The application of the timed automata for FlexRay start-up testing

Jan Malinsky[1]

[1] *CTU FEE Prague, Technicka 2, 166 27Prague 6, malinsj@fel.cvut.cz*

***Abstract*-**This contribution deals with modelling of a selected part of the new automotive communication standard FlexRay. In particular, it focuses on the mechanizm ensuring start of a FlexRay network. The model has been created with the use of timed automata and verified. For this purpose the UPPAAL software tool has been used that allows modelling of discrete even systems with the use of timed automata and subsequently to verify the model with the use of suitable queries compiled in the temporal logic. This model can be used for searching incorrect settings of time parameters of nodes in the network that prevent network startup and subsequently the start of the car. The existence of this model also opens the way for finding possible errors and faults in the standard.

I. Introduction

Together with the current trend of development of electronic systems in cars the requirement of a new communication standard has appeared. A new standard has been mainly requested by the x-by-wire technology, which makes it possible to exclude some existing mechanical and hydraulic parts from cars and to replace them with intelligent reliable electronic systems. This is mainly the case of the following technologies: steer-by–wire, i.e. electronic transfer of the turning angle of the steering wheel to the steering angle of the wheels, brake-by-wire, providing electronic transfer of the position of the brake pedal to the operational intervention of the braking system, and drive-by-wire, i.e. electronic transfer of the position of the accelerator pedal. The x-by-wire systems for cars have been taken over from aircraft (fly-by-wire systems), where they have been reliably used for several years. These systems lay the foundations for new intelligent cars where the control system is informed about the driver's intentions (to turn, brake, accelerate, etc.) electronically and together with other electronic systems and sensors decides on safety of the driver's request and subsequently, to enhance safety of the passengers, can intervene into the control of the vehicle. For deeper studies of this new standard the reference sources [1] and [2] dealing with the description of the link and physical layer of the FlexRay communication standard of the latest version 2.1 can be recommended.
FlexRay is based on the TDMA (*Time Division Multiple Access*) method, where time slots are used for the communication. For this reason there must be mechanisms that enable startup of the network. It is very important that this prosses function correctly. This article will describe the model of the startup mechanism of the FlexRay network. For this purpose the UPPAAL [3] software tool has been used that allows modelling of discrete even systems with the use of timed automata [4] and subsequently to verify the model with the use of suitable queries compiled in the temporal logic. UPPAAL has been created in cooperation of the Uppsala University, Sweden and Aalborg University, Denmark.

II. Startup of the FlexRay Network

This chapter will describe the start of the FlexRay network briefly from the moment of initiation of individual nodes through their integration in the network to the achievement of the normal active status. Startup mechanizm means that the nodes must define a common start of the communication cycle and check the same pattern of the time schedule. A complete description can be found in [1].
Fig. 1 presents a simplified view of the startup mechanism of the FlexRay network. Before the startup procedure the nodes must be first initiated to get to the ready status (ready to be incorporated in the network). In each FlexRay network there must be at least two nodes with the coldstart node status. These two nodes first select the coldstart leader among themselves. It will be the node that will first send the CAS (*Collision Avoidance Symbol*) symbol. The other coldstart node will automatically become the following coldstart node. The coldstart leader will then initiate communication by sending startup frames. The following coldstart node, on condition of agreement of time parameters with the leader, will also start sending startup frames to the corresponding time slot. This way both the coldstart nodes

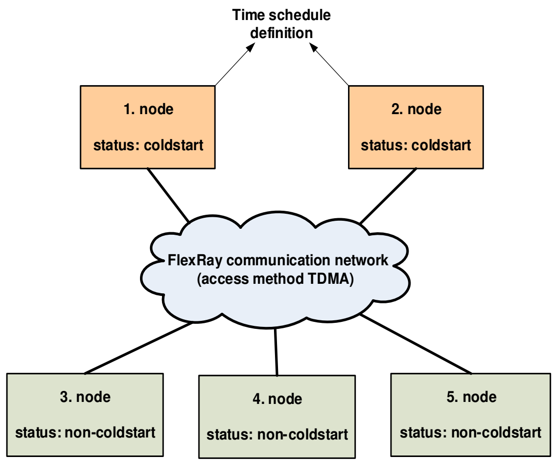

Figure 1. Simplified view of the startup mechanism

define the time schedule and the other nodes with the non-coldstart status, which are subordinated to the coldstart nodes, are able to be integrated in the network on the basis of these startup frames. This integration in communication will only be successful for a non-coldstart node if it has the same timing as it has been set by the coldstart nodes. If everything proceeds correctly, in the communication bus there is a pair of startup frames from both the coldstart nodes in each communication cycle. On the basis of these frames that also have the synchronization function all the other non-coldstart nodes in the network will perform the initial synchronization of their time bases. All the nodes in the network will pass into the normal active status by which the network is started and ready for operation. However, if a non-coldstart node has a different timing from the timing set by the coldstart nodes, it will never manage to get connected to the network, which prevents communication disturbances in the running network. If the time schedules of the coldstart nodes do not correspond to each other, the network will not be started at all, which means that the car will not start either.

