Fire Safety Alarm Transmission in Networked Building Automation Systems

Georg Neugschwandtner, Wolfgang Kastner, Bernhard Erb
Automation Systems Group, TU Vienna – {gn, k, berb}@auto.tuwien.ac.at

Abstract

Comparing the structure of a modern fire alarm system with the one of a networked Building Automation System (BAS) reveals important common characteristics. Both systems have nodes distributed in the building, communicating over an electric wire. They typically use the same cable for both communication and power supply of nodes, and both can be implemented using a similar network structure. This paper presents the work-in-progress regarding the implementation of safety-related functionality using the infrastructure of a networked BAS. It shows how the link layers of KNX/EIB and LonWorks can be adapted in a downward compatible way to ensure that safety related messages have priority over any other BAS traffic. In addition, a system-neutral protocol for the robust and timely transmission of fire alarm messages is proposed.

1 Introduction

Building Automation Systems (BAS) are concerned with the automatic control of building services, key areas being Heating, Ventilation and Air Conditioning (HVAC), lighting and shading. In common practice, BAS do not provide safety related services. The most relevant safety application in buildings concerns fire detection and alarms. These systems are typically independent and stand-alone. Data exchange (e.g., to realise fume extraction in case of fire) is done via gateways. However, only shared use of resources allows to take advantage of synergies. This concerns both hardware, such as wires, sensors, actuators and controllers, and software, like tools for configuration and management. For this reason, a *tight integration* of such systems into the very fabric of networked BAS is desirable. But safety related systems, in particular those concerned with life safety, impose special requirements on the underlying communication system. Thus, it has to be ensured that these requirements are met.

For fire alarm systems, they are laid down in formal standards. Therefore, key points of relevant standards are outlined in Section 2. Existing alarm system implementations are not available for comparative analysis as relevant information is not publicly available in sufficient detail.

Next, we show how an integration of fire alarm systems into popular networked BAS (i.e., KNX/EIB and LonWorks) can be achieved. As a first step, it is shown how safe communication can share the BAS network stack up to the link layer. To be able to consider safety traffic as independent from BAS traffic as possible, safety messages are given absolute priority. This is done by adapting the medium access control protocols of EIB/KNX TP1 (Twisted Pair 1) and LonTalk (e.g., applicable to the FT-10 free topology channel) by design extension and/or appropriate configuration. Further, a system independent, robust transport layer protocol is introduced. It provides additional fault detection and correction capability, which we believe to be useful in this context.

The present paper is independent of the EU collective research project "SafetyLon" which was established in 2005. The objective of this project is to provide a safety extension to LonWorks. No publications appear to have been made to date, however.

2 Standards for fire alarm systems

A fire alarm system (FAS) as defined in [2] is a system with a single central fire alarm monitoring station (CFAMS) and a number of fire detectors or manual fire indicators communicating over transmission paths. In the following, both automatic detectors and manual indicators shall be cumulatively referred to as *sensors*. A CFAMS and its associated sensors communicate in a master slave relationship. The CFAMS acts as master, supervising the communication and responding to significant state changes with appropriate actions. It has to be guaranteed that no more than 32 fire detectors are affected by a single short circuit or cable break.

Interconnection patterns between FAS are specified in [1]. In this standard, a *networked FAS* is defined as a number of FAS interconnected in a peer-to-peer manner without a central point of control. In this case, every participating CFAMS is assigned the task of communication surveillance. In contrast, a *hierarchical FAS* combines multiple FAS under the control of a master CFAMS. This master CFAMS is responsible for the communication between all participating CFAMS. Every FAS however remains self-contained regarding fire detection and alarm signalisation.

FAS as defined in [3] are allowed to share transmission paths and devices with other systems. However, the absence of feedback must be guaranteed. This implies that on a shared transmission path messages from the FAS are assigned the highest priority. This way messages from

1-4244-0379-0/06/$20.00 ©2006 IEEE.

