CARRING II: A Reliable Real-Time Network

Marcel Wille, Harald Richter, Christian Asam
Department of Computer Science, Clausthal University of Technology
Julius-Albert-Str. 4, D-38678 Clausthal-Zellerfeld, Germany
Email: marcel.wille|richter|christian.asam@in.tu-clausthal.de

***Abstract*—The paper introduces CARRING II, a further developmentof the CARRING communication network. CARRING is a reliable, ring-based, real-time Local Area Network (LAN). It was developed to connect e.g. mechatronic systems in cars, but it is also applicable to general LANs. The proposed network tries to solve the known problems of LANs with bus topology, e.g. limited scalability or limited reliability. For CARRING II, the node structure, frame types, a buffer management with a special reservation mechanism and some other basic concepts are described here. Furthermore, CARRING II applies a medium access scheme that merges concepts of Time Division Multiple Access (TDMA) and Register Insertion (RI).**

I. Introduction

Today, high-end cars contain up to 80 microprocessor-based controller units. The combination of controller units, sensors and actuators in cars is called a mechatronic systems. The number of such systems and the involved communication requirements are increasing while the interconnection technology for Local Area Networks (LANs) in cars remains at the level of conventional bus topologies. Furthermore, several different car network types are required to connect controllers with sensors and actuators. State-of-the-art in car communication are the Controller Area Network (CAN) [1] for real-time functions such as motor control or airbag ignition, the Local Interconnect Network (LIN) [2] for simple functions like windshield wiper or door locks, the Media Oriented Systems Transport (MOST) [3] for multimedia communication, and Flexray [4] for high-speed real-time functions in next generation cars. Future cars will provide so called "x-by-wire" functions, e.g. steer-by-wire, brake-by-wire, or throttle-by-wire. All parts of such an application must be highly reliable, robust, redundant,
and real-time capable. A single standard solution for real-time communication in cars is not in sight. The result is that car manufactures have serious quality and performance problems because of many communication networks, controllers and its software. Therefore, the motivation was to develop a new reliable real-time communication network, especially for current safety critical functions and future "x-by-wire" applications in cars.

CARRING II is a part of the "steer-by-wire" research project at Clausthal University of Technology which started in 2004. The goal of CARRING II is to develop a single unifying solution for a communication network that solves the known problems of LANs with bus topology such as limited scalability and limited reliability. It has to provide a reliable connection of all electric components required for steer-by-wire in real-time. Pure computer LANs such as Fast Ethernet, Fire Wire or USB are inapplicable for cars since they are either too expensive, too complex or for lack of real-time capability.

Previous developments of CARRING are illustrated in [5],[6],[7],[8]. In comparison to older specifications, CARRING II applies an extended node structure that corresponds to the new medium access scheme and the new buffer management. The medium access scheme contains a variant of Time Division Multiple Access (TDMA) together with Register Insertion (RI). An extension algorithm allows recycling of unused time slots. For real-time transfers, a maximum frame delivery time can be guaranteed because within a TDMA cycle each node can send one data frame in its time slot. Register Insertion is used to effectively transmit short acknowledges. All together, the medium access is cost-effective, achieves maximum bandwidth utilization, and it behaves stable even in case of overload. The buffer management controls different buffer classes to implement the transaction concept and the buffer reservation scheme. A transaction concept is implemented for reliability reasons. Furthermore, CARRING II supports systems of rings with a routing from ring to ring to achieve scalability.

The scope of this paper is the presentation of new features and concepts of CARRING II, e.g. node structure and buffer reservation scheme. Some concepts are further developments of the older CARRING communication network. The proposed concepts define a reliable, real-time network with a ring topology. A simulation software called RingSim is still under development. First results will be presented in future works which deals with the medium access scheme.

The paper is organized as follows. Section 2 gives an overview of the CARRING II features. Section 3 describes the extended node structure and section 4 describes frame types and frame layout. In section 5 the transaction concept is illustrated. Section 6 gives an overview of the medium access scheme of CARRING II. Section 7 describes the buffer management together with a buffer reservation mechanism. The timeout mechanisms are presented in section 8. The paper concludes with a summary and an outlook to future work.

II. CARRING II Features

CARRING II is a high speed communication network. It combines high electromagnetic compatibility and high overall reliability with low cost. The proposed network uses proper cabling. A maximum frame delivery time can be guaranteed which is mandatory for real-time. Furthermore, it provides high scalability. Fair bandwidth allocation is guaranteed with the medium access scheme. The following subsections describe the mentioned features in detail.

