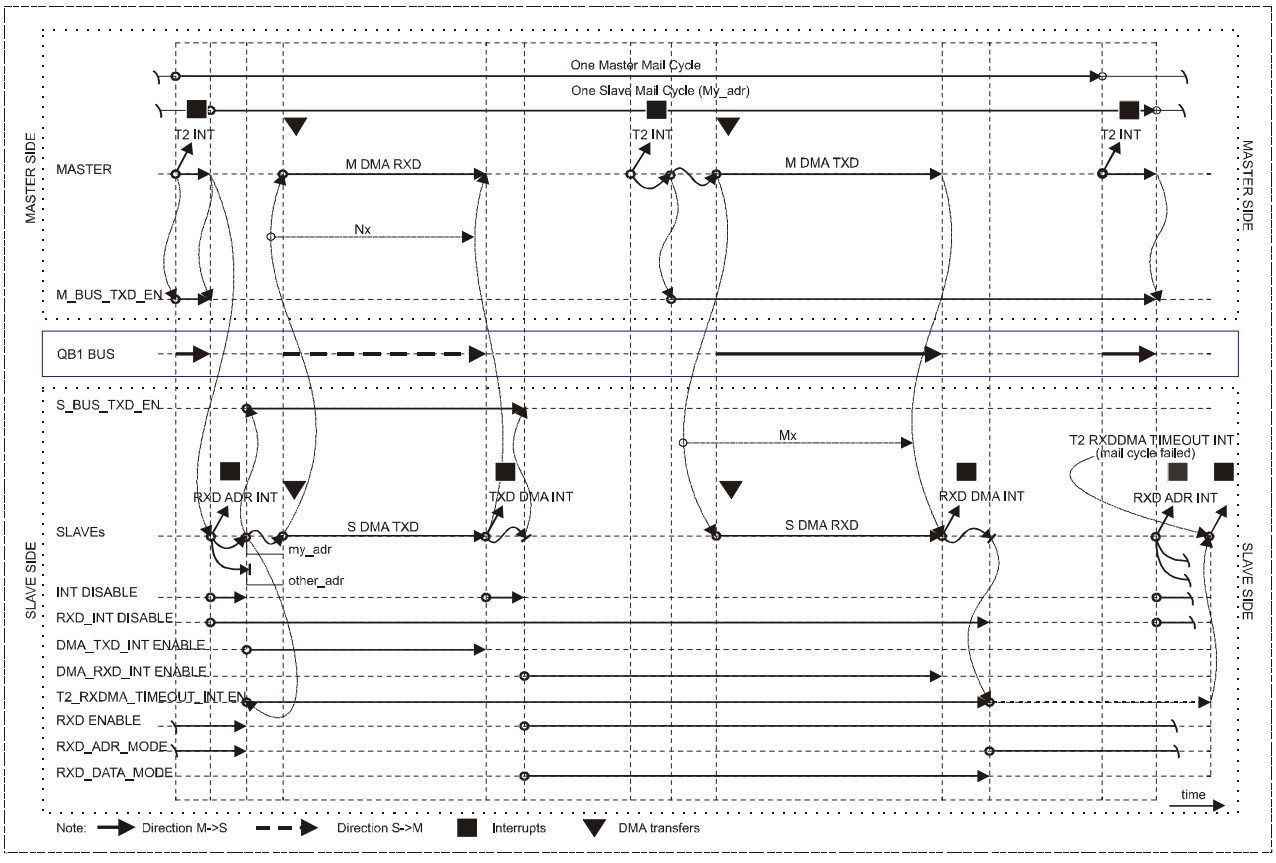
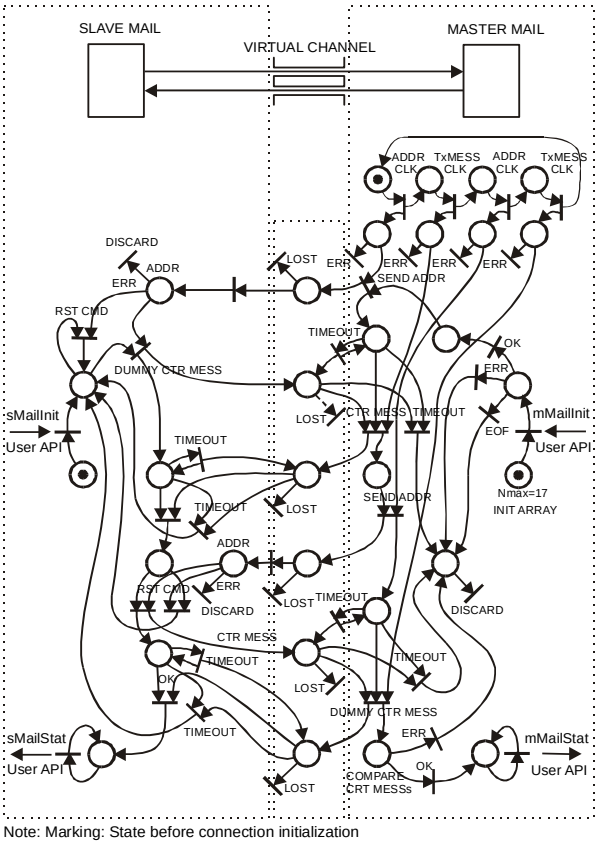