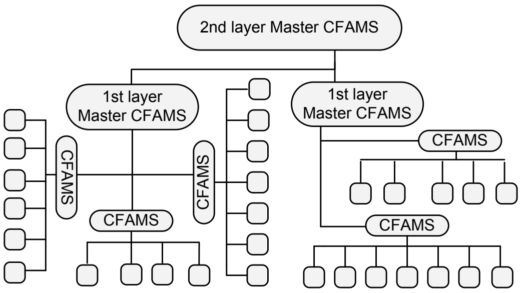

Figure 1. Topological model

other systems are not able to affect the FAS communication. Additionally, fire alarm messages have to be assigned the highest priority within the FAS. This implies that failure messages, for example, must be assigned a lower priority than fire alarm messages.

Requirements on the response times of fire alarm systems are again defined in [1]. Any fire alarm must be signalised within 10 seconds at the local CFAMS; within 20 seconds at any other CFAMS in a networked FAS; or equally within 20 seconds at the master CFAMS in a hierarchical FAS. Failures (e.g., sensor failure) must be reported within 100 seconds locally and within 120 seconds remotely.

3 General model

The hierarchical FAS pattern appears most appropriate as the basis for a general model of implementing FAS on networked BAS. In our model, every FAS is associated with a single KNX/EIB line (or LonTalk subnet). As a first approach, only the lowest hierarchy layer contains sensors. For simplicity, master CFAMS just collect data from multiple slave CFAMS, but do not have sensors attached on their line/subnet. CFAMS functionality may, but need not be implemented by KNX/EIB or LonTalk routers.

Second, since at most 32 sensors must be affected by a transmission channel failure, the number of sensors in every subline/subnet shall be limited to 32. This allows maintaining the free topology BAS engineers and installers are used to.

In case of a short circuit or a permanently faulty node blocking all communication on the subline/subnet, all 32 sensors will fail. The fault however remains contained to one single CFAMS. Although this will cause the entire loss of automatic fire detection for a certain part (e.g., one floor) of the building, the situation is not different with traditional FAS. Such a failure has to be detected and signalled in due time, however. If it happens on a higher hierarchy level, the CFAMS remain operational.Only central monitoring and alarm propagation will be affected.

KNX/EIB allows 225 lines at the lowest hierarchy layer. LonTalk allows 255 subnets per domain. This translates to a maximum of 7200 and 8160 sensors, respectively. This appears sufficient even for large installations. In addition, the remaining address space is still open for legacy nodes.

Finally, it shall be defined that FAS devices (sensors as well as CFAMS) can be in one of the following states:

- Ready: The device is ready to detect and report a fire alarm condition.
- Alarm: The device has detected a fire.
- Failure: The device has detected an internal error (communication error, physical sensor failure, . . .).
- Init: The device is establishing communication with its associated CFAMS (or sensors, respectively).

The general idea is that the master collects status information from the slaves. It then evaluates this information and decides upon the system state, taking appropriate measures. Cable breaks and silent sensor failures have to be detected automatically.

4 Adapting the KNX/EIB Link Layer

Medium access on EIB/KNX TP1 is controlled using carrier sense multiple access (CSMA) with bit-wise arbitration on message priority and device address. Four priority levels (*low, normal, high, and system*) are provided. Within these levels, messages repeated due to a previously failed transmission are further prioritised.

In addition to bitwise arbitration, the required idle time between messages is also modified. Non-repeated messages of high or low priority (i.e., those belonging to the "standard" priority class as shown in Fig. 2) are subject to an additional waiting period of 3 bit times in addition to the minimum of 50 bit times.

The highest priority level (*system*) is used by the transport layer for the exchange of control messages during reliable connections. This is an unsuitable basis for the integration of FAS, since the priority of safety-related traffic cannot be guaranteed. Therefore, the medium access control (MAC) mechanism needs to be extended to provide an additional priority class. For this reason, we allow to start the transmission of safety messages already after 47 bit times. "High" (i.e., messages with high or system priority, and any repeated message) and "standard" priority classes follow as usual in standard KNX/EIB.

