Scheduling Theory in Networked Measurenent – Control Systems Design

Emil MICHTA
Department of Electrical Metrology, University of Zielona Góra
65-246 Zielona Góra, Poland

ABSTRACT

Networked measurement – control systems are widely used in applications ranging from difficult process control to simple discrete manufacturing. Usually, this systems impose real-time requirements to the nodes and communication networks. In the paper current trends in measurement – control system architecture and use of scheduling theory to verify on designing stage deadline constrains are presented. At the end use of offline scheduling method to worst-case response time in one level system based on CAN network is shown.

Keywords: Measurement – Control Systems, Fieldbus Networks, Real Time Systems, Scheduling Analysis.

1. INTRODUCTION

Measurement – Control Systems (MCS) can be composed into four components: applications, communication system, measurement – control aparatus and envinroment (fig. 1).

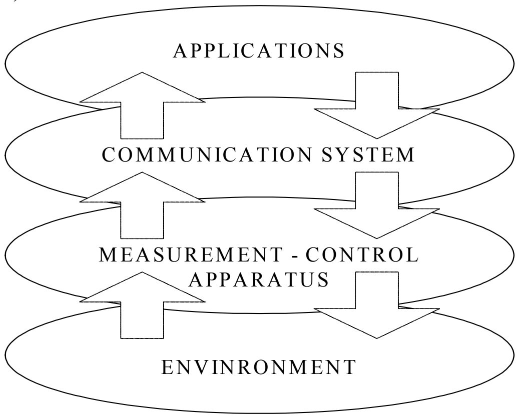

Fig. 1. MCS components.

Design of MCS means for given object (envinronment) selecting set of sensor and actuator nodes which are connected each other and/or connected with computer level running application program e.g. visualisation, configuration, dignostics, expert systems. In the past years three upper components changed dramatically [1,2,4,7,12]. It means that also design issuses are become new. Many of the systems we are designing have to meet soft or hard real-time requirements. Usually the economic cosequences of missing real time deadlines can be serious, leading to loss of life and/or property. A key requirement of a real-time system is to asses timing designing stage. To predict behaviour of a new system designer can use two approches: analytical methods based on scheduling theory for analysing worst-case behaviour of the system and discret-event simulation for determining average-case behavior of the system (fig. 2) [5,16]. Common way to find out the worst-case timing behavior of designed MCS is use of simulation method. Since bus traffic is non-deterministic, the number of test cases necessary for a complete testing is infinite. This method can give observed task response time R lower than worst-case response time R_{max} and deadline can be missed during exploitment. Using simulation methods allows estimate average response time. It can be useful for soft real-time systems requirements. Worst-case response time received by means of analytical methods is more pessimistic but for hard real-time systems that methods are prefered.

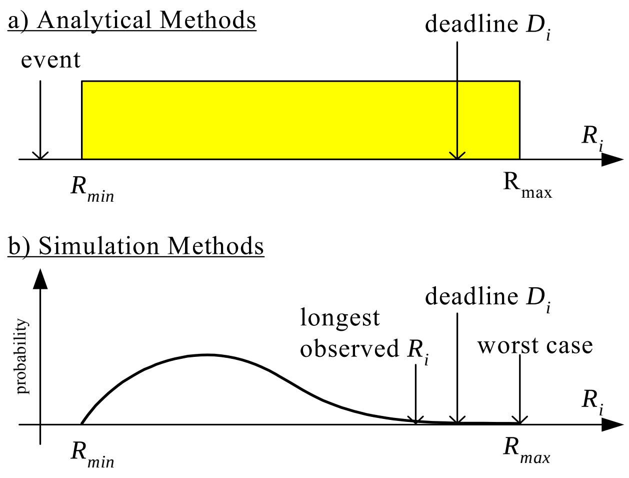

Fig. 2. Calculations of a response time for task *i*.

Use of scheduling theory in designing of networked MCS is very fruitful because temporal model applied for nodes can be used also to fieldbus level. The temporal models can be used with some modification to represent, analyse and track the latencies and communication delays associated with fieldbus elements [3,5,10,14,18].

2. MCS ARCHITECTURE

Past decade it is time of changes of MCS from point-to-point to network architecture (fig. 3). Determining network structure and optimizing the network bandwidth for different application requirements is now a basic design decission.

