Embedded Networking

CarRing II: A Real-Time Computer Network as Successor of Flexray?

M. Wille and H. Richter
Department of Computer Science, Clausthal University of Technology
Julius-Albert-Str. 4, D-38678 Clausthal-Zellerfeld, Germany
E-Mail: marcel.wille|richter@in.tu-clausthal.de

Abstract

This paper presents a proposal how to avoid today's problems in car electronics, mainly controllers, with respect to scalability, usability and effectiveness. Contemporary car controllers lack of common standards in data exchange as a consequence of incompatible field buses and protocols, while intra-car communication is increasing at the same time. A dedicated, fully-fletched, real-time computer network is presented that is capable to replace all types of field buses in cars. CarRing II exhibits functions from 7 ISO layers, instead of layer 1 and 2 functionality field buses such as FlexRay or CAN. CarRing II aims at general intra-car-communications from a system's perspective and can be used to connect arbitrary sensors, actuators and controllers to a distributed mechatronic system. It optimally supports future x-by-wire driver-assistance systems. Its main goals in comparison to field buses are better usability and effectiveness as well as better reliability and real-time capability.

1. Introduction

Today, most of the innovations in cars take place in mechatronics which is the compound of microcontrollers, sensors, electro motors and electromechanic actuators. Current cars can contain more than 80 controllers for different car functions, e.g. for motor management, for passenger safety, or to support the driver by assistance systems. Classical examples are the anti-lock braking system (*ABS*) or the electronic stabilization program (*ESP*). In future „x-by-wire" applications, such as steer- or brake-by-wire, basic mechanical car functions are replaced by mechatronics. With steer-by-wire in cars, the front wheels are steered in real-time by means of two electric motors according to the rotation angle and torque of the steering wheel. In the extreme case, there is no mechanical connection between steering wheel and front wheels. In the near future, steer-by-wire will assist drivers in dangerous situations, and it will correct driving errors together with *ESP*. In the far future, it will be needed by an auto pilot computer that drives the car with little user intervention to its destination.

In today's cars, about 6 field buses of different types are required. While the number of mechatronic systems per car and the requirements for intra-car communication are increasing, interconnection technologies remain at the level of application-specific field buses of the 1980s exhibiting only layer 1 and 2 functionality of the ISO model. Missing higher-level functions must be emulated on layer 7 by each application individually. As a consequence, most of the efforts for design, implement and test are laid upon the application programmer of a controller, with its obvious consequences in quality, reliability, costs and interoperability. Moreover, only a relatively small number of controllers can be connected to one field bus, resulting in limited scalability, flexibility and high diversity. Open standards for intra-car communication are not in sight while inter-car communication is fostered by several research projects already now. The lack of standards have negative implications for costs and quality and holds up innovative developments [1] [2].

Inside cars, four different types of field buses are dominating as interconnection means for different tasks [3], [4]. These are: the Controller Area Network (*CAN*) [5], the Local Interconnect Network (*LIN*) [6], the Media Oriented System's Transport (*MOST*) [7], and *FlexRay* [8]. Less important are *TTP* [9] and *Byteflight*.

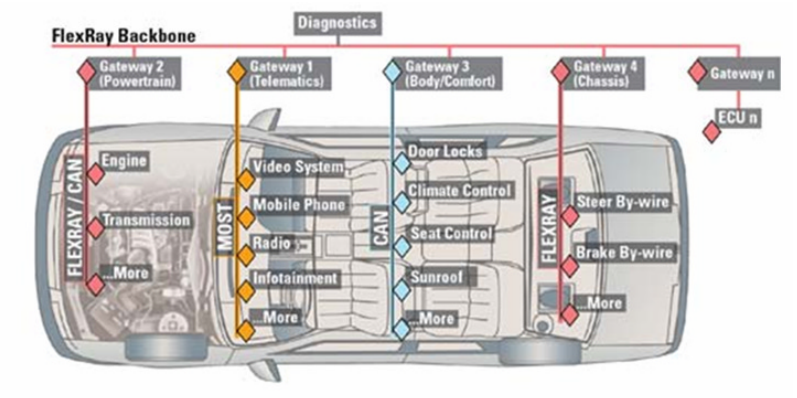

Figure 1: Different field buses in a car.

LIN is a low cost field bus that operates at a data rates of 20 kbits/s. It is based on one master and several slaves connected to LIN. To start a transaction, the master sends a header frame and selects a slave. The slave responds by appending its data to the header frame. LIN is collision free since only the master allocates send bandwidth to slaves.