A. High Speed Network

CARRING II uses bit serial communication. It achieves data rates of 200 Mbit/s with plastic optical fibers (POFs).

B. Proper Cabling

A cabling is used that is suitable for a dusty and oily car environment. Three different types of cables may be used in CARRING II: plastic optical fiber (POF), plastic cladding silica (PCS), and fiber optics in the prototype.
POF is an optical fiber for short distances of up to 400 m that is made of polymer as the core and fluorinated polymer as the cladding material. It is nearly as cost-effective as twisted pair copper cables because of low overall sytem cost that comprises costs of fiber, conversion modules, connectors, installation, test equipment, and training requirements. POFs are flexible, easy to handle, and easy to connectorize. Main advantages are the immunity to electromagnetic interferences, an increased reliability and security, and much lower weight than copper cables. A disadvantage is that POFs have high absorption. But this fact is negligible for short distances.
PCS is a hybrid fiber that is made of silica as the core and a polymer as the cladding material. It offers many advantages of POF but also twice the cost of POF. One advantage is that PCS is capable to operate at higher temperatures of up to 125 degree Celsius which may occur in the engine compartment.

C. Electromagnetic Compatibility

CARRING II uses a HOTLink II™[9] transceiver on the physical layer for optical communication. Nodes are connected in a point-to-point manner using POF or PCS which are absolutely immune against all electromagnetic noise and grounding problems.

D. Fair and Reliable Bandwidth Allocation

Medium Access has to be provided in each LAN in a fair, fast, and reliable manner. A single master node would simplify medium access, but also exhibits a single point of failure. Instead, a decentralized scheme is proposed that bases upon a combination of a variant of Time Division Multiple Access (TDMA) and Register Insertion (RI). The sum of all time slots is equal to the number of nodes. Its value is called TDMA cycle. To achieve fair bandwidth allocation, each node can always use its own time slot to send a data frame.

E. Real-time Capability

The medium access scheme ensures that each node can send one data frame per TDMA cycle. Therefore a maximum frame delivery time can be guaranteed also under overload conditions. The latter is mandatory for real-time transfers.

F. High Reliability

CARRING II uses 8B/10B coding [10],[11] to transfer data bytes and to detect transmission errors at the physical layer by invalid code words. The coding reliably indicates begin and end of data frames by sending special code words, and it allows transmitting out-of-band signals by control code words.
Multiple rings are supported for scalability and redundancy. In case of ring failure another ring can be used to transmit a data frame from sender to receiver by adaptive routing. Each node has redundancy and supports reconfiguration in hardware. A 8 bit cyclic redundancy checksum (CRC) is used to protect the header and short data frames of 2, 4, 8, 16, 32, or 64 bytes of payload.
With continuous circulating empty data frames of the TDMA variant, the operability of the ring can be checked by every node using timeouts. No master is needed to regenerate global clock and assign time slots to nodes, thus no single point of failure exists.
Furthermore, a transaction concept instead of simple send or receive calls, automatic frame retransmission in case of negative acknowledge, comprehensive time out mechanisms and many other features are implemented in hardware for reliability reasons.

G. High Scalability

First, systems of rings ensure scalability, e.g. the network for a small car can be a subset of that of a large car. There are individual rings for different controllers and requirements, together with a routing from ring to ring. Second, for a limited number of nodes CARRING II scales with the number of nodes in a ring which was shown by simulation. Third, the packet size is scalable because the payload length, i.e. the length of the data field is selected during the ring initialization.

H. Low Cost

CARRING II uses 1mm POFs that are just as cost-effective as copper cables, have less weight than copper cables and are very easy to handle. Only 1 POF cable is required for transmission between 2 neighbor nodes. Unidirectional point-to-point POFs between every 2 nodes form a closed polygon called "ring". There is a cost-sharing among the nodes because one ring connects several nodes. All data packets circulate from sender to receiver and vice versa via the ring. That corresponds to a full duplex capability via an unidirectional ring.
Furthermore, a medium access scheme is used that efficiently allocates ring bandwidth by three extensions of standard TDMA (see section 6).