Fig. 6. Q1 bus 1 Mbps mail – PDT timing and master/slave cooperation



Note: Marking: State before connection initialization

Fig. 7. Q1 bus 1 Mbps mail – NMT protocol

4 Conclusion

The presented architecture of LAN communication services proved to be fully operable, efficient, user (programmer) friendly and reliable. Computer network is operated in 6 MW / 3 kV_{dc} electric locomotives SKODA 93E running over 200 khours with correct behavior and no apparent problems.

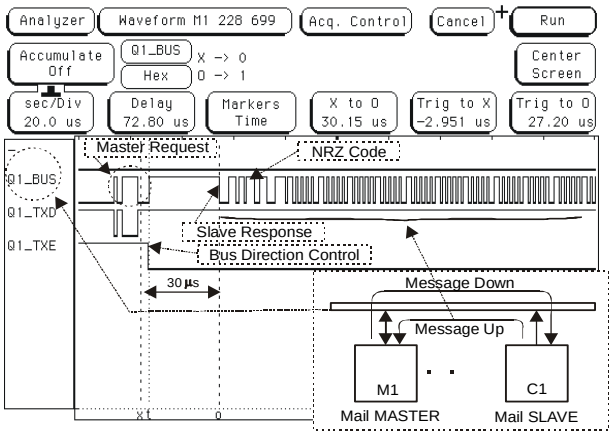


Fig. 8. Q1 bus 1 Mbps frame header slave response

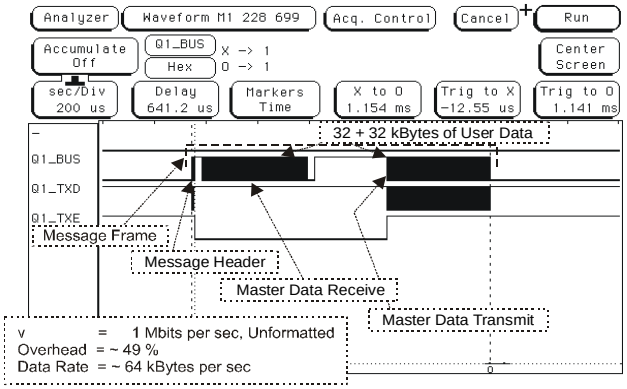


Fig. 9. Q1 bus 1 Mbps one PDT up/down message cycle

5 Acknowledgment

The research was partly supported by the RD prog. No.-MSM6849770015 of the CTU in Prague sponsored by the MEYS of the Czech Republic

References:

[1] A.M.K.Cheng, *Real-Time Systems: Scheduling, Analysis and Verification*, John Wiley, 2002. ISBN-0471184063.

[2] J.Zdenek, L.Koucky, P.Mnuk,, "Network Node for Temperature Control of Scientific Equipment for High Temperature Material Processing in Space Station". *Proc. of IFAC ws. PDS2003*, Jan. 2003, Ostrava, pp.291-294.

[3] H.Kopetz, *Real Time Systems: Design Principles for Distributed Embedded Applications*,Kluwer, 2003. ISBN-0792398947.

[4] J.Zdenek, L.Koucky, P.Mnuk, "Node for Drive Control of Scientific Equipment for High Temperature Material Processing in Space Station", *Proc. of Int. Conf. EDPE2003, Sept. 2003*, High Tatras, Slovak Rep., pp.201-206.

[5] J.Zdenek, "Control Electronics of Scientific Space Technological Facility", *Proc. of 11th Int. Conf. EPE-PEMC2004*, Sept. 2004, Riga, CD ROM.

[6] J.Zdenek, "Real Time Operating System in Distributed Control Computer of Electric Locomotive", *Proc. of XIII. Int. Symp. ISEM2005*, Sept.2005. Prague, pp.206-212.

[7] J.Zdenek, "Efficient DMA Based Local Computer Network Communication Services for Traction Vehicle", *Proc. of Int. Conf. SPRTS2005*, Oct. 2005, Bologna, pp.44-51.

[8] J.Zdenek, "System Design and Software Architecture of Traction Vehicle Control Computer", *Proc. of 12th Int. Conf. EPE-PEMC2006*, Aug. 2006, Portoroz, in print.