III. The model of the startup mechanism

This chapter deals with modelling of the start of a FlexRay standard network. For this purpose the UPPAAL [3] software tool has been used that makes it possible to create a model with the use of timed automata and its subsequent verification [6]. Verification queries for testing the correctness and robustness of the design of the modelled part of the standard have been compiled.

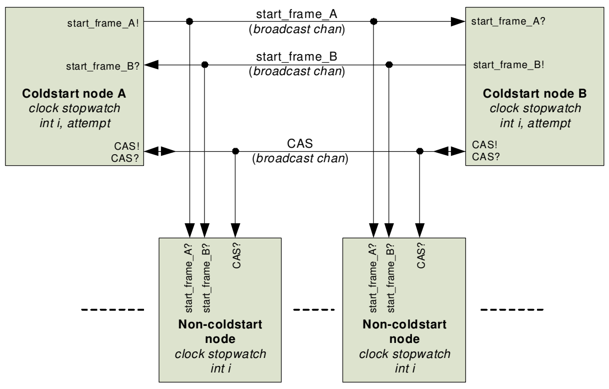

Figure 2. Interfaces of processes modelling the coldstart and non-coldstart nodes

The startup mechanizm of each node is modelled by one process. Individual processes work in parallel within the system (FlexRay network). Two types of processes have been created. The first type models a node with the coldstart status, the other one a non-coldstart node. Fig. 2 indicates interfaces of processes modelling the coldstart and non-coldstart nodes and their mutual interconnection. These processes are interconnected via synchronization channels (having nothing in common with the synchronization of the time base of a FlexRay node, it is just the case of the agreement of terminology of the FlexRay standard and the UPPAAL tool). For modelling of sending and receiving of startup frames synchronization primitives (variable of the *broadcast chan* type) *start_frame_A* , *start_frame_B* have been used. In the coldstart node ***A*** *start_frame_A* represents the transmitted synchronization frame and *start_frame_B* is the frame received from the coldstart node ***B***. In the coldstart node ***B*** it is the other way round. The synchronization channel is also used for modelling of transmitting and receiving the CAS symbol. Fig. 3 presents the startup mechanism in a non-coldstart node and fig. 4 shows startup mechanism in a coldstart node. Both of them were created by using the UPPAAL tool.