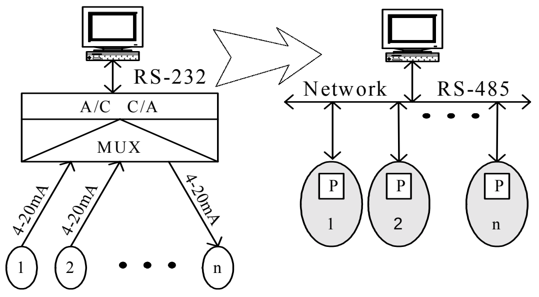

Fig. 3. Mux- to networked architecture migration.

Migration of an information technology, open systems, and intelligent sensor/actuators networks to measurement and control applications brings a new opportunity to increase data flow to management systems and to streamline operations by linking business management systems to the factory envinronment. The emergence of open systems on the factory floor and demand for more information at the management levels is moving factory automation to introduce new technology. In the past, company's information infrastructure provided little data on manufacturing and process operation to management, and what was available did not arrive in real time [4,12]. Boundaries were the result of proprietary control systems and data acquisition systems, factory information networks, and non-intelligent sensors. But as innovations from the computer and information technology have migrated to the factory floor, manufacturing standards have improved significantly and change architecture of industrial communication system. To meet the demands of the next generation of measurement and control system, the network will require a new architecture comprising following key dimensions: real time, migration from proprietary systems, resilience, management, performance and cost [1,3,10,13,14].

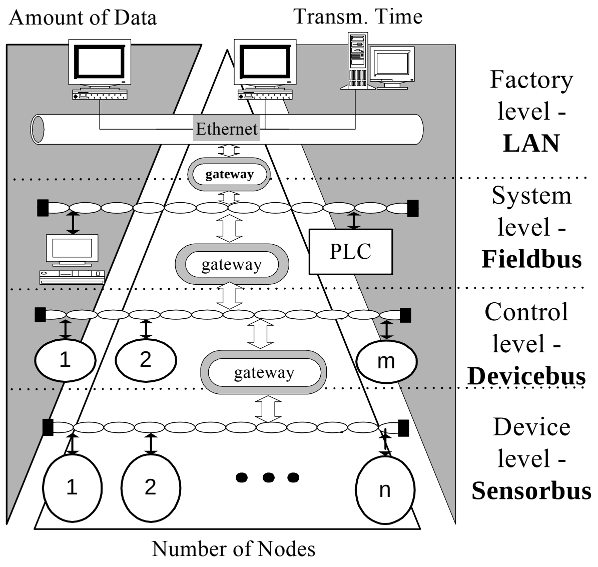

Fig. 4. NMCS multilayer architecture.

Network architecture covers the major components of a network and how they relate to one another. Network architecture defines state of actual implementation of a network but not specify the exact size and placement of components.
Model of communication architecture (fig. 4) describes levels and standards for all levels of the industrial automation and control environment, managing and handling information from sensors, transducers, instruments, and actuators, which intercommunicate with factory computer equipment. Industrial networks need to provide two views of the factory or process – a view of operations and a view of configuration, management and diagnostics data. With Internet, Extranet or Intranet access to the measuring-control network, manager being on the top levels is able to view the production level data and activity easily and cost-effectively [2,8].
The demand for open industrial communication systems is driven by end users wanted to move away from older, centralised factory control strategies to distributed control in the field. End users want enabling technology that provides true device interoperability, enhanced field-level control simplified maintenance and reduced installation costs. The only network architecture capable of delivering against this requirements will offer deterministic high performance, be standard based and non-vendor specific.
Most factory data collection applications use a batch approach, where data is transmitted at the end of the shift or other low usage times of the day. New networking technologies will change this model to real-time, with plant information being continually and automatically collected and analysed without any operator intervention. Factory operations will increase by directly connected in a client/server model to host computers and servers. Controllers, PLCs and EMS systems will be able to access any sensor, actuator connected to control and device network. The result will be better information on manufacturing processes.

3. SCHEDULING THEORY

Schedulability analysis can be performed online or offline (fig. 5). In online case (dynamic scheduling) schedulability of task set is analysed by node processor at run time. This approache is used if tasks parameters are not known prior to run time.

