ATLANTIS PRESS

Advances in Engineering Research, volume 155

2nd International Conference on Advances in Energy, Environment and Chemical Science (AEECS 2018)

Temporal Boundary Analysis on Startup Algorithm for Time-Triggered Architecture with Bus Topology

Bao-yue YAN [1, a], Xiang LONG [1,2, b], Mu LI [1, c*]

[1] School of Computer Science and Engineering, Beihang University, Beijing, China

[2] State Key Laboratory of Virtual Reality Technology and Systems, Beijing, China

[a]beyer@buaa.edu.cn, [b]long@buaa.edu.cn, [c]limu@buaa.edu.cn

Keywords: time-triggered architecture, startup algorithm, temporal boundary analysis, arbitrary nodes.

Abstract. The Time-triggered Architecture (TTA) is seen as a widely-recognized design framework for the domain of large distributed embedded real-time systems. This paper derives an elaborated startup scheme and discusses the temporal boundary of it for real-time systems based on TTA, which normally require predictable communication in TDMA environments. The scheme presents an arrival time window (ATW) with a dedicated lower time boundary for contention resolution during startup phase without detecting collisions directly. Although many previous model checking approaches have been taken for analyzing the temporal attributes of the startup algorithm, it is hard to model the startup scenario at arbitrary number of nodes and arbitrary propagation between them. This paper gives the upper boundary of startup time for systems based on TTA with arbitrary number of nodes towards the dedicated startup scheme by formal deduction.

1. Introduction

Safety-critical real-time applications such as aerospace, factory automation, automotive electronics and etc., call for the high reliability and the safety of the computing and communicating systems where a system failure may cause a catastrophe. For many years, these systems have been hand-crafted in an unreliable manner[1]. The Time-triggered architecture(TTA) gives a solution for such systems by establishing a blueprint and a design framework for them[2]. The TDMA is the basic access pattern in TTA with bus topology, which requires the nodes to synchronize their clocks to share a common notion of time. However, the system is basically asynchronous after power on. Consequently, the startup algorithm must be specified to bring a system from asynchronous into synchronous operation within bounded time.

The startup algorithm involves complex interactions of system nodes both in synchronous mode and asynchronous mode and is influenced by the system topology and the channel redundancy strategy, thereby tied to the concrete system implementation and deployment[3]. The correctness of the startup algorithm is the basis of the correctness and reliability of time-triggered systems. Model checking approaches such as timeout-based models, calendar-based models and the mix of timeout-based and calendar model, have been taken for the verification[4,5,6]. But it is hard to model the startup scenario at arbitrary number of nodes and arbitrary propagation between them, thereby posing difficulties for formal analysis of the upper time boundary of the startup algorithm.

In this paper, the authors elaborate the startup scheme based on TTA. The scheme presents an ATW (Arrival Time Window) with a dedicated lower time boundary for contention resolution during startup phase. Moreover, a more efficient contention detecting mechanism is given by the scheme towards the multi-clique problem[7] during the time window for reducing the startup time overhead than standard startup scheme which simply rejects the first received correct startup frame under any circumstances. Formal upper boundary of startup time for a TTA system with any number of nodes is given in this paper towards the delicate startup scheme by formal deduction. Due to space limitation, the paper focuses on the startup scenario only on fault-free circumstances.

2. Startup Model of Time-Triggered Architecture

Basic Concepts. A *node* in a TTA system consists of an HC (Host Computer), a CNI (Controller Network Interface) and a CC (Communication Controller), as shown in Figure 1, which is, also, called an SRU (Smallest Replaceable Unit). The nodes are typically connected by a *TTA bus* who contains dual channels – *channel 0* and *channel 1* normally, and two or more of the nodes form an FTU (Fault Tolerant Unit) defined by OSEK/VDX[8]. The inter-connected system containing the peripherals (sensors and the actuators typically), the nodes and the dual-channels is called a *cluster.*