III. Node Structure

Each node in CARRING II consists of a node chip and up to 15 attachments that are e.g. the elements of a mechatronic system. Several node chips are connected with POFs to a ring. The node chip manages the access to the ring, temporarily stores data frames, and transfer data from and to the attachments. Attachments are connected to the node chip via an attachment bus. Figure 1 shows the extended node chip setup of CARRING II.

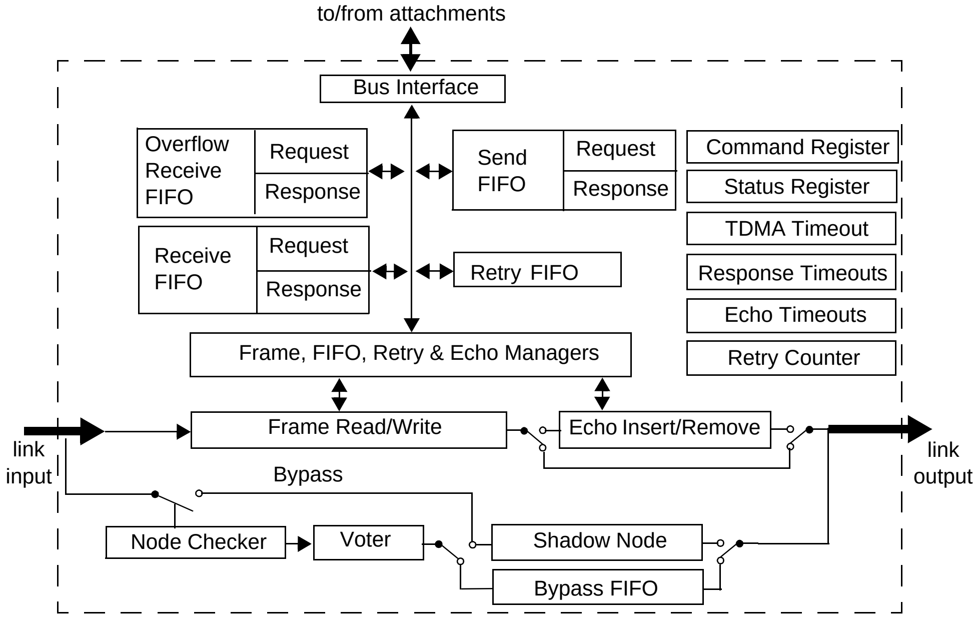

Fig. 1: Node chip setup

A node chip consists of a link interface to CARRING II and a bus interface to the attachment bus. There are separate send and receive buffers for request and response frames to avoid deadlocks. Both are implemented as circular buffers. The receive buffer temporarily stores received frames for further processing in the attachments and the send buffer temporarily stores outgoing frames from the attachments. The buffers have to preserve the time ordering of incoming or sent frames because for x-by-wire applications a step-by-step sequencing is indispensable. A Retry FIFO stores transmitted request and response frames for automatic retransmission until a positive acknowledge (echo) arrives. Retry frames of rejected data frames, i.e. with negative echo, have an increased priority. Therefore, retry frames have to be sent before new request or response frames are transmitted. An additional overflow receive buffer is provided for buffer reservation mechanism. It is used by retry frames and frames with high priority if the regular receive buffer is full.

Echo frames are injected in the ring independent of time slots by Register Insertion. Register Insertion means that a register is temporarily included in the ring in order to send a short echo frame. The receiver of an echo frame has to check the echo in order to decide if the sent request or response was successfully stored at the receiver. If the echo frame returns to the sender after one circulation it is excluded again. Such a register is planned in the node chip as Echo Insert/Remove unit. The register of the Echo Insert/Remove unit is implemented as circular buffer that holds the incoming data frames as long as the echo is inserted into the ring. The Frame Read unit checks the frame fields of incoming frames in real time to initiate appropriate actions, e.g. comparison of frames target ID and the node ID. The Frame Write unit modifies the fields of outgoing frames in real time, e.g. sets frame's target ID or inserts payload data from send buffer. Both are provided as one serial shift register with "write after read" semantics.

The Bypass Logic consists of a node checker, voter, shadow node, and a bypass FIFO. The node checker accomplishes a plausibility check to ensure that the node works correctly. A voter compares results from main node and shadow node in order to decide if the main node fails and to activate the bypass logic if necessary. The shadow node is backup hardware that contains an identical node chip setup. If the main node fails the shadow node is activated by the voter. A bypass FIFO forwards frames if main node and shadow node fail to keep the ring operational.