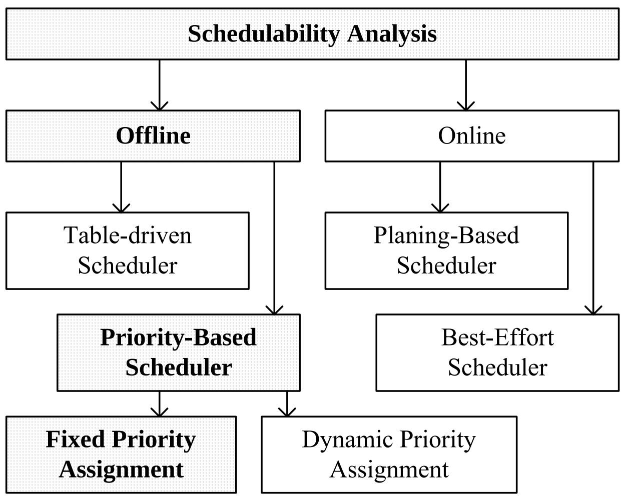

Fig. 5. Most important types of schedulability analysis.

There are two types of online schedulers [3,13,14,17]: planning-based and best effort. In offline case (static scheduling) schedulability of tasks set is performed befor system is run. In most cases of NMCS, number of tasks and task parameters are known before system is runing,

therefore in designing NMCS offline schedulability approaches can be effectivly used. This approach requires little processor run time overhead and schedulability of tasks is known on designing stage, befor system is runing. There are two types offline scheduling approaches: table driven scheduler and priority based scheduler. Table driven scheduler produce a schedule according to which tasks are dispatched at runing system. In the priority based approach no explicit schedule is created. At system runing time, tasks are executed on a highest priority basis. This approach is much more flexible than table driven.
Choice of scheduling approach depends on the system properties. Table driven dispatcher is simpler and requires less CPU resources than a priority one. Fixed priority scheduling is particularly suitable when there are external events that require a response deadline. In fixed priority the next runable process can start as soon as previous is completed, what improves resource utilization.
One of the most used priority assignment schemes is to give tasks a priority level based on its period: the shorter period, the higher priority. This assignment is explained by the fact that more critical devices will provide inputs more frequently, or indeed will be polled more frequently. Thus, if they have shorter periods, the worst-case response time should also be shorter. This type of priority assignment is known as the rate monotonic (RM) priority assignment [3,13,14,17].
If some of the tasks are sporadic, it may not be reasonable to consider its relative deadline equal to the period. A different priority assignment can be to give the tasks a priority level based on its deadline: the shorter the relative deadline, the higher the priority. This type of priority assignment is known as the deadline monotonic (DM) priority assignment [3,13,14,17].
In both RM and DM priority assignments, priorities are fixed (static), in the sense that they do not vary along time. At run-time, tasks are dispatched the highest-priority-first. A similar dispatching policy can be used if the task that is chosen to run is the one with the earliest deadline. This corresponds to a priority-driven scheduling where priorities of the tasks vary along time, hence corresponding to a dynamic priority assignment. This type of dynamic priority assignment is known as earliest deadline first (EDF) priority assignment [3,13,14,17].
In all three cases, the dispatching will take place when either a new task is released or the execution of the running task ends. This however may not stand in a non pre-emptive context. With priority-based scheduling, a higher-priority task may be released during the execution of a lower priority one. In a pre-emptive system, the higher-priority task will pre-empt the lower-priority one. Contrarily, in non-preemptive systems, the lower-priority task will be allowed to complete its execution before the other executes.
To analyse a time constrains of a networked, multilevel, distributed system, we may use a combination of a different above-mentioned scheduling policies (fig. 6). For example, fieldbus layer (the bus) may be priority driven with use non-preemptive RM or DM polices while nodes (processors) may be priority or time driven with use both preemptive and non preemptive approaches. Different nodes (processors) may have different scheduling policies due to manufacturer preferences thus during the NMCS designing phase we have real influence mainly on choice of a communication system.

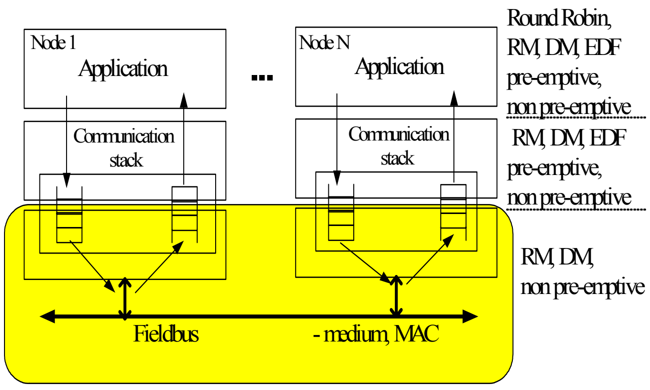

Fig. 6. Different scheduling policeses on one level NMCS model.