CAN is event-triggered and allows data rates of up 1 Mbits/s as well as message priorities. It uses Carrier-Sense Multiple-Access with Collision Detection (*CSMA/CD*) [10], [11] as medium access method. If a collision is detected on the common transmission cable, the transmitter with the highest-priority frame wins. Other nodes suspend their transmissions and resume them when the medium is free again which may never happen. *CAN* is not reliable for this reason, and thus not appropriate for safety-critical applications such as steer-by-wire.

MOST is a low-cost transport technology for multimedia communication with a data rate of 24.8 Mbits/s. Data streams are transferred in blocks of frames. Each block contains 16 frames of 64 bytes data. Because of these block transfers, *MOST* is not appropriate for data transfer from A/D or to D/A converters which are part of sensors and actuators. Therefore, *MOST* was not designed for mechatronics and can not be used for it.

FlexRay merges the best features of *TTP* and *Byteflight* and is widely viewed as their successor. *FlexRay* defines synchronous and asynchronous data transfers with time slot allocation according to Time-Division Multiple-Access (*TDMA*). However, one of the main disadvantages is that unused time slots are wasted. Further details of *FlexRay* are given in section 2.

In 2004, a research project on steer-by-wire was started at Clausthal University of Technology with Carring II as one subproject. Details of the prerunner project *CarRing I* are given in [12], [13], [14] and [15]. To overcome the disadvantages of field buses, the *CarRing II* computer network was developed. It is designed for current safety-critical functions as well as for future x-by-wire applications. It has to be noted, that existing computer networks such as *Industrial Ethernet* or computer periphery buses such as *FireWire* are inapplicable in cars since they are not reliable enough and not real-time capable. *CarRing II* is a dedicated, fully-fledged, real-time computer network that exhibits functions from all 7 ISO layers. It will optimally support future x-by-wire driver-assistance systems, and its main goals in comparison to field buses are better reliability and real-time capability as well as better usability and effectiveness. *CarRing II* is a ring-based network with data rates of up to 1 Gbits/s on plastic fibers. It scales to 4096 nodes. Its medium access scheme ensures fairness, provides livelock- and deadlock-avoidance, guarantees an upper limit for packet latency and efficient bandwidth allocation. It allows for automatic car-wide routing, authentication, authorization, common data formats and cares for a new programming model comprising distributed registers and remote interrupts. A safe, two-phase-handshake performs read and write operations between sender and receiver.

The scope of this paper is presentation, evaluation and comparison of some concepts of *CarRing II*, mainly node structure and medium access. Features of *CarRing II* are compared to *FlexRay* by a simulation software called *RingSim*. Simulation runs showed significant performance differences between Carring II and Flexray.

This paper is organized as follows: Section 2 gives an overview on *FlexRay*. Section 3 describes basic concepts of *CarRing II*. In section 4, the comparison results between *FlexRay* and *CarRing II* are presented. The paper concludes with summary and outlook.

2. FlexRay

In this section the main features of *FlexRay* protocol are illustrated. The focus is on topology, communication scheme, medium access and frame format. *FlexRay* was developed for hard real-time applications such as steer-by-wire. It is most comparable to *CarRing I* and *II*. *FlexRay* provides reliability and fault-tolerance for car communication. It can connect up to 64 nodes and has data rates of 10 Mbits/s. *FlexRay* is a field bus, thus only the physical and the data link layers are existing.

There are several ways to deploy a *FlexRay* network. A linear topology is supported as well as a star or multi star. Each *FlexRay* controller is termed *ECU* (Electronic Control Unit). The *FlexRay* protocol supports two channels. This means, an *ECU* is connected with up to two independent communication channels and each channel can have its own topology. There are two different scenarios to use these two channels. First, a frame is transmitted on both channels in order to achieve high reliability and redundancy. Even if one communication channel fails, the transmission can be continued without an error. The second scenario is used to increase bandwidth since an *ECU* uses both channels independently to transmit different data frames. Each data frame contains payload and a header with additional information, e.g. frame identifier and CRC checksum. The Flexray frame format is depicted in Fig. 2.

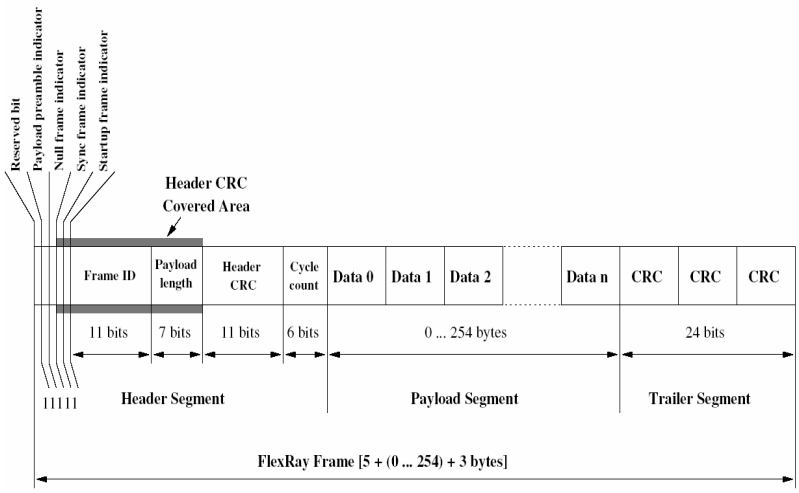

Figure 2: FlexRay frame format.