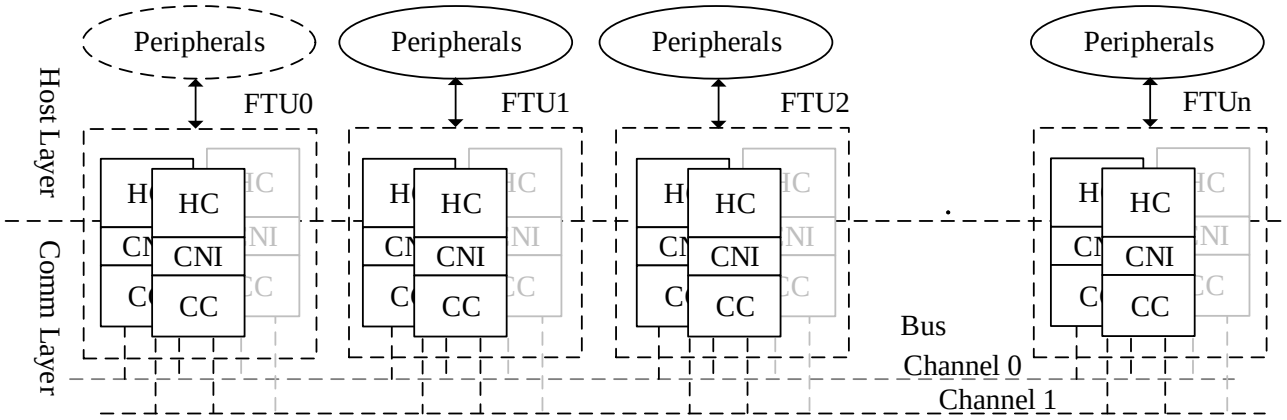

Figure.1 The typical structure of TTA network

A TDMA-based bus access pattern is used by the nodes in a cluster, which means that every active node has a certain amount of reserved bandwidth, the *node slot*, thereby making the bus available for all receiving nodes in the same instance. As to the TDMA access pattern, the periodic sequence of nodes is called a *TDMA round*, and the pattern of the periodically recurring TDMA rounds is called a *cluster cycle*, as illustrated in the left one of Figure 2. The same slot in Different TDMA rounds of a cluster can be assigned to different nodes, but a node can only be specified with a slot in a TDMA round, thus guaranteeing a fair and contention free predictable communication for all nodes.

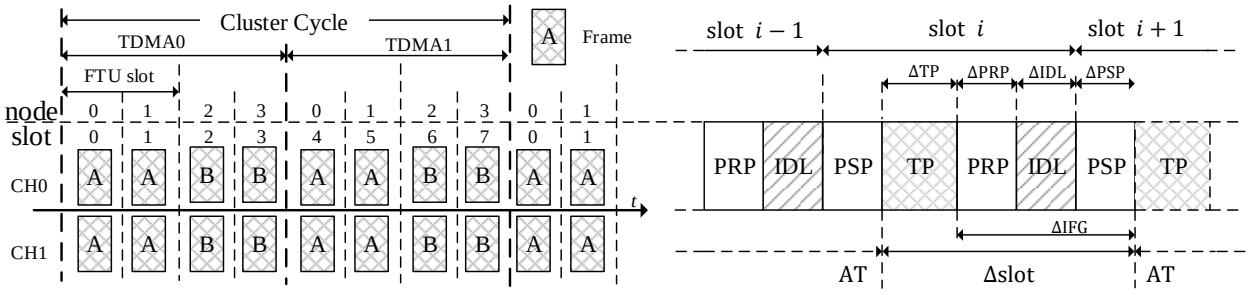

Figure.2 The slot pattern (the left one) and the slot inner timing structure (the right one)

A slot comprises several phases, beginning with the PSP (*Pre-Send Phase*) and ending at the beginning of the PSP of the next slot, as depicted in the right one of Figure 2. The PSP phase is designed for preparing to transfer data and performing other protocol services. The duration of the PSP is denoted with the symbol Δ_{PSP}. The TP (*Transmission Phase*) specifies the point (also called the *Action Time*) and the duration (Δ_{TP}) in time when a data frame is planned to transfer. The action time is perceived on all synchronized nodes of a cluster as the same instance within a predefined precision interval maintained by an FTA (*Fault-Tolerant Average*) clock synchronization algorithm. During the PRP (*Post-Received Phase*), the controller processes the corresponding services according to the received frame data, such as mode change handing, clock synchronization, implicit acknowledgement and etc. within time interval Δ_{PRP}. The IDL(*IDLe*) phase is designed for slot extension, which is merged into the PRP phase for simplification in this paper.