Most NMCS models assumes that the number of tasks is constant and that either static priorities or static cyclic scheduling is used. Results from scheduling theory, regarding fixed priority scheduling, can be used to evaluate different NMCS designs with application on different communication protocols.

4. FIXED-PRIORITY ASSIGNMENT

For the RM priority assignment, an utilisation-based pre-run-time schedulability test, which if satisfied, guarantees that all n tasks will meet their deadlines is following [3,5,16]:

$$\sum_{i=1}^{N} \frac{C_i}{T_i} \leq N \times \left(2^{1/N} - 1\right) \tag{1}$$

where: C_i is the worst-case computation time of task i, and T_i is the minimum time between task i releases (period).
Utilisation-based tests are sufficient but not necessary conditions. This utilisation-based test is valid for periodic independent tasks with relative deadlines equal to the period and for preemptive systems. For the non pre-emptive case, a similar analysis can be adapted to include task i blocking time B_i, during which high priority tasks are blocked by low priority tasks [3,5,16]:

$$\sum_{i=1}^{i} \left(\frac{C_i}{T_i}\right) + \frac{B_i}{T_i} \leq i \times \left(2^{1/i} - 1\right), \forall_{i,1 \leq i \leq N} \tag{2}$$

It was proved [23] that the worst-case response time R_i, of i-task is found when all tasks are synchronously released at their maximum rate (critical instant). For that instans R_i can be computed by the following recursive equation:

$$R_i^{n+1} = \sum_{j \in hp(i)} \left(\left\lceil \frac{R_i^n}{T_j} \right\rceil \times C_j \right) + C_i, \tag{3}$$

where $hp(i)$ denotes the set of tasks of higher priority than task i priority.
Initial value for R_i is zero. The recursion ends when $R^{n+1} = R^n = R_i$ If worst-case response time R_i exceeds T_t (in the case of RM priority assignment) or D_i, (in the case of DM priority assignment) the task i, is not schedulable. This result is valid for the preemptive context.

Taking into account blocking factor B_i worst-case response time for non-preemptive approach can be derived from following recursive equation:

$$R_i^{n+1} = B_i + \sum_{j \in hp(i)} \left(\left\lceil \frac{R_i^n}{T_j} \right\rceil * C_j \right) + C_i \tag{4}$$

where: $B_i = \max \{C_j\}$ for *lp(i)*, where *lp(i)* denotes tasks with lower priority than task *i*.

5. TASK MODEL

Communication level of NMCS connects nodes using a network. The nodes co-operate throught the network to provide the end-to-end functionality. In real-time system this end-to-end functionality from event to response, must be provided within specified deadline. Response time R_i on event *i* should be lower than deadline D_i. Timing analysis is applied to a resource shared between multiple activities. According to NMCS model shown on fig. 6, on an application level, in sensor and actuator node several functions are executed on single microprocessor. On a communication stack level output and input messages waits in queue to be served and finlly on a network layer one bus carries a number of messages sent by nodes.

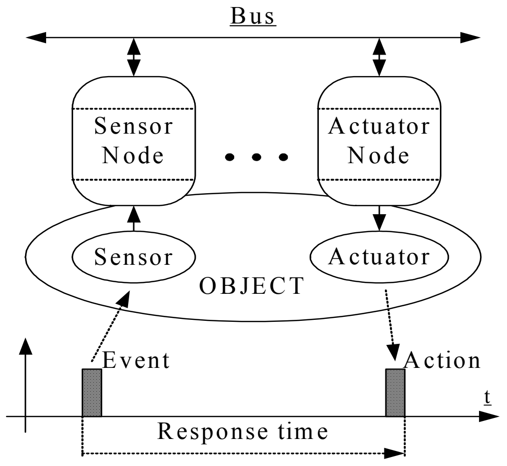

Fig. 7. Sensor-actuator response time.