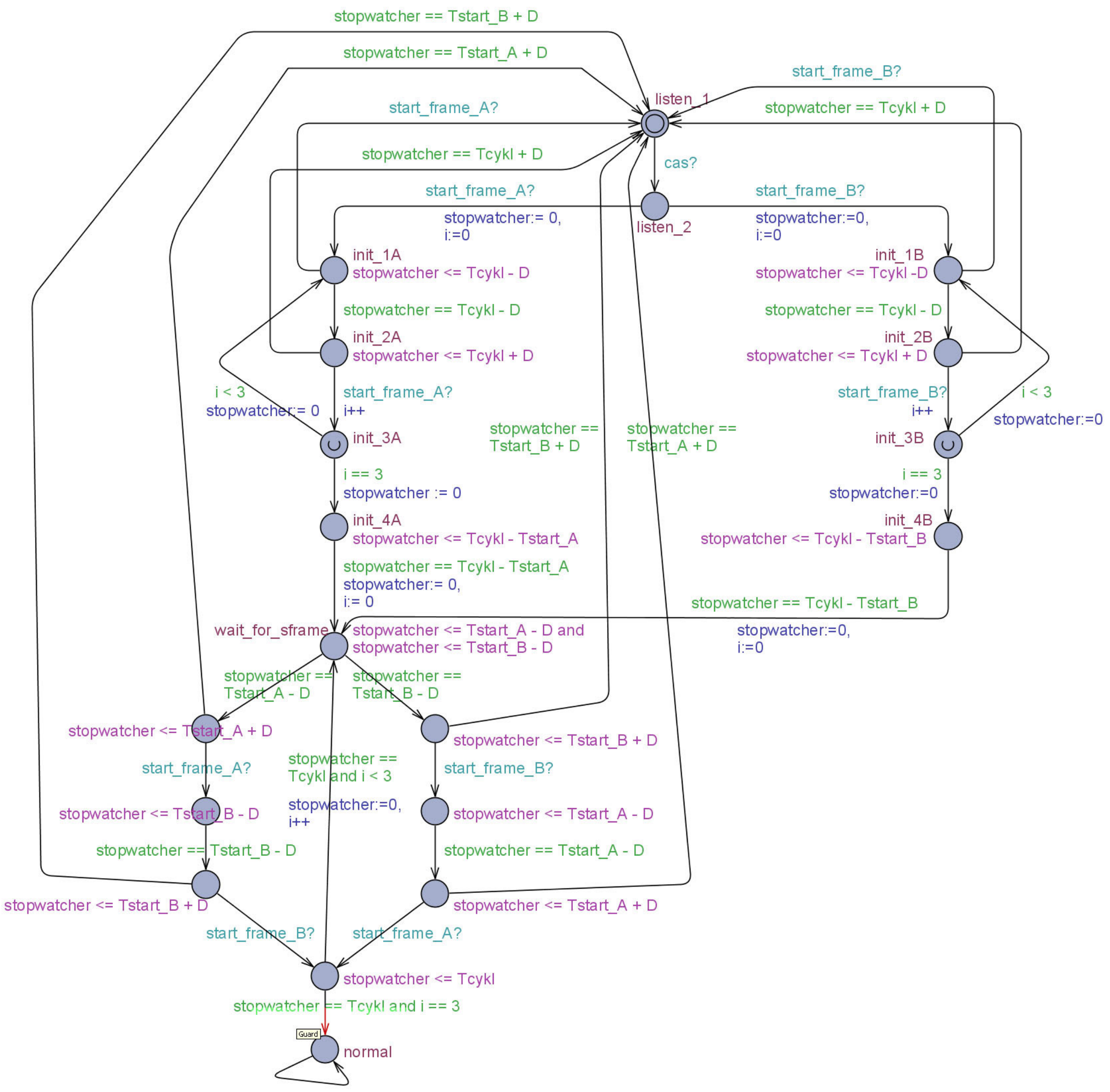

Figure 3. Startup mechanism in a non-coldstart node

The timing of the model in each node is given by:

Tstart [int]
Defines the number of microticks from the start of the communication cycle after which the coldstart node will send its startup frame to the network.

Tstart_other [int]
Defines the number of microticks from the start of the communication cycle after which the coldstart node will expect the arrival of the startup frame from the other coldstart node.
Tstart_A [int]
Defines the number of microticks from the start of the communication cycle after which a non-coldstart node will expect the arrival of the startup frame from the coldstart node A.

Tstart_B [int]
Defines the number of microticks from the start of the communication cycle after which a non-coldstart node will expect the arrival of the startup frame from the coldstart node B.

Tcycle [int]
Defines the number of microticks of the nominal length of the communication cycle of the particular node.

D [int]
Permitted maximum deviation (in microticks) from the expected time of arrival of the startup frame. The coldstart node tolerates the arrival of the startup frame in the interval (Tstart_other – D, Tstart_other + D). A non-coldstart node tolerates the arrival of startup frames in the interval (Tstart_A – D, Tstart_A + D) and (Tstart_B – D, Tstart_B + D).

Stopwatch [clock] - Freely running time base of a node.
Each node has its own assigned time base in the form of the *stopwatch* variable of the clock type. The frequency deviations between some time bases are given by time settings of the model. These deviations involve different view of an accurate global time base. In other words by view of global time base there will be a difference between departure time of start frame from a coldstart node and arrival time of this start frame from other node.
Variables *i* of the integer type serve the execution of internal loops in the processes and *attempt* of the integer type is used for summarization of the number of unsuccessful startup attempts (the leader did not accept the startup frames from the following coldstart node).