The status register is responsible for error handling. A retry counter counts the number of frame retransmissions for each transaction. A maximum of three retries per transaction are allowed in order to avoid ring overload. The command register contains instructions that are important for timer programming, payload initialization or soft reset.

The timeout for each transaction is realized by three timer values. A TDMA timeout checks if the empty frame (transport frame) returns in time. The echo timer checks if the acknowledge of a request or response arrives in time and the response timer detects lost frames that were not received as an answer of a request.

Finally, the node chip contains miscellaneous components that manages e.g. the buffer classes, the comparison of frame ID and node ID, or the generation of echo and retry frames.

Contrasting the application view of the node structure, the implementation view consists of the Input Manager, Input Register, Receive- and Send Buffers, a Cyclic buffer, and an Echo Insertion Manager. Data arrives at the input port at a constant rate which is determined by the physical layer. The Input Register delays the received data that the Input Manager can evaluate the header information. The Input Register is implemented as shift register. Echo frames are removed silently by not shifting them to the Cyclic Buffer.

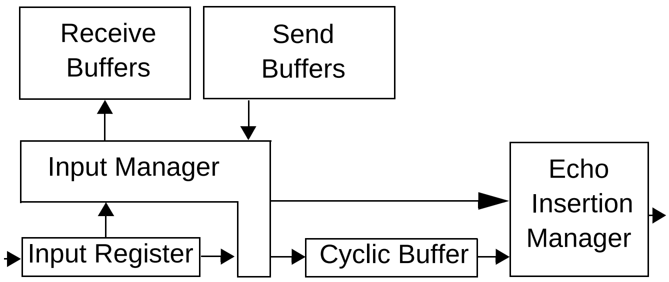

Fig. 2: Implementation view

Frames with data destined to the node are shifted into a Receive Buffer while either reusing the frame to send on data or converting the frame into a transport frame. The Echo Insertion Manager temporarily inserts echo frames into the ring by stopping the read-out of the Cyclic Buffer and generating an echo frame on-the-fly. The Cyclic Buffer has to support parallel read and write operations. Furthermore, the Cyclic Buffer size depends on the number of nodes in the ring because in worst case all nodes send data to one receiver and the receiver has to acknowledge each frame. Therefore the buffer size must be at least n times the size of an echo frame where n is the number of nodes.

IV. Frame Types and Format

The following 8 types of frames are provided in CARRING II: transport, opened transport, init, request, request retry, response, response retry and echo frame. The latter 5 implement the transaction concept that is explained in section 5. Different frame types are indicated by the Type field in the frame setup.

The init frame is used in the ring's start-up phase to generate a time slot for each node, an individual node ID, and to disseminate the chosen data length the ring has to deal with. Echo frames are responsible for the acknowledgement of request and response frames (positive and negative acknowledgements). Since echo frames are only local acknowledges in the same ring that contain no data in the Command and Data field a separate echo frame format exists. Echo frames have the same buffer ID as the corresponding request or response frame but inverse local source and target ID. To identify negative and positive echo frames, the Flow_CTR field is used. In case of a negative acknowledge, an automatic frame retransmission takes place. Transport frames are responsible for bandwidth allocation according to the medium access scheme. Opened transport frames are required for one of the extended medium access schemes called TDMA with open time slots (oTDMA). Such frames have the bit in the Opened Frame field set. All medium access methods are described in section 6. Request and response frames are used to transfer data by utilization of transport frames. To convert a transport frame into a request, response, or retry frame, the Frame type field has to be modified by the Frame Write unit. If the header CRC is invalid, the request or response frame is converted back to a transport frame. The Frame ID can be reconstructed since Frame IDs are sequential. If the receiver has failed, a request or response frame is converted back to a transport frame by the sender after one ring circulation of the frame in the ring. If the sender fails after having sent a request or response frame, the echo of the receiver is consumed after one ring circulation by the receiver.

Payload is only contained in request, response, and retry frames. Transport and init frames have data fields with undefined payload. Init, request, response, retry, and transport frames have the same data length, that is selected during ring initialization. The length remains constant until the next initialization. Although echo frames are not transmitted via the former transport frames, the time slots are preserved since echo frames are not used for TDMA. Frame and echo format are shown in Figure 3.