To simplify timing analysis of a sensor node to actuator node communication we can partition analysis into components that can be analysed inependently (fig. 8). For task *i* release jitter time J_i, blocking time B_i and interference time I_i are joined together and represented by the gray area (waiting time) to the left in the execution time windows on fig. 8. End to end response time depends on the response time of each component involved in meeting deadline. Different offline scheduling policies could be investigated using this simple control loop as an example. If preemption is used, the blocking time corresponds to the longest duration of a low priority code sequence that must run in mutual exclusion of the task. Without preemption, blocking is the duration of the longest lower priority task. The interference time equals the total execution time of higher priority tasks that may appear during the maximum response time.

On fig. 8 are shown execution times of tasks on the two nodes (sensor and actuator) and the communication task execution time on the bus. An execution window represents each activity. The left end of the box represents task arrival and the right end represents the latest task completion. The length of the box represents the task response time on given level. The activity may be finished at any time after the minimal execution time represented by the length of the white box. In the worst case, release jitter and blocking and interference from other tasks may delay task execution. Note that for preemption approach execution and interference may be interleaved but this is not shown in the diagrams on fig. 8.

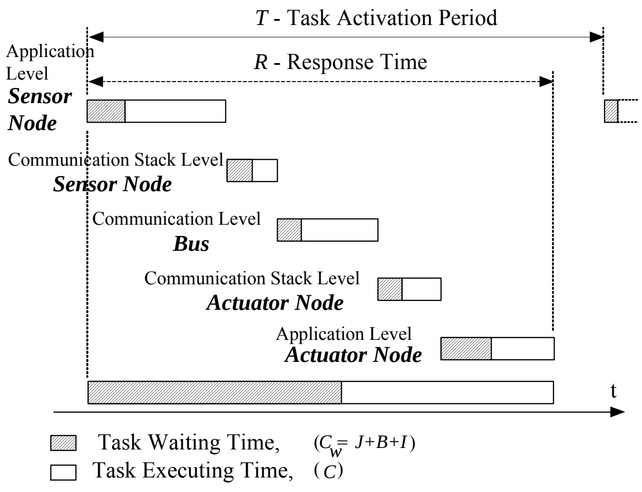

Fig. 8. Sensor-actuactor periodic action path.

The task execution time *C* on each level is different and depends on many causes. On processsor node level execution time *C* depends on processor its clock and task complexity. On the fieldbus level execution time depends on transmission rate, data and control field length and type of media access control. For a given node and given fieldbus ecexution time can be treat as a node and communication layer parameters. On fig. 9 there are presented execution time exapmles for five popular fieldbus protocols for two transmission rates 500kbps and maximal rate for each one. The task is to send two data bytes over fieldbus. Every part of the process control and automation industry, from embedded systems to the factory level has recognised the importance of Ethernet and TCP/IP. Ethernet has become the dominant network technology at the controller supervisory level. Every controller, PLC and DCS vendor has an Ethernet interface and it is now moving downward device and I/O level [2,8,12]. For Ethernet minimum data length field is 48 bytes and transmission rate is 10Mbps. For Profibus, frame schema with floating lenght data field is used. Execution time for given protocols was received from following equations:

$$C_{can} = \frac{1}{V_{tr}} \left(\frac{34 + 8 * L_{dane}}{5} + (L_{ster} + L_{dane}) * 8 \right), \tag{5}$$

$$C_{profi} = \frac{1}{V_{tr}} \left[2 * T_{syn} + \left(L_{ster}^{ini} + L_{ster}^{odp} + L_{dane}^{ini} + L_{dane}^{odp} \right) * 11 \right], \tag{6}$$

$$C_{ibus} = \frac{1}{V_{tr}} \left(13 * (L_{ster} + N * L_{dane}) \right), \tag{7}$$

$$C_{Eth} = \frac{1}{V_{tr}}\left((L_{ster} + L_{dane})*8\right), \quad (8)$$

$$C_{Lon} = \frac{1}{V_{tr}}\left(T_{syn} + (L_{ster} + L_{dane})*8\right). \quad (9)$$

where: L_{dane} – data field lenght, L_{ster} – control field lenght, T_{syn} – synchronisation time (interframe gap).
Execution times shown on fig. 9 for mentioned protocols differ significantly. It means that choice of protocol and its transmission rate is the key decission on the NMSC communication level design phase.