The „*Payload preamble indicator*“ defines the type of payload transmitted in the frame. „*Null frame indicator*“ signals that the frame contains no payload. Time slots without payload are called *null frames*. *Null frames* are used for clock synchronization. A frame with the bit „*Sync frame indicator*“ set is used for fine clock tuning. A „*Startup frame indicator*“ denotes frames that set up the network in a startup phase. Furthermore, the header contains an identifier (*Frame ID*) to indicate the time slot that may be used for frame transmission. Another field in *FlexRay* frame format contains the payload length which can be up to 254 bytes. The header ends with a checksum (*CRC*). The frame format provides no explicit addressing scheme. Therefore, *FlexRay* is called a broadcast network which means that each *ECU* connected to one channel receives all frames. The *ECUs* contain a filter in order to decide whether a frame is addressed to the respective receiver.

FlexRay provides synchronous and asynchronous data transfers in one communication cycle. Each cycle is divided into synchronous (*fixed*) and asynchronous (*flexible*) time slots

(Fig. 3). Synchronous transfers guarantee an upper limit for frame delivery time which is necessary for real-time. *FlexRay* is an interconnection network with shared communication cables. Therefore, a collision free assignment protocol is necessary for medium access in order to meet real-time requirements.

Medium access in the synchronous part is according to Time-Division Multiple-Access (*TDMA*) [11], [16]. In standard *TDMA*, a round-robin scheme is used to assign bus access to each node. During the synchronous part of a communication cycle, every node has the possibility to send one data frame in its time slot. Collisions are thus avoided. For *TDMA*, every node has to know its time slot number and every slot has to have the same length. The length of the time slots and the slot identifiers are defined during a setup phase. In the asynchronous part, an extension of *TDMA* is used called Flexible *TDMA* (*FTDMA*). The asynchronous part applies a priority-based time slot allocation. It has to be noted, that an *ECU* with low priority may not get any asynchronous time slot. Thus, a frame may never be transmitted. With *FTDMA*, the bandwidth utilization is improved at the expense of real-time capability. Therefore, *FTDMA* can not be employed for real-time transfers since an upper limit for frame delivery time can not be guaranteed. The quota for the synchronous and asynchronous part is flexible. However, an unused synchronous time slot is wasted, which is one of the main disadvantages of *FlexRay*.

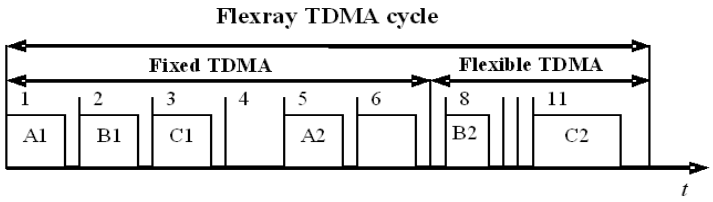

Figure 3: FlexRay communication cycle.

3. CarRing II

CarRing II is part of the interdisciplinary „steer-by-wire" research project at Clausthal University of Technology. The proposed computer network has several main objectives. First, it is developed to solve the problems of field buses such as limited scalability, reliability and flexibility. Second, it meets demands of future car innovations, especially real-time applications. Third, it provides interoperable data exchange between mechatronic components of different manufacturers. Last, it unifies and replaces the multitude of individual field buses required today. Altogether, *CarRing II* is specialized for system-wide intra-car communication and will be fast, comfortable and highly reliable though cost-effective. Currecnt requirements for *CarRing II* are similar to those for *FlexRay* and *FlexRay* can be seen as a first step towards reliable, real-time communication in cars. However, for future applications and increasing requirements a successor of *FlexRay* is needed. Therefore, in comparison to field buses *CarRing II* introduces new communication and programming paradigms for intra-car communication. It is based on systems of rings with end-to-end routing in up to 256 rings. It operates with 1 Gbits/s data rate on Plastic Optical Fibers (*POFs*) at the physical layer. *POFs* are flexible, easy to use, lighter than copper cables, cheap and totally immune against electric, magnetic and electromagnetic noise. Basically, *CarRing II* differentiates between nodes and attachments (Fig. 4). Each node can contain up to 15 attachments, which are e.g. sensors, actuators or controllers. Each ring consists of up to 16 nodes. *CarRing II* scales to 4096 sending or receiving nodes, thus covering every future demand in intra-car communication. It supports future driver assistance applications with better flexibility and more comfortable software interfaces while still maintaining real-time performance. All 7 layers of the ISO model are provided for the application programmer with easy-to-use programming interfaces.