To ensure that legacy devices which are unaware of this extension lose arbitration, safety nodes starting a transmission shall fill the safety and high priority class time windows with dominant states.

Within the safety related communication, the priority of alarm messages over others (e.g., failure notifications or connection establishment) must be ensured. Also, sensors need to obtain relative priorities for bus access. This can again be provided by bitwise arbitration. Also, routers need to understand the priority extension.

5 Integration in LonWorks

For the TP medium, LonTalk uses a CSMA variant. At the start of the arbitration phase, before the contention period, priority time slots are available for urgent messages. 128 different priorities are available for each subnet. The

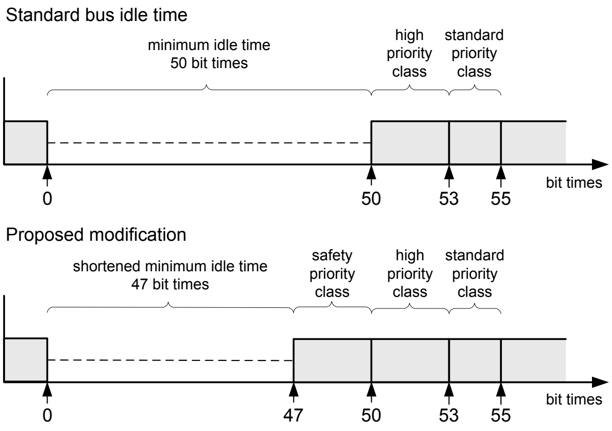

Figure 2. EIB/KNX TP1 MAC adaptation

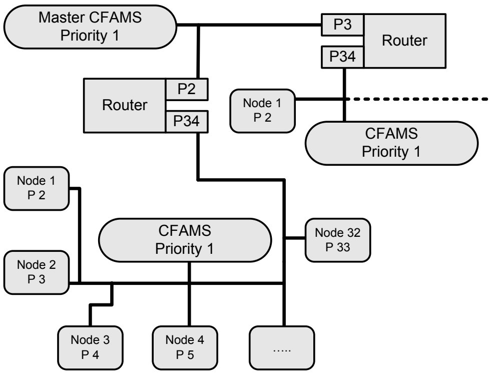

Figure 3. Priority assignment in LonTalk

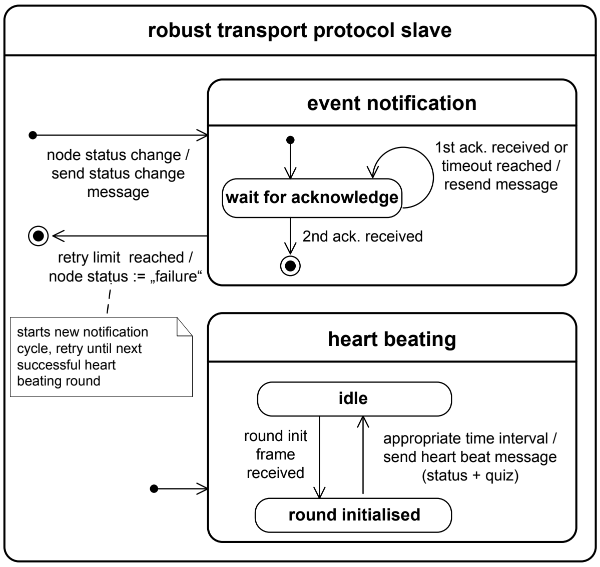

Figure 4. Slave protocol

actual number of priority slots (i.e., available levels) is determined in the engineering phase.

In contrast to KNX/EIB, priority levels are assigned to nodes instead of messages. This mechanism can be used for the unconditional prioritization of safety nodes over others in a straightforward manner. Within a subnet at the lowest hierarchy layer, the highest priority is assigned to the CFAMS. The 32 subsequent priority levels are assigned to the sensors.