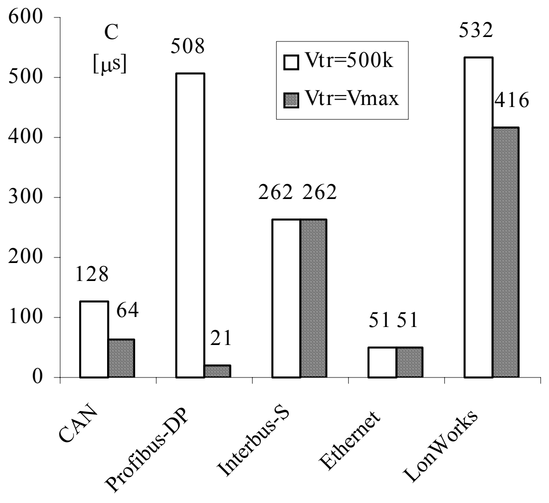

Fig. 9. An execution time of a communication level task.

5. EXAMPLE

The increasing use of communication networks in time critical applications presents designers with fundamental problems with the determination of worst-case response times of communicating distributed nodes. Experiences with single processor scheduling analysis has shown that models, which abstract away from implementation details, are at best very pessimistic and at worst lead to unschedulable system being in fact schedulable [5,8].
Controller Area Network (CAN) is a well-designed, peer-to-peer communications bus for sending and receiving short real-time control mesagess of up to 1 Mbit/sec [6,16,19]. One of the drawbacks to CAN is the inability to bind accurately the worse case response time of a given task. This example shows simple four CAN nodes network, each with the independent periodic task with given parameters of a communication and processor level: task period time *T*, task execution time *C* and task deadline time *D*. Each node and network typically need to meet a number of deadlines (e.g. network carries many unrelated messages each with their own deadlines), and these per-component scheduling problems are decoupled by the its deadline. The transmission of frames in CAN is similar to Ethernet protocol, except that the collision avoidance is different and there is no address field in the frame. Each node with a frame to send listens for bus idle time. If this is seen, each node starts sending its frame. Due to a dominant/recessive bit approach on the bus, the bits collide and give a new pattern. This pattern is exploited to form the arbitration scheme.
Each frame starts with an identifier that uniquely marks the frame. In our example task 1 has id=5, task 2 id=6, task 3 id=25 and last one id=42. This identifier is transmitted most-significant bit first. If a node sends a '1' but reads back a '0' then it backs off until the next bus idle time period. As the protocol runs out the frame identifier, nodes will back off, until at the end only one node is still arbitrating. This node goes on to transmit its frame. It is important that no two nodes transmit frames with the same identifier because otherwise the arbitration will not work. The effect of the arbitration scheme is to assign a priority to each frame, being the value of the identifier, where a smaller number is *a* higher priority. This priority based scheduling of the network is just what is needed to apply offline schedulability analysis.

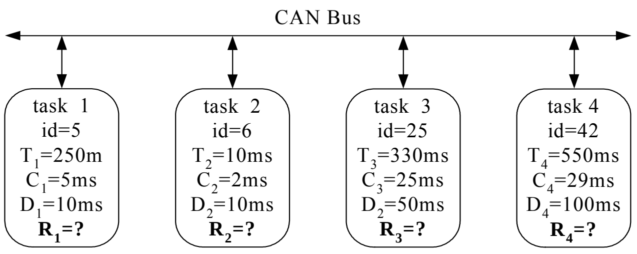

Fig. 10. CAN-based NMCS - response time calculation.

Even for given on fig. 10 four nodes time parameters is it very hard to assess worst-case response time without use of analytical or simulation methods.
Results of analysis and prediction of the worst-case response timing behavior of the NMCS based on CAN bus and one processor nodes using non-preemptive RM and DM approach for communication level and with preemptive RM and DM approach for nodes level are shown in tables 1-4.

Table 1. Pre-emptive **DM** (node)

	Task 1	Task 2	Task 3	Task 4
Ti	250	10	330	550
Ci	5	2	25	29
Di	10	10	50	100
Ri	5	7	38	75

Table 2. Non pre-emptive **DM** (bus)

	Task 1	Task 2	Task 3	Task 4
Ti	250	10	330	550
Ci	5	2	25	29
Di	**10**	**10**	**50**	100
Ri	**34**	**36**	**54**	54

Table 3. Pre-emptive **RM** (node)

	Task 1	Task 2	Task 3	Task 4
Ti	250	10	330	550
Ci	5	2	25	29
Di	10	10	50	100
Ri	7	2	38	75

Table 4. Non pre-emptive **RM** (bus)