3.1. Reliability Features

On ISO layer 1, *HOTLink II* is used as an industry standard compliant with Industrial Ethernet and allowing reliable *8B/10B* data encoding. Layer 1 control words and a Hamming distance of two are provided for double bit-error detection, and in some cases single error correction without deviating to CRC. Higher reliability can be achieved by double rings in analogy to the double channels of *FlexRay*.

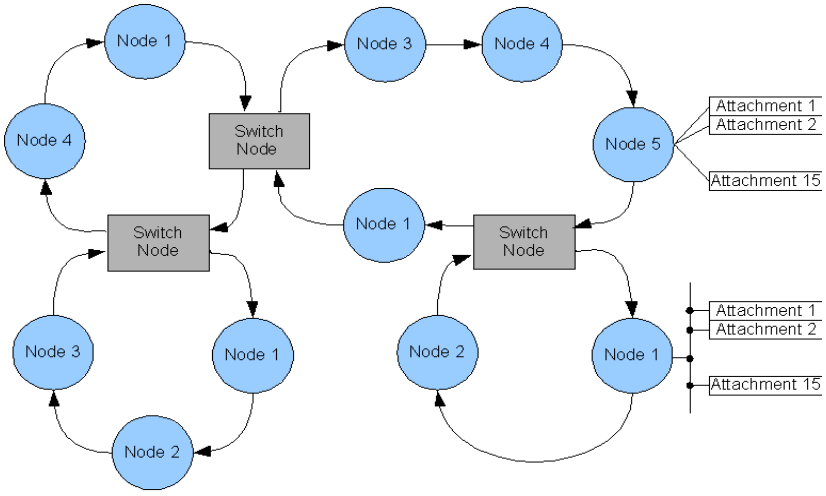

Figure 4: Example of a CarRing II network

Layer 2 increases reliability by redundancy inside each node which is accomplished by means of shadow nodes as backup hardware and a bypass to let packets pass by in case of node failure. The medium access is based on *TDMA*. However, time slots are not implemented by a central master clock but by circulating frames. Thus, the operability of the ring can be checked by every node individually and simultaneously using timeouts. Also an initialization of each ring takes place locally in every node at the same time. There are two CRC checksums, for header and payload respectively. Each received frame must be acknowledged. In case of a negative acknowledge, an automatic frame retransmission inside a ring is performed by the sender. Only one retransmission is made for real-time reasons. The arrival of acknowledges can be checked by timeouts. Receiver contain an overflow buffer that accepts high-priority packets if the normal receive buffer is full. Separate buffer classes are used to avoid deadlocks. On layer 3, reliability is built-in by adaptive routing. Packets will be delivered even if the communication system is partially not operational.

Reliability on layer 4 is augmented by end-to-end connections with a two-phase handshake protocol that is implemented with request/respone packets. A read request has the consequence that a subsequent response delivers the content of the addressed register or memory location. Up to 4 Giga memory locations can be addressed in each of the 15 attachments of a node. A write request effects that a register is written at the receiver and an acknowledge packet is returned to the sender. A timer checks that the corresponding response arrives in time. Packet sequence numbers allow detection of packet loss, duplications and out-of-order delivery. ISO layer 5 is further enhanced by authentication and authorization. Each sender has to identify itself at that node to which the target register or memory cell belongs to. This node checks the access rights of the requester and differentiates between read and write requests. Authentication and authorization is used to

avoid unwanted interferences which are a consequence of the programming model composed of distributed registers and remote interrupts. Furthermore, all end-to-end connections between senders and receivers are checked periodically. Layer 6 provides a common data format for information interchange between controllers to provide for better interoperability.

3.2. Real-Time Capability

Real-time capability is established on layer 1 by several measures such as inline bit processing which is similar to wormhole routing. Short data frames of 64 bytes maximum payload length reduce the delivery time. The restricted set of fixed payload lengths (6, 8, 16 and 64 bytes) allow prediction of frame delivery times according to the chosen payload length. On ISO layer 2, the modified *TDMA* medium access scheme ensures that each node can send one data frame in its transmission cycle. Therefore, a maximum frame delivery time is guaranteed. This access scheme has two add-ons which boost data transmission rate by recycling unused time slots. It avoids livelocks by construction where one node occupies all ring bandwidth on the costs of other nodes. Short acknowledges are inserted into the ring independently from TDMA time slots by a special insertion register in each node. Acknowledge insertion effects the least possible packet latency and is an important feature for real-time capability.