Startup Problems. Collision is the first problem in bus-based distributed TTA systems when the synchronized global time is not available. So the most important step is to guarantee a contention-free bus access pattern for all startup nodes within bounded time, that is, if several nodes have

produced a contention at their n-th access to the bus, the startup algorithm must guarantee a pre-designed x such that the (n+x)-th access of the startup nodes is contention-free[3]. Two kinds of collisions exist for bus-based TTA systems during startup phase, the physical contention and the logical contention, as described in the left one of Figure 3. The physical contention refers to bus access collision when there are nodes sending cold-start frames at approximately the same time and the signals of these frames physically overlay from the perspective of a receiving node. And the logical contention refers to the stagger result of frames because of the long propagation. In logical contention, the simultaneous sending problem happens, but no receiving nodes detect physical overlap of these signals, thus leading to a failure for approaches that rely on hardware collision detection mechanism, for example, the well-known CSMA/CD. As illustrated in Figure 3, if the node m powers on after t_3, then the logical contention will happen but fails to be detected.

The second aspect for the startup of TTA-based systems is the upper bound time of the startup algorithm. The precise upper time boundary of startup possesses essence on time performance estimation and fault diagnosis towards time-triggered real-time systems, while the related model checking approaches pay little attention to the startup time upper boundary. The literature[4] only gives an experimental value at slot granularity for the boundary by increasing the value of the *timeliness* property with small steps until counterexamples are no longer produced by the model checking. The Literature[9] gives a subjective formal value of it, but lack of rigorous proof.

The third problem for the startup algorithm is the unpredicted startup instance for every startup node because of the lack of synchronized global time, which is, also, an inducement of the logical contention problem. TTA recommends that the startup nodes can reject the first receiving frame, thereby compelling the nodes to restart, which works but lengthens the startup time. For this paper, the authors present an ATW (Arrival Timing Window) to figure out some circumstances where the first receiving frame doesn't need discarding, thus reducing the startup time from the perspective of statistics, although the worst case is not improved.

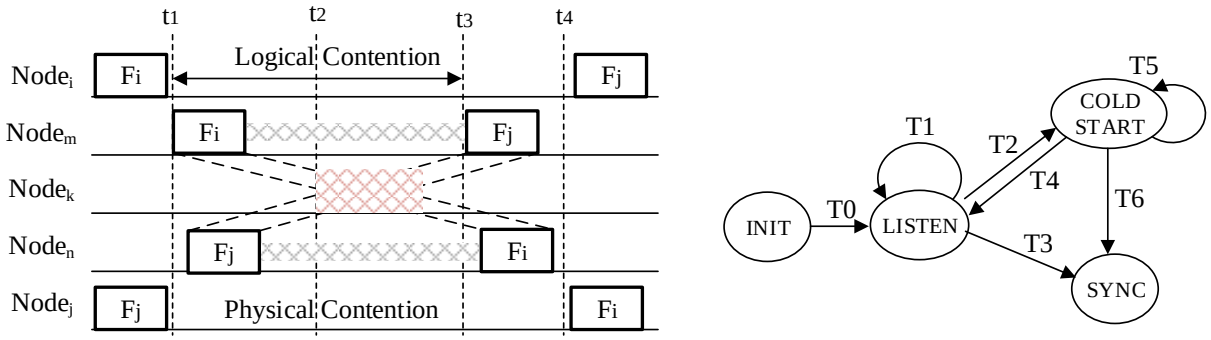

Figure.3 The contention scenario (the left one) and the startup FSM (the right one)

Startup Model. The startup process for a node is depicted by an FSM, as shown in the right of Figure 3. When a valid node finishes the INIT after power on instance, it enters into LISTEN state (T0). In the LISTEN state, the node *i* listens to the channels for time $\Delta^i_{listen} = 2\Delta_{TDMA} + \Delta^i_{startup}$, where the Δ_{TDMA} is the interval of a TDMA round and the $\Delta^i_{startup}$ is the predesigned time delay specified with the value of $\sum^i_{j=0} \Delta slot$, where the $\Delta slot$ is the interval of the j-th slot (assume that the interval of every slot is the same when cold starting). If the node detects traffic signals in any channel, it shall open the ATW with time duration Δ_{ATW} (the duration will be analyzed in the following section). After the end of the ATW, the node shall check whether the received frame is i-frame or cs-frame (it is the name of a kind of frames, it carries the information of the current cluster global time and which slot the cluster is in); if it is, the node transits into SYNC state for i-frame (T3) and transits into COLDSTART state for cs-frame (T2); if no suitable frames are received, the node will reenter into LISTEN state (T1); if no traffic signals are detected during Δ^i_{listen} of node *i*, the node will prepare for cold starting, that is, sends a cs-frame then transits into COLSTART state (T5). In COLDSTART state, the node will try to perform a synchronized operation within time duration Δ_{TDMA}. If the node is trusted by the clique detection algorithm, it will transit into SYNC state (T6); if not, the node will

wait for time duration $\Delta^i_{startup}$ for a frame reception where the node has the same behaviors as them in LISTEN state(T4 or T6); if no traffic signals are detected during $\Delta^i_{startup}$, the node will re-prepare a cold starting (T5).