	Task 1	Task 2	Task 3	Task 4
Ti	250	10	330	550
Ci	5	2	25	29
Di	**10**	**10**	**50**	100
Ri	**36**	**31**	**54**	54

Analysis was carried out by use of a recurrence equqtions (2) and (3) for non-preeemption and preemption approach. The analysis was done for each task and was completed when recurrence equation have converged to consistent response time. As we can see task with the same parameters on node (processor) and fieldbus level are scheduled only on node level, which use preemption approach in access to resources and can misses its deadlines on CAN bus level for three tasks. Only for task 4 time requirements are meet. Time requirements for task 3 are almost meet and for soft real-time systems it could be acceptable. Task 1 and task 2 both for RM and DM offline scheduling approach ecxeed its deadline significantly. To reach schedulability on the CAN bus level we should increase transmition rate to shorten execution time *C*. If it is impossible then designed NMCS topology should be redesign or new protocol could be taken into account.

8. CONCLUSIONS

In high volume objects and manufacturing lines there is installing new generation of networked MCS with soft or hard real-times constrains. Scheduling theory is an effective approach that can be used during all phases of multilevel NMCS design, development and implementation to guarantee system predictability and performance. Scheduling theory help us to appropriate mapping of system functionality to available resources, nodes with processors and networks with different protocols, in such a way that system timing constrains are meet.

9. REFERENCES

[1] Alippi C., Ferrari S., Piuri V., Sami M., Scotti F.: New Trends in Intelligent System Design for Embeded and Measurement Application. IEEE I&M, June/99, pp. 36-44.

[2] Automation Research Corporation: Device &Field Network Global Outlook. Market Studies, 1999.

[3] Audsley N., Burns A., Richardson M., Tindell K., Wellings A.: Applaying New Scheduling Theory to Static Priority Pre-emptive Scheduling. Software Engineering Journal, Vol. 8, No. 5, pp 285-292.

[4] Boroń W.: Distributed Control Systems. Pomiary Automatyka Kontrola, 6/1998, pp. 203-206. (in polish)

[5] Burns A.: Scheduling Hard Real-Time Systems. Software Engineering Jurnal, Special Issue on Real-Time Systems, May 1991, pp. 116-128.

[6] CAN Product Guide. CiA, Erlangen, 1997.

[7] Finkelstein L., Finkelstein A.: Design Theory and Measurement Science. XVI IMECO World Congres, Viena, 2000.

[8] Hewlett-Packard. Industrial Ethernet. 1998.

[9] Jordan J. R.: Serial Networked Field Instrumentation. Wiley Series in Measurement Science and Technology. 1995.

[10] Joseph M., Pandaya P.: Finding Response Time in Real-Time System. The Computer Journal, 1996, Vol. 29, No. 5, pp. 390-395.

[11] Michta E., Markowski A: Real time analysis in distributed measurement – control systems. III Conference Measurement Systems in Research and Industry. Zielona Góra 2000, pp. 159-168. (in polish)

[12] Michta E.: Architecture of an Industrial Computer Networks. Polish – German Symposium Sience Research Education. Zielona Góra, 2000, pp. 161-166.

[13] Ripoll I., Crespo A., Mok A.K.: Improvement in Feasibility Testing for Real-Time Tasks. Real-Time Systems Journal, Vol. 11, No. 1, July 1996, pp. 19-40.

[14] Stancovic J.: Real-Time Computing Systems: the Next Generation. Hard Real-Time Systems. IEEE Computer Society Press, 1988, pp. 14-38.

[15] Sydenham P., Thorn R.: Handbook of Measurement Science. Vol. 3, New York, J. Wiley & Sons, 1992.

[16] Tindell K., Burns A., Wellings A.: Calculating CAN Message Response Time. Control Engineering Practice, 1995, Vol. 3, No. 8, pp. 1163-1169.

[17] Tovar E., Vasques F.: Guaranteeing Real-Time Message Deadlines in PROFIBUS Networks. Proceeidings of the 10th Euromicro Workshop on Real-Time Systems, pp. 79-86.

[18] Werewka J., Żaba S.: Message Scheduling in Distrubuted Real Time System Based on Fieldbus. Electronic and Telecommunication, 1999, pp. 25-50. (in polish).

[19] Zuberi K., Shin K.: Scheduling of Messages on CAN for Real- CIM Applications. IEEE Transaction on Robotics and Automation, 1997, pp. 310–314.