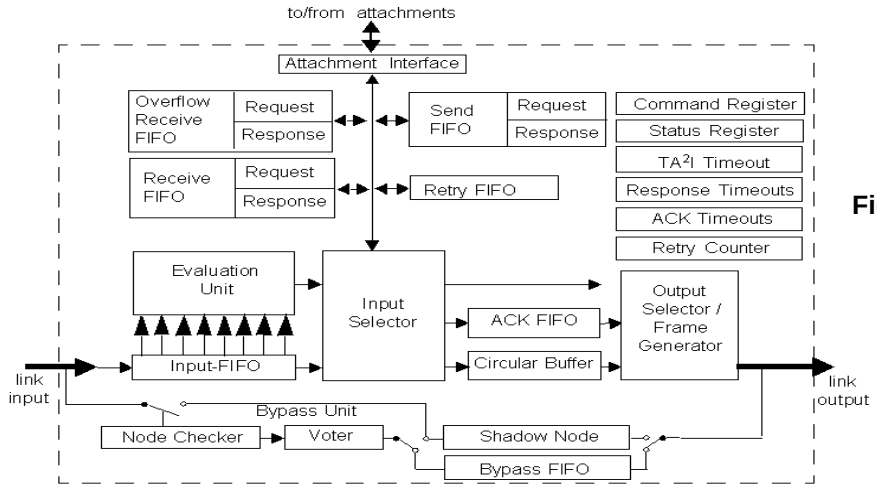

Figure 5: Node structure.

Layer 3 uses virtual connections. A connection set-up phase establishes a fixed route between sender and receiver before data is transmitted. The route is abandoned only in case of failures. On ISO layer 4, real-time capability is kept in contrast to *TCP*, because no sliding-window buffer management and no adaptive time-out is used. In *CarRing II*, the worst-case for an end-to-end packet delivery time $\boldsymbol{D}$ can be calculated in advance by the sum of the transport frame access + the frame delivery to destination + 2 possible retries in layer 2 and 4. This sums up to $\boldsymbol{D = wT(n+1)+Lt}$, where $\boldsymbol{w}$ is normally 1 and increases to 2 if layer 2 and 4 retries are performed. $\boldsymbol{T}$ is one ring circulation time, $\boldsymbol{n}$ is the number of rings involved, $\boldsymbol{L}$ the layer 1 frame length, and $\boldsymbol{t}$ the inverse of the data transmission rate which is 1 ns.

3.3. Node Structure

Each node in *CarRing II* manages the access to the ring, stores data frames temporarily, and transfers data from and to its attachments. There are separate send and receive buffers for request and response frames to avoid deadlocks. A *Retry FIFO* contains sent request and response frames for automatic retransmission if the first acknowledge was negative. An

Overflow Receive Buffer is provided for advanced buffer reservation. It is used by retry frames and frames with high priority if the regular receive buffer is already full. Acknowledges are inserted into the ring by a acknowledge insertion register. This is performed in the *Output Selector / Frame Generator*.
The *Evaluation Unit* checks fields of incoming frames in real-time to initiate appropriate actions, e.g. detection of the frame type. It generates control signals for the *Input Selector* which modifies the fields of outgoing frames, and it changes frame type and inserts payload. Furthermore, it forwards frames which are not addressed to the current node, initiates acknowledge generation and removes own acknowledges from the ring that have circulated the ring one time. Acknowledges are inserted into the data stream by means of the *ACK* FIFO. This FIFO is temporarily switched into the ring, thus increasing its circumference. While the acknowledge is inserted a special *Circular Buffer* with variable length and FIFO semantics holds incoming data frames so that they are not lost. Additionally, the node contains several timers, and command and status registers to manage buffer classes, timeouts, and frame retransmissions.
A *Bypass Unit* is activated if the main node fails and forwards frames to the next node. A *Node Checker* and *Voter* accomplish plausibility checks to detect operational faults. The *Shadow Node* is an optional backup.

3.4. Frame Format

Several frame types are implemented in *CarRing II*, e.g. request, response, acknowledge and transport frames. Most of them have a unique format that is shown in Fig. 6. The type is indicated by a *Frame Type* field.

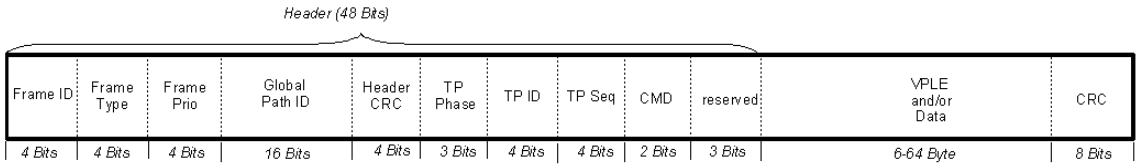

Figure 6: CarRing II standard frame format.