The safety of the startup algorithm has been verified in literature[1], this paper only focuses on the temporal boundary during the execution of it. As for the ATW, the receiving node (node that receives frames in their LISTEN state or COLDSTART state) has a criterion to judge whether the first frame received during its ATW should be discarded, hence there is no influence on the safety of the result of model checking approaches. The one criterion of ATW is that if a node transits into LISTEN state from INIT state (means that it is the first received frame from power on instance), the first frame received during its ATW shall be discard; otherwise it can be adopted.

3. Temporal Boundary Analysis

Definitions. Symbol definitions are listed in Table 1, where the $TR^k_i(n)$ refers to the arrival instance of a frame sent from a cold start node i to node j at the n-th access to the shared bus, and the $T^i_{start}(n)$ refers to the instance that the node i starts the post-cold-start process, which is specified with the instance of the end of the current slot for cold-start sending nodes and the instance of the end of the ATW for cold-start receiving nodes.

Table.1 The symbol definitions

definitions	annotations
S	The set of nodes that are allowed to perform a cold-start
$S_t(n)$	The set of nodes that send frames at their n-th access to the shared bus
$S_l(n)$	The set of nodes that receive frames at their n-th access to the shared bus
$S_z(n)$	The set of nodes that have not powered on
$T^i_{listen}(n)$	The listen timeout instance of node i at their n-th access to the shared bus
$prop^k_i$	The propagation delay between node i and node k
Δ_{prop}	The max propagation delay in a TTA cluster
Δ_{frame}	The max time consumption for sending a frame
N_k	The pre-designed slot number for node k
$TR^k_i(n)$	The arrival instance of a frame from node k to node i at their n-th access to the shared bus
$hd(n)$	The first node that sends cs-frames in time at their n-th access to the shared bus
$T^i_{start}(n)$	The instance for node i to transit into other state at their n-th access to the shared bus
$T^i_{atstart}(n)$	The instance to open ATW for node i at their n-th access to the shared bus

Boundary of ATW. There are several lemmas in this paper. Due to the space limitation, this paper would not give all the detailed proofs, readers can conclude them according to the definitions and the descriptions of the startup scheme or call for the manuscripts from the authors.

Lemma 1 For the n-th access to the shared bus of nodes, if node i $\in S_l(n)$ and node j $\in S_t(n)$, then the Eq.1 is entailed.

$$\left|T^i_{atstart}(n) - T^j_{atstart}(n)\right| \le \Delta_{prop} \tag{1}$$

Denote that m is the nodes number of set $S_t(n)$, n is the nodes number of set $S_l(n)$ and g is the cold start node, then the Δ_{window} shall be constrained by Eq.2, Eq.3, Eq.4, Eq.5 and Eq.6

$$min(\Delta_{window}) \ge max\{TR^i_k(n) - TR^j_k(n)\} + \Delta_{frame} \tag{2}$$

$$0 \le prop^i_j \le \Delta_{prop}, \forall i,j \in \{1,2,\dots,m+g\} \tag{3}$$

$$\left|T^i_{listen}(n) - T^j_{listen}(n)\right| \le \Delta_{prop}, \forall i,j \in \{1,2,\dots,m\} \tag{4}$$

$$prop^i_j = prop^i_k + prop^j_k, \forall i,j \in \{1,2,\dots,m+g\}, i \le k \le j \tag{5}$$

The $min(\Delta_{window})$ can be generated by MIP solutions if the constrains constants are specified, but for the constrains in this paper, the author choose to conclude the result by formal reasoning.

$$min(\Delta_{window}) \ge max\{T^i_{listen}(n) + prop^i_k - T^j_{listen}(n) - prop^j_k\} + \Delta_{frame}$$

$$\geq \Delta_{prop} + max\{prop_k^i - prop_k^j\} + \Delta_{frame} \geq 2\Delta_{prop} + \Delta_{frame}$$

Boundary of Contention Window. The contention window refers to the time duration from the instance that the node "realizes" the contention to the instance that the node "ensures" the elimination of the contention. The duration of the window is constrained by some lemmas below.