Each frame begins with a Starting Delimiter that indicates the beginning of a frame by consisting of an out-of-band control code word in 8B/10B coding. A unique frame ID is assigned to each transport frame during ring initialization. This identifier is stored in the Frame ID field and corresponds to the node ID where it was created. The Type field is used to identify the frame type. The Frame Type subfield indicates init, echo, transport, request, response, and retry frames. It is necessary for the medium access scheme to differentiate between transport and opened transport frame by a separate bit. Opened transport frames have the bit in the Opened Frame subfield set. Frame types have already been mentioned above. The Local Target ID is used to send request, response and echo frames to their destinations in the current ring. The Local Source ID contains the sender ID in the current ring. Both address up to 16 nodes in a ring. The Global Target ID contains the ID of the destination node in the target ring. The Global Source ID field stores the node ID of the sender in the source ring and provides the destination address for response frames. A Target Ring ID is used to identify the ring where the destination node is located. The ring ID of the sender is stored in the Source Ring ID. Both are used to route request and response frames to their destinations. The target of request, response, or retry frames can be a node in the whole network while echo frames are just local acknowledgments that operates only in the ring where they are originated. Therefore the frame format (Fig. 3) differs in the number of Target ID and Source ID fields. In request, response, and retry frames the Global Target ID can address up to 16 nodes in the target ring and the additional Target Ring ID can address up to 256 rings in the whole network. Echo frames have only a four bit Local Target ID and Local Source ID to address all nodes in a single ring. A Buffer ID field stores the buffer address of a sent request or response. Together with the Local Target ID, a subsequent echo can be clearly assigned to a sent request or response without the need to compare global ring and node IDs. A TR_Prio field provides transaction priorities. Transaction priority is composed of the most significant bit (MSB TR_Prio) and the least significant bits (LSBs TR_Prio). Frames that have the most significant bit of transaction priority (MSB TR_Prio) set use the overflow receive buffer if the regular receive buffer is full. The Command field contains a transaction command such as read, write or move. The 4-bits TR_ID field indicates which transaction the frame belongs to.

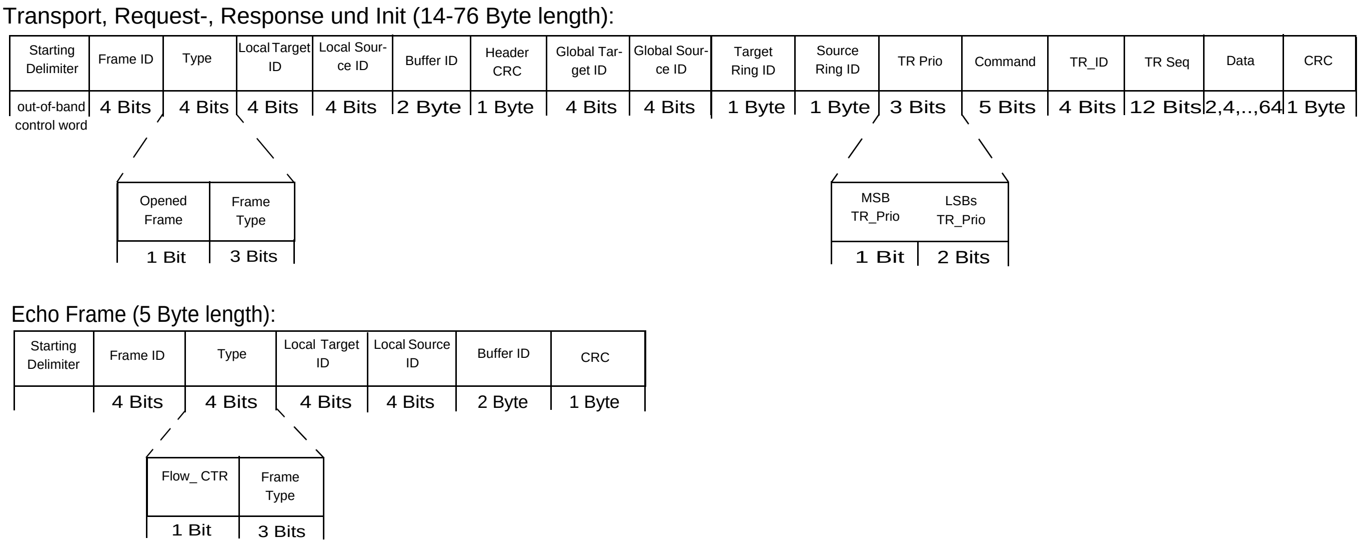

Fig. 3: Frame setup

A transaction can consist of several sequences. The sequence number of a transaction is stored in the TR_Seq field. It is required to reorder sequences of a transaction that may be permuted during the routing. Request, response and retry frames contain up to 64 bytes of payload in its Data field. Header and Data are protected by a 1-byte CRC checksum field.