Special attention must be directed to routing. Every LonTalk router in LonWorks is associated two different priority levels, one for each transmission path. Every message contains a (Boolean) priority flag. When a message with this flag set passes a router, it is transmitted with the priority level of the outgoing interface.

In a hierarchical FAS implemented on LonTalk, the hierarchy-upward interface of the router needs to be assigned one of the 32 highest priority levels. The master CFAMS is assigned the highest priority in the higher-ranking subnet. This pattern corresponds to the assignment of priority levels to sensors and (slave) CFAMS on the lowest hierarchy layer.

The hierarchy-downward interface of the router is assigned the lowest priority of safety nodes within a subnet. In that way it is guaranteed that the propagation of alarm states from the sensors to the CFAMS (or from slave to master CFAMS, respectively) takes prevalence over the downward propagation of state information from other fire alarm systems. A configuration example is given in Fig. 3.

As a first important restriction of this approach, the use of prioritized messages is allowed for safety-related traffic only. Otherwise, hierarchy-upward prioritised traffic would be able to starve the communication between slave and master CFAMS on the higher-ranking subnet. As another, assigning relative priorities within safety-related traffic is not possible using this approach. For example, a sensor constantly reporting a non-critical error condition could starve fire alarm messages from a sensor with a relatively lower priority level.

These problems could be addressed by assigning multiple node addresses to every safety node, which then would allow assigning separate priorities to different types of messages. Also, routers could be extended to modify the priority of outgoing messages depending on their priority level on the incoming side.

6 Robust protocol

The protocol presented in the following is designed for the exchange of status information between sensors and CFAMS. It can be implemented on top of the adapted link layers presented in the previous two sections. However, the sole requirement on the underlying channel is that one node can be assigned a higher priority for transmission than any other. Otherwise, it is system independent. It also makes no assumptions about the reliability of the underlying channel. It can deal with transient communication errors, silent sensor failures and cable breaks.

The protocol is inspired by the *“two message dependency algorithm”* found in [2]. It basically rests upon an event driven approach enriched by a heartbeating feature. The general idea is to enhance the error detection capability by transmitting every status change event twice.

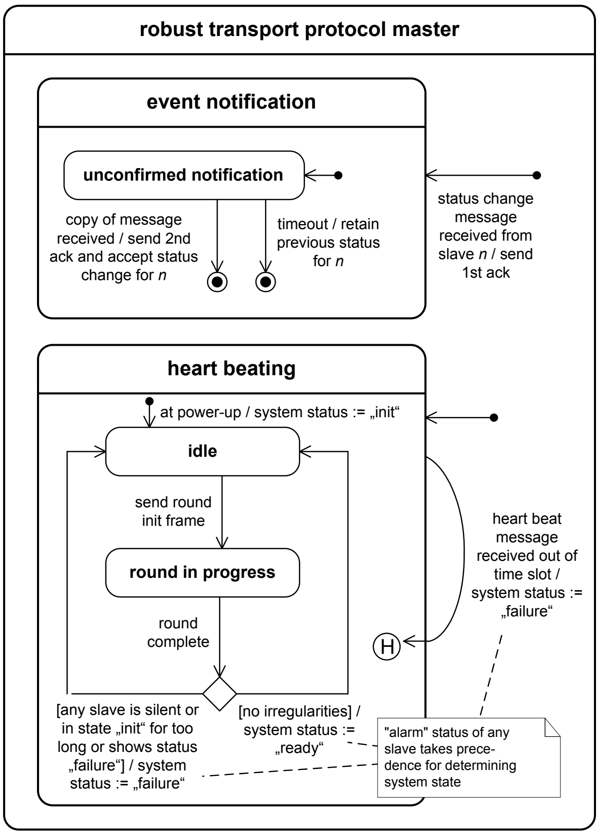
robust transport protocol master
event notification
unconfirmed notification
copy of message received / send 2nd ack and accept status change for *n*
timeout / retain previous status for *n*
status change message received from slave *n* / send 1st ack
heart beating
at power-up / system status := „init“
idle
send round init frame
round in progress
round complete
heart beat message received out of time slot / system status := „failure“
H
[any slave is silent or in state „init“ for too long or shows status „failure“] / system status := „failure“
[no irregularities] / system status := „ready“
"alarm" status of any slave takes precedence for determining system state