A *Frame ID* is assigned to each transport frame to identify the node and the time slot it belongs to. *Frame Prio* contains the frame priority. High-priority frames can use the overflow receive buffer. *Global Path ID* contains implicitly a system-wide unique identifier for nodes. It is used for routing. The most important header parts are protected by a *Header CRC*. Subsequent fields are required to transmit control information for layer 4 connections. *TP Phase* is responsible for connection set-up and termination. *TP ID* identifies a layer-4-connection and *TP Seq* contains its sequence number. Command such as read, write and interrupt are available in the *CMD* field. *VPLE* is a mechanism to encode variable payload length. The size of the payload (*Data*) field can be selected during a start-up phase. The maximum size is 64 bytes.

3.5. Medium Access

The proposed medium access scheme is a variant of Time-Division Multiple-Access (*TDMA*) together with Register Insertion (*RI*). It is called ***T**ime Slot **A**ccess with **A**cknowledge **I**nsertion (TA^{2}I)*. *TA^{2}I* guarantees real-time capability and a fair bandwidth allocation. For reliability reasons, *TA^{2}I* implements time slots of *TDMA* not by a master clock but by empty circulating frames called transport frames. Therefore, bandwidth allocation is not a single point of failure since no master is needed for clock synchronization and time slot allocation. All transport frames are created during a start-up phase and each transport frame is allocated to one node. The time required for a transport frame to circulate the ring one time

is called a transmit cycle. Each node can use its own transport frame to send one data frame per transmit cycle by converting it into a data frame. At the receiver, the data frame is converted back into a transport frame for later usage by its owner. Transfers are synchronous because sending takes place with the arrival of the sender's transport frame. Each node has a timeout to detect the loss of its transport frame. One advantage of *TA²I* is that no master is needed for bandwidth allocation. The second is that the operability of the ring is checked permanently by monitoring the arrival of transport frames. The drawback of *TA²I* is that the performance drops in case that one sender wants to send more than the others. Furthermore, unused transport frames are lost. Two extension algorithms solve these problems.

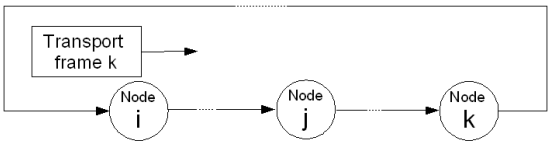

Figure 7: „i-to-j" transfer.

The first extension is called ***E**xtended **T**ime Slot **A**ccess with **A**cknowledge **I**nsertion (ETA²I)* that allows the recycling of unused time slots by *"i-to-j" transfers*. To ensure such an asynchronous usage of time slots, a so-called i-to-j transfer must exist. Assume there is a directed ring of *n* nodes and transport frames *1,...,n*. The transport frame *k* may belong to node *k*. An *"i-to-j" transfer* exists, if transport frame *k* can be reused by nodes *i* and *j* if they are located "before" node *k* (Fig. 7).

The disadvantage of *ETA²I* is that if no such transfer exists, transport frames can not be reused and bandwidth is still lost. The second extension solves this problem. With ***O**pen **T**ime Slot **A**ccess with **A**cknowledge **I**nsertion (OTA²I)*, each node can grant access to its transport frame to any other node for an arbitrary period of time. Such frames are called opened transport frames. The owner of a transport frame converts it into an opened transport frame if the node's send buffer is empty. If there is data to be send the node converts the opened frame back. In the worst case, a node has to wait one transmit cycle to use its own transport frame if it is used by another node. The owner of an opened transport frame converts the frame to inform other nodes that it can not be reused in the next circulation. Real-time is maintained because at most two cycles are needed for opening and closing transport frames. Both described extensions can be combined giving ***E**xtended **O**pen **T**ime Slot **A**ccess with **A**cknowledge **I**nsertion (EOTA²I)*.

Beside transport frames, short acknowledges are inserted into the ring to allow for a handshake thus enhancing reliability. To improve performance, acknowledge insertion occurs independently from transport frames ensuring efficient bandwidth utilization. An acknowledge is inserted into the ring by a FIFO holding its bits. The maximum ring circumference corresponds to the number of nodes in the ring. In the worst case, all nodes insert acknowledges simultaneously. The receiver of an acknowledge has to check if it is positive or negative. In case of a negative acknowledge, the frame is retried. After one circulation, the acknowledge is removed by the sender's acknowledge FIFO thus the ring is scrubbed.