V. Transactions

CARRING II uses the concept of transactions, i.e. conventional send or receive operations of a LAN are replaced by a handshake protocol called "transaction". The concept was derived from SCI [13] and chosen for reliability reasons. A single transaction is split in time and consists of four phases termed request, request echo, response, and response echo (Fig. 3). For example, if a controller wants to read an input value from a sensor, it sends a request frame with read command and gets a response frame with write command.

A sender starts a transaction by emitting a request frame to a specific receiver. If a request or response frame with correct cyclic redundancy check (CRC) was received and payload could be copied in the receive buffer, a positive echo is sent back to the sender. Otherwise, a negative echo is issued and the frame is rejected. Any sender automatically retries rejected request or response frames to simplify layer 7 software. The number of automatic retransmissions is limited to three for each transaction in order to save bandwidth. After the receiver has processed the request it answers by sending a response frame. Also the response has to be acknowledged positively or negatively. By this handshake mechanism, reliable read and write operations can be performed since each data frame has to be acknowledged.

Each node can have up to 16 pending transactions, i.e. 16 request frames can be issued per node before the first echo must be received. Thus the ring bandwidth is utilized to a maximum. Each transaction can be identified by the TR_ID field contained in the frame format. Furthermore, a transaction can have up to 4096 sequences that allow transactions with more than 64 bytes payload. Transaction sequences are indicated by the TR_Seq field.

The proposed transaction concept has several advantages. First, a handshake protocol with four phases is safer than single read or write operations. Second, the concept provides a full symmetric communication, i.e. every node can read and write to any other node. Third, different phases of different transactions from different senders can be mixed on the same ring at the same time, i.e. the ring is time-shared between many transactions being in different steps of execution. Therefore, no bandwidth is lost if a node is waiting for the response of a slow receiver.

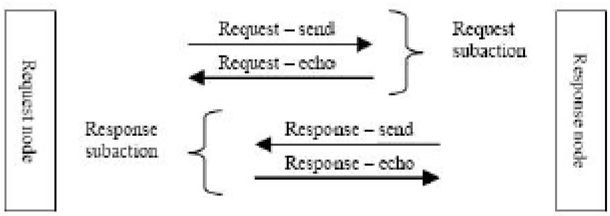

Fig. 4: Transaction concept

VI. Medium Access

CARRING II uses an innovative medium access scheme that is a combination of a variant of Time Division Multiple Access (TDMA) and Register Insertion (RI). TDMA guarantees real-time capability and a fair bandwidth allocation among the nodes. RI was added for better utilization of ring bandwidth. To prepare the medium access with TDMA a initialization procedure has to be executed.

A. Ring Initialization

During the start-up phase, a master node has to be selected among the set of all nodes by assigning node ID 1 to it. All nodes have an unique init ID which is not equal the node ID. To select the master, each node sends its init ID to all other nodes. The node with the highest init ID is selected as master. During operation, no master is required thus avoiding a single point of failure. To initialize the ring the master generates an init frame with its node ID 1 and sends it to the next node in ring. The neighbor node increments the ID of the incoming init frame, takes it as own node ID, and assigns the new ID to the init frame. After that the node creates a new transport frame with its own node ID and appends it to the init frame in order to forward both to the next node. The same procedure repeats until the init frame returns to the master. Finally, the master converts the init frame to a transport frame and assigns its own node ID, i.e. 1, to the new transport frame. During ring operation no init frame appears unless a ring reinitialization is desired.

The initialization procedure has several advantages. First, it has not to be known a priori to the master how many nodes are installed. Second, all transport frames circulate in an ascending order. Third, the variant of TDMA can be applied since each transport frame represents a time slot. Fourth, time slots are created without a global clock. Fifth, there is no single point of failure during normal ring operation since no master node is needed. Last, in case of node fails it is sufficient to reinit the ring to exclude the fault node from the ring because a new initialization only generates time slots for active nodes. A fault node only froward frames through its Bypass FIFO.