Lemma 2 For the n-th access to the shared bus of nodes, if node i $\in S_l(n) \cup S_t(n)$ and the node j $\in S_l(n) \cup S_t(n)$, then the Eq.6 is entailed.

$$\left|T_{start}^i(n) - T_{start}^j(n)\right| \leq \Delta_{prop} + \Delta_{PRP} \quad (6)$$

Lemma 3 As to the node j $\in S_l(n)$, if node i $\in S_t(n+1)$, then Eq.7 is entailed. That is, if the node is a receiving node at the n-th access to the shared bus when the contention occurs, the node will still be a receiving node at the (n+1)-th access to the shared bus.

$$T_{listen}^j(\mathrm{n}+1) - TR_j^i(n+1) \geq \Delta_{slot} + \Delta_{frame} \quad (7)$$

Lemma 4 For the n-th access to the shared bus of nodes, as to the node i $\in S_z(n)$, if the node powers on before the instance of the (n+1)-th access to the shared bus, then node i $\in S_l(n+1)$. For this lemma, the same constrains can be concluded as Eq.7.

Lemma 5 As to the node i $\in S_t(n)$, if N_i is the smallest one, then node i $\in S_t(n+1)$ and for any node j $\in S_t(n) \wedge N_j > N_i$, the Eq.8 is entailed.

$$T_{listen}^j(n+1) - TR_j^i(n+1) \geq \Delta_{PSP} + \Delta_{frame} \quad (8)$$

The temporal constrains exported by the lemmas above illustrate the temporal relationships between the deinitions of the startup senorio depicted in Figure 4. According to the constrains, the conclusion can be drawed that if the contention occurs at the first access of the nodes to the shared bus, then the contention will be eliminated at the next bus accessing; if at least two nodes are powered on but no contention occurs in the first bus accessing, then the contention is judged to "occur".

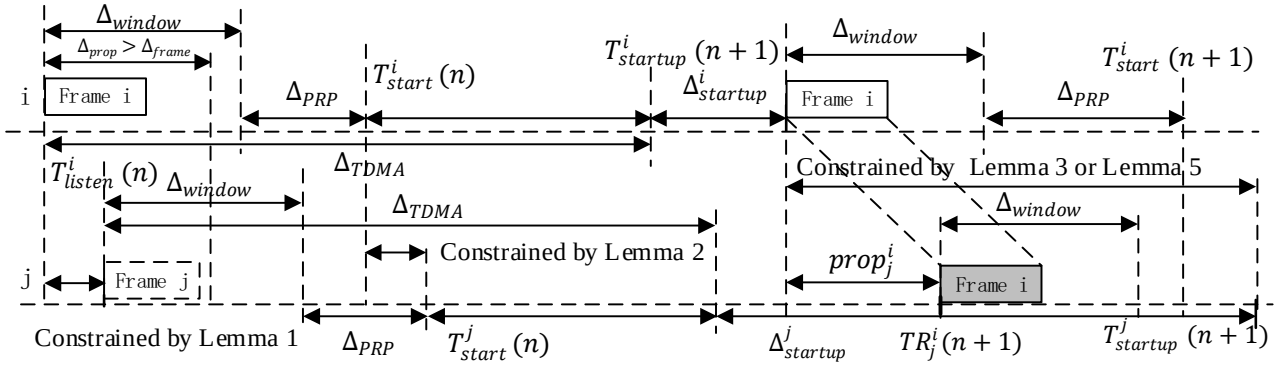

Figure.4 The temporal constrains scenario of the startup scheme between node i and node j.

Boundary of Startup Time. The upper boundary of the startup time, denoted by Δ_{UBS}, refers to the longest time duration from the first sending instance that the cold start node sends the cs-frame when at least two nodes enter the LISTEN or the COLDSTART state, to the instance that at least two nodes reach the SYNC state in this paper. The startup phase can be divided into two phases, the contention eliminating phase and the clique detecting phase. The temporal constrains of the contention eliminating phase have been analyzed in the previous subsection. The clique detecting algorithm specifies a vote mechanism, that is, the sending node has to judge its' dependability according to the status of the received frames in the previous TDMA round before sending frames again. Hence, the Δ_{UBS} can be reasoned by Eq.9, where the $\Delta_{contetion}$ denotes the time duration of contention eliminating phase and the Δ_{vote} denotes the time duration of clique detecting phase.