Figure 5. Master protocol

The master only accepts a state change indication if corresponding messages are successfully delivered.

Heartbeating is included to allow a periodic liveness check of all slaves, adding a time-triggered aspect. Since the propagation of slave status changes within an adequate amount of time is ensured by the event-driven part of the protocol, the heartbeat interval can be left at the maximum delay allowable to leave the silent failure of a slave go unnoticed. This reduces the bandwidth consumption significantly in comparison to a pure time-triggered approach.

Details of the slave and master protocols are shown in Fig. 4 and 5, respectively. Messages are uniquely assigned to a particular notification cycle by including a sequence counter or toggle bit. This allows the master to accept a status change after any two repetitions of the same slave message, regardless of whether they were actually sent in sequence by the slave. Such a situation can occur when acknowledgements get lost. On the other hand, the master needs to send two distinctive acknowledgements, since they trigger different activities in the slave. The first merely prompts the slave to send its message again, while the second one signals that the state change was accepted and no further repetitions are necessary.

Until a slave receives this second acknowledgement, it continuously retransmits its status change report with a fixed frequency. Yet, this is done for a limited number of times only before it changes its node status to “failure”. The slave then tries again to propagate this condition to the master via event notification. However, it only tries until the next heartbeating round to reduce the risk of uselessly monopolising the network. After the heartbeating round, the failure state should have been propagated to the master unless the network is entirely broken.

Heartbeating is organised in rounds. The master initializes a heart beat round by sending a round init frame. Every slave then responds in its previously configured time slot. In addition to reporting their node status, slaves are required to solve an easy, but non-trivial problem (e.g., multiplication of two integers). This is known as a *quiz* and provides additional confidence that a node is operating properly.

In case the master finds a slave to be in alarm state (regardless of whether by event notification or by a heart beat report), it also enters the alarm state. If no slave is in alarm state, the master will reflect the presence of any node failure status in the system status. If everything appears to be OK, it still should check upon nodes that are silent or in the init state. This is acceptable during normal operation (maybe a watchdog expired or a sensor is performing a periodic self-test), but nodes remaining excessively long in such states probably indicate a problem.

The master is assigned the highest priority for transmission. Since slaves stop event notification once they receive a round init frame and the heartbeating round includes status exchange, fire alarm reports cannot be infinitely suppressed.

7 Conclusion

The present paper discussed the integration of fire alarm applications into networked BAS. Backward compatible approaches for ensuring absence of feedback at the MAC level for KNX/EIB and LonTalk TP and a system neutral robust protocol were presented. Our next steps will include a first implementation for EIB/KNX to determine compatibility issues and how easily the extension can be integrated. In addition, we will evaluate an entirely time-triggered alternative for the robust transport protocol. Further steps will need to address sharing of the node resources while maintaining absence of feedback also in this respect. Another interesting topic would be to compare the dependability/reliability of our solution to a traditional FAS.

References

[1] DIN EN 54-13, *Fire detection and fire alarm systems - Part 13: Compatibility assessment of system components*, 2005.

[2] DIN EN 54-2, *Fire detection and fire alarm systems - Part 2: Control and indicating equipment*, 1997.

[3] DIN VDE 833-1, *Gefahrenmeldeanlagen für Brand, Einbruch und Überfall - Teil 1: Allgemeine Festlegung*, German Std., 2003.