B. A Variant of TDMA

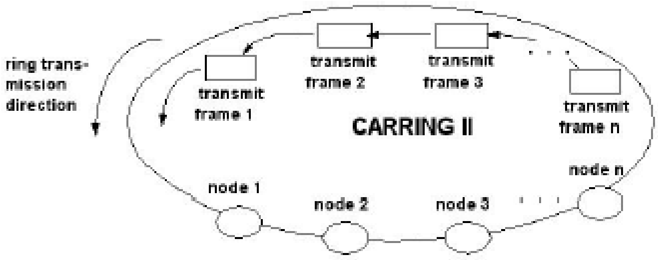

Fig. 5: Time slots after ring initialization

For reliability reasons, CARRING II uses a variant of TDMA that implements the time slots of TDMA not by a clock but by empty data frames called transport frames. Therefore a single point of failure is avoided since no centralized master is needed for clock synchronization and time slot allocation. According to TDMA each node has its own time slot (transport frame). The set of all transport frames is created during ring initialization one after the other. After the initialization phase the sum of all transport frames is equal to the number of active nodes (Fig. 5). Its value is called a TDMA cycle. To achieve a fair bandwidth allocation each node can use its own transport frame to send one data frame per TDMA cycle, i.e. to send one request or response frame. For each node a TDMA cycle begins when a transport frame with frame ID equal to the node ID passes by. The TDMA cycle ends when the same transport frame returns again. In worst case, the frame delivery time takes two TDMA cycles. To send a data frame, the node has to wait until its transport frame arrives. In this case the node can convert the transport frame to a request or response frame in order to send data. The sender updates frame fields (e.g. Target ID and Source ID) while the frame passes the Frame Write register in the node chip. At the receiver the request or response frame is converted back into a transport frame. To detect losses of transport frames each node has a timeout mechanism for its transport frame. If such a timeout occurs the node has to generate a new transport frame.

The advantages of this TDMA variant are that it is not necessary to know how many nodes are installed in the ring because of the initialization procedure, and failed nodes are automatically excluded during ring initialization. Furthermore, with the circulating transport frames the operability of the ring can be checked in every node by timeout. Last, no master is needed to regenerate global clock and assign time slots to nodes.

However, TDMA has also disadvantages. First, if a node wants to send two data frames in series it takes three TDMA cycles in worst case plus the time to send the echo frames by RI. Second, if a node does not want to send data the time slot is lost for one TDMA cycle and thus bandwidth is not efficiently used.

The outlined disadvantages of TDMA can be solved by two extension algorithms that were proven by simulation. The first extension algorithm allows the recycling of unused time slots. This extension is called extended TDMA (eTDMA). In case of eTDMA, a transport frame that is not used by its own node can be reused by other nodes in the same circulation.

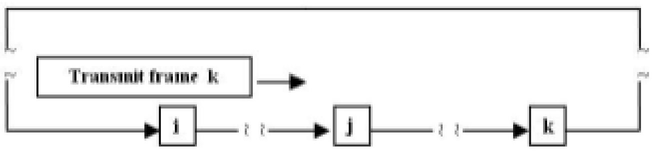

Fig. 6: Reusing time slot k by an i-to-j transfer

To ensure this, special precautions have to be made in the extension algorithm. Assume an transport frame k. The owner of the transport frame k is node k. Now, transport frame k can be reused by nodes i and j if the nodes are located before node k. The transport frame k is converted to a request or response frame by node i and converted back to a transport frame by node j before it arrives at node k. It is mandatory for eTDMA that such an "i-to-j" transfer exists (Fig. 6). The condition, that the i-to-j transfer takes place before node k is reached, is necessary because otherwise node k could not use its transport frame for the own data transfer. Since a ring topology is used, it is unclear whether i is before j and j before k. This problem is solved by the extension algorithm that decides if i and j are located before k.

It has to be noted that the recycling of transport frames can not be employed for real-time transfers since it is not guaranteed that a recycling is always possible. But an advantage is that ring bandwidth may be exploited more efficient.

The disadvantage of eTDMA is that if no i-to-j transfer exists transport frames can not be reused by other nodes and bandwidth is still lost. The second extension algorithm tries to solve this problem by opening its time slot to any other node for an arbitrary period of time. Open time slots are defined as opened transport frames and