$$\Delta_{UBS} = \Delta_{contetion} + \Delta_{vote} \quad (9)$$

The upper time duration of contention eliminating phase can be reasoned by Eq.10 from the lemma 1 and the lemma 5 when at least two nodes send frames simultaneously.

$$max(\Delta_{contetion}) = max\left(\Delta_{startup}^k + \Delta_{TDMA} + \Delta_{window} + \Delta_{PRP} + T_{listen}^k(n) - T_{listen}^{hd(n)}(n)\right)$$

$$= 2\Delta_{TDMA} - \Delta_{PSP} + \Delta_{prop} \quad (10)$$

The max time duration of clique detecting equals to the duration of one TDMA round if there is only one node as the sending node, meanwhile there is only one node as the receiving node according to the clique detecting algorithm described in literature[10].

$$max(\Delta_{vote}) = (n-1)\Delta_{slot} + \Delta_{PSP} \quad (11)$$

The max time duration of startup can be concluded by Eq.10 and Eq.11, as depicted in Eq 12, where n is the number of nodes which a TDMA round contains.

$$\begin{aligned}\Delta_{UBS} &= max(\Delta_{contetion} + \Delta_{vote}) \\ &= 2\Delta_{TDMA} + \Delta_{slot} - \Delta_{PSP} + (n-1)\Delta_{slot} + \Delta_{PSP} \\ &= 3n\Delta_{slot}\end{aligned} \quad (12)$$

4. Summary

This paper elaborates the startup scheme of distributed real-time systems based on TTA with bus topology and presents an ATW with a lower time boundary equaling to $2\Delta_{prop} + \Delta_{frame}$ for reducing the startup time from the perspective of statistics. Also, the upper boundary of startup time for systems based on TTA with arbitrary number of nodes towards the dedicated startup scheme is given by formal deduction. The max time duration of startup algorithm with bus topology in case of fault-free is constrained within $3n\Delta_{slot}$.

Acknowledgements

The work is supported in part by the research fund of the State Key Laboratory of Virtual Reality Technology and Systems.

References

[1] Indranil Saha, Suman Roy and S.Ramesh. Formal Verification of Fault-Tolerant Startup Algorithms for Time-Triggered Architectures: A survey[J]. Proceedings of IEEE, Vol 104, Issue 5, May 2016, p:904-922.

[2] H. Kopetz. Real-Time Systems: Design Principles for Distributed Embedded Applications[M]. Second Edition, German, Springer Publishers, 2011.

[3] Wilfried Steiner, H. Kopetz. The Startup Problem in Fault-Tolerant Time-Triggered Communication[C]. Proceedings of the International Conference on Dependent System and Networks(DSN), June 2006, pp:35-44.

[4] W. Steiner, J. Rushby and etc. Model Checking a Fault-Tolerant Startup Algorithm: From Design Exploration to Exhaustive Fault Simulation[C]. in Proc. DSN, 2004, pp. 189–198.

[5] B. Dutertre and M. Sorea. Modeling and Verification of a Fault-Tolerant Real-Time Startup Protocol Using Calendar Automata[C]. In Proc. FORMATS/FTRTFT, 2004, pp:199–214.

[6] I. Saha, J. Misra, and S. Roy. Timeout and Calendar Based Finite State Modeling and Verification of Real-time Systems[C]. In 5th International Symposium on Automated Technology for Verification and Analysis (ATVA), Lecture Notes in Computer Science. Berlin, Germany: Springer-Verlag, 2007, vol. 4762, pp:284–299.

[7] W. Steiner, M. Paulitsch, H. Kopetz. Multiple Failure Correction in the Time Triggered Architecture[C]. The 9th IEEE International Workshop on Object-Oriented Real-Time Dependable Systems(ISADA), Anacapri, Italy, April 2003, pp:3447-347.

[8] OSEK/VDX. Fault-Tolerant Communication Specification[R/OL]. OSEK/VDX Steering Committee, http://www.osek-vdx.org, July 2001.

[9] W. Steiner and M. Paulitsch. The transition from asynchronous to synchronous system operation: An approach for distributed fault-tolerant systems. In The 22nd International Conference on Distributed Computing Systems, Vienna, Austria, July 2002, pp:329– 336.

[10] G.Bauer and M.Paulitsch. An investigation of Membership and Clique Avoidance in TTP/C[C]. In Proceedings of 19th IEEE Symposium on Reliable Systems(SRDS), Nurnberg, Germany, 2000, pp:118-124.