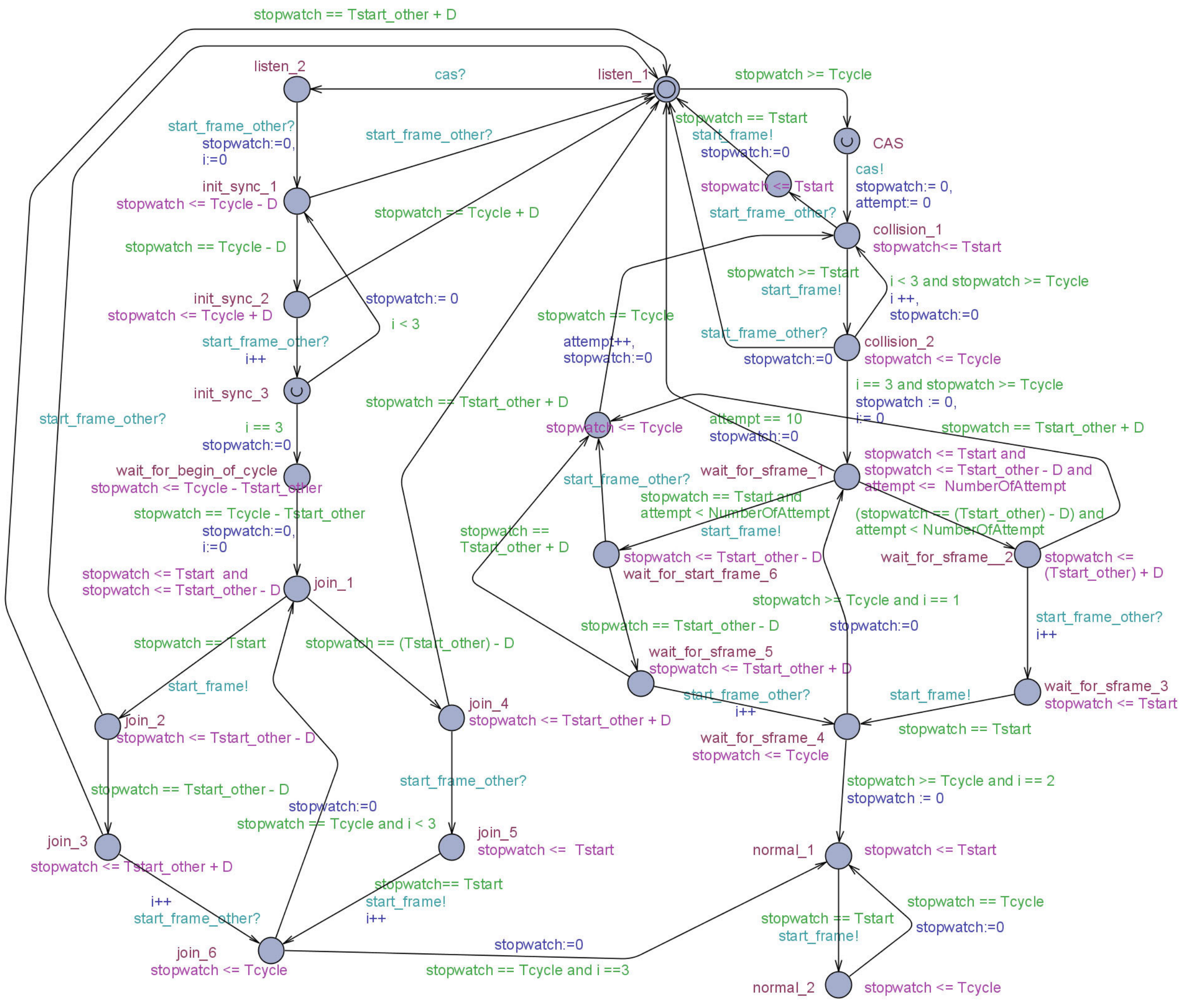

Figure 4. Startup mechanism in a coldstart node

A. Model verification
The UPPAAL software tool also allows you to execute verification queries in the created model. It uses temporal logic. The system where the verification was carried out consists of two coldstart nodes *cs_node_1* and *cs_node_2* and one non-coldstart node *non_coldstart_node_1*. For testing of the network startup four following verification queries were used:

Q1 : Is the system without a deadlock?
A[] not deadlock

Fulfilled only if for all possible paths in the system it is guaranteed that no deadlock status will occur, i.e. such a system status when it is not possible to go any further under any circumstances.

Q2: Is it possible to start the FlexRay nodes A and B? What is the duration of the shortest path to this start?
E<> (cs_node_1.normal) and (cs_node_2.normal)
Fulfilled if in the system there is a sequence of how to get from the initial status of the system to the status where both the processes modelling the coldstart nodes *cs_node_1* and *cs_node_2* have achieved the *normal* points. The duration of the shortest path to this startup can be established with the use of the global time variable *global_stopwatch* and a suitable setting of the UPPAAL tool for looking for the shortest path in the system.

Q3: Is it possible to startup the non-coldstart node? What is the duration of the shortest way to this startup?
E<> (non_coldstart_node_1.normal)

Fulfilled if in the system there is a sequence of how to get from the initial status of the system to the status where the process modelling the normal non-coldstart node *non_coldstart_node_1* has achieved the *normal* point. The process strongly depends on the processes of the coldstart nodes. If for some reason the coldstart nodes are not started correctly, this query will never be fulfilled. The duration of the shortest path to this startup can be established in the same way as in the case of the previous query Q2.