3.6. A Reliable Communication Scheme

A lasting end-to-end connection between sender and receiver is called session and is established on layer 5. Within each session, a sender can set up at most 16 layer-4-communications at the same time to allow for multithreading. Each communication is attributed with a *TP ID* to differentiate between the 16 transfers. By sender multithreading, bandwidth is used efficiently because the node is not busy waiting for a response. A layer-4-communication has a two-phase handshake implemented by request/response. A response packet is the answer of its previous request packet. Data transmission keeps cause and effect by strict time management in each layer-4-communication. On this layer, packet losses, packet duplications and out-of-order delivery are also detected. A system-wide *Global Path ID* is assigned to each sender-receiver-pair. This identifier is stored in each

intermediate node to make routing decisions. Address information of 48 bits for the packet's destination are transmitted only in the communication set-up phase to the target. Each node can perform read and write operations according to the new programming model of distributed registers and remote interrupts. Reliability is achieved by authorization and authentication process that protect nodes from unwanted reads or writes.

4. Simulation Results

For *FlexRay,* a simulation software was developed by us which is based on the *NS-2 simulator*. Simulations show throughput and latency. The obtained diagrams are performance comparisons between *CarRing II* and *FlexRay* under normal and overload conditions. All runs were made with the following parameters: simulation time is 32 ms, buffer size is 4, payload length is 64 bytes, and the node number is 6. *FlexRay* connects the nodes by a single bus while *CarRing II* uses a single ring. Sender data rates were ramped up between 0 and 10 Mbit/s. *CarRing II* applies the *EOTA*^{2}*I* medium access scheme. It has to be noted that the data rate of *CarRing II* was intentionally reduced to 10 Mbit/s to make the comparisons.

Fig. 8 shows simulation results for a network with one ongoing communication (1 sender, 1 receiver). All other nodes are inactive. The left diagram shows the summed data rate that senders try to transmit (*Gross Send Rate*) versus the payload really received at the destination *(Net Receive Rate*). *Gross Send Rate* includes payload and header while the

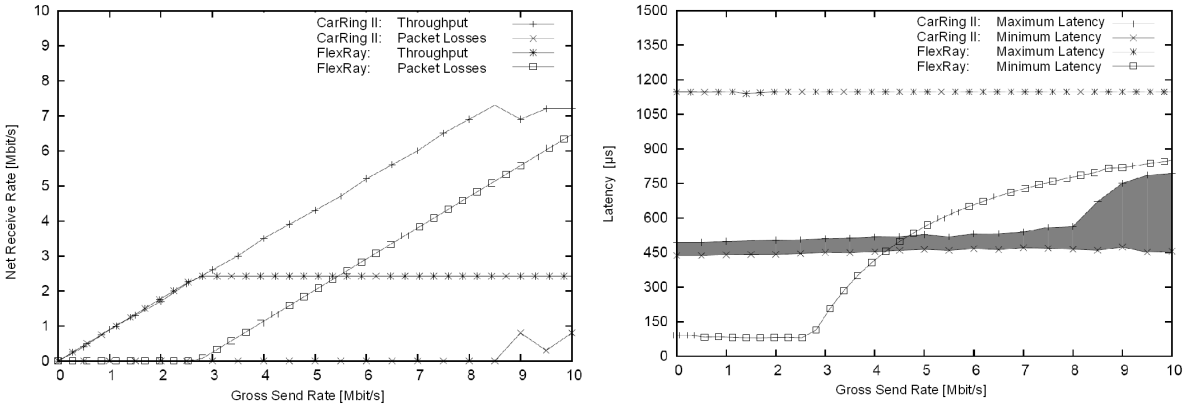

Figure 8: Throughput and latency for a network with 6 nodes (1 sender, 1 receiver).

Receive Rate denominates the payload. Minor convergence in the curves for packet losses can be explained by the header size. With the given 64 bytes payload, *CarRing II* has to transmit a larger header as Flexray. For *CarRing II*, the throughput saturation point is reached if more than 8.5 Mbit/s are tried to be transmitted. The saturation of *FlexRay* is already reached at 3 Mbit/s. The explanation is that *CarRing II* can use all time slots that were otherwise lost because of its advanced medium access scheme.

Maximum and minimum latencies are considered since this is important for real-time applications. Simulation results are depicted in Fig. 8 on the right. It shows that latencies are stable and predictable for *FlexRay* and *CarRing II* under normal and overload conditions as well, provided that Flexray uses only TDMA and not its FTDMA extension. The reason for this restriction is that FTDMA leads to unpredictable packet delivery times. Packets may never be delivered.

The diagrams in Fig. 9 show the results for 3 senders and 3 receivers. *CarRing II* achieves a significant better bandwidth utilization in this case, while the latencies of *FlexRay* are getting

better compared to *CarRing II* when the number of nodes is increasing. However, *CarRing II* operates only with 1% of its possible data rate of 1 Gbit/s, and therefore its latency is high.