Q4: Is the network always started after the proper selection of one leader? In other words every path from the proper selection of one leader (LISTEN point) leads to the fully startup of the network.
(cs_node_1.listen or cs_node_2.listen) --> (cs_node_2.normal and cs_node_1.normal and non_cs_node_1.normal)

Only fulfilled if after the successful selection of one leader (one node has become the leader and the other one the following coldstart node) all the other possible paths of the system lead to the status where both the coldstart node are in the normal point and all the non_coldstart_startup nodes in the normal point.

B. Evaluation of the results of verification queries
Each query Q1 to Q4 was executed more times, but always with different time parameters of the nodes. In accordance with time parameters settings of the nodes startup mechanism has different results:

- The network can be completely started without complications and the system will not get to the deadlock status.

- While the coldstart nodes are started together, the non-coldstart node is not. The network can not be started completely. The system will not get to the deadlock status.

- The Q2 and Q3 queries finding out whether there is a path how to mutually start the coldstart or non-coldstart nodes are fulfilled (i.e. there is a path enabling the complete startup of the whole network). However, the Q4 query is not fulfilled as for its fulfilment it requires that each path must lead to the full startup of the network. There is no deadlock in the system.

- The coldstart nodes are never started between each other and subsequently the normal node is not started either. The reason is given by too big difference between departure time and arrival time of a startup frame. There is no deadlock in the system.

- The length of the communication cycle is shorter than the arrival times of the startup frames. Not only cannot the network be started, even in parts, but there is also a deadlock in the system.

IV. Conclusions

The article has described a model of the startup part of the FlexRay communication standard including the results of its verification. The result shown that there exist some incorrect settings in the system when FlexRay network is not able to start and subsequently the car will not be started either. Future work will focus on creating a model of the synchronization mechanism. This model together with the already created model of the startup mechanism will provide a clear answer to the question of influencing of the FlexRay network by incorrect time parameters settings. The behaviour of the model will be compare with real system created in according to references [9], [10].

V. Acknowledgment

This research is supervised by Doc. Ing. Petr Kocourek, Cs.c, Ing. Jiri Novak Ph.D and is supported by the Czech Ministry of Education project No. 1M0568 - Josef Božek Research Center of Engine and Automotive Engineering.

References

[1] FlexRay Consortium, "FlexRay Protocol Specification v.2.1 Rev. A", *FlexRay Consortium*, 2005.

[2] FlexRay Consortium, "FlexRay Electrical Physical Layer Specification v.2.1 Rev. A", *FlexRay Consortium*, 2005.

[3] UPPSALA Universitet, AALBORG University: UPPAAL – integrated tool environment for modelling, validation and verification of real-time systems modelled as networks of timed automata. Available at: www.uppaal.com.

[4] Rajeev Alur, David L. Dill.: "A theory of timed automata". *Theoretical Computer Science*, Vol. 126, Issue 2, April 1994, pp. 183–235. ISSN: 0304-3975.

[5] Krakora, J., Hanzalek, Z, "Testing of Hybrid Real – time System Using FPGA Platform", *IEEE Symposium on Industrial Embedded Systems* – IES 2006 [CD-ROM], Lyon: CNRS-ENS, 2006, ISBN 1-4244-0777-X.

[6] Krakora, J., Waszniowski, L., Hanzalek, Z.: "Timed Automata Approach to Distributed and Fault Tolerant System Verification". *In 1st NeCTS Workshop , Networked Control Systems & Fault Tolerant Control*. Nancy: Université Henri Poincaré , 2005, pp. 45-50.

[7] Krakora, J., Hanzalek, Z.: "Checking Real-Time Properties of CAN Bus by Timed automata". *Journal: ACTA 2003, Advanced Control Theory and Applications*. Plovdiv: Technical University, 2003, pp. 130-134. ISBN 954-8779-44-7.

[8] Godary, K., Parrend, P., Augé-Blum, I.: "Comparison and Temporal Validation of Automotive Real-Time Architectures". *In International Conference on Industrial Technology (ICIT'04)*, Hammamet, Tunisia, December 2004.

[9] Malinský, J, "FlexRay Communication Development", *Proceedings EDS 06 IMAPS CS International Conference Brno*, pp. 34-37, ISBN 80-214-3246-2.

[10] Malinský, J, "Measuring System Based on Standard FlexRay and Diagnostics Methods Development". *Applied Electronics 2007*, International Conference, Pilsen: University of West Bohemia, pp.129-132, ISBN 978-80-7043-537-3.