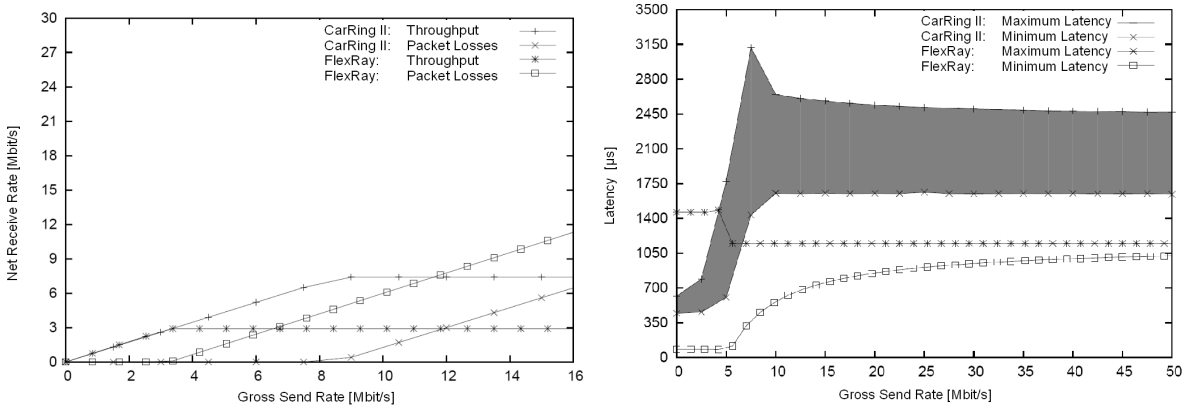

Figure 9: Throughput and latency for a network with 6 nodes (3 senders, 3 receivers).

5. Conclusion and Future Work

A capable successor of *FlexRay* called *CarRing II* is presented. It is a reliable, ring-based, real-time, high-speed computer network for intra-car communication that can replace field buses. It is a proposal for avoidance of today's restrictions of field bus communciation and meets requirements of future X-by-wire applications as well. Its medium access scheme guarantees fair and efficient bandwidth allocation. Simulation results show that *CarRing II* achieves a significant better performance than *FlexRay*. Future work deals with implementation and verification by measurements at a test stand.

References

[1] M. Randt , *Proc. Workshop Bussysteme im Automobil ECT 2002*, Augsburg 2002.

[2] Frost & Sullivan, *European Automotive Market for X-by-Wire Technologies*, B015-18, London, 2001.

[3] K. Reif, *Automobilelektronik - Eine Einführung für Ingenieure*, Vieweg Verlag, Wiesbaden, 2006.

[4] W. Zimmermann and R. Schmidgall, *Bussysteme in der Fahrzeugtechnik*, Vieweg, Wiesbaden, 2006.

[5] Robert Bosch GmbH, *CAN bus specification*, ISO std. 11898/11519, http://www.can.bosch.com

[6] LIN Consortium, *LIN bus specification*, http://www.lin-subbus.org

[7] MOST Cooperation , *MOST bus specification*, http://www.mostnet.de

[8] Flexray Consortium, *Flexray specification version 2.1*, http://www.flexray-group.org

[9] TTTech Group, *TTP specification*, http://www.tttech.com, 2006.

[10] IEEE, *Carrier Sense Multiple Access with Collision Detection (CSMA/CD): Access Method and Physical Layer Specifications*, IEEE Standard 802.3, IEEE, 1985.

[11] A.S. Tanenbaum, *Computer Networks*, 4th ed., Prentice Hall, 2003.

[12] H. Richter et.al., *A concept for a reliable, cost effective, real-time Local Area Network for automobiles*, Proc. Embedded in Munich and Embedded Systems, Munich, 2004.

[13] H. Richter and M. Wille, *Efficient Bandwidth Allocation in a Ring-Based Real-Time Network for Automobiles*, Proc. 12th IEEE Int. Symposium on Modeling, Analysis and Simulation of Computer and Telecommunication Systems (MASCOTS), 2004, pp. 668-671.

[14] H. Richter and M. Wille, *A High-Performance Local Area Network for Cars*, Proc. 3rd IASTED Int. Conference on Communications, Internet, and Information Technology (CIIT), 2004, pp. 303-310.

[15] M. Wille and H. Richter, *Carrings Medium Access Methods in Comparison to FDDI*, Proc. 2005 IEEE Intelligent Vehicles Symposium (IV'05), 2005, pp.530-537.

[16] I. Rubin and J. Baker, Media *Access Control for High-Speed Local Area and Metropolitan Area Networks*, Proc. of the IEEE, vol. 78(1), 1990, pp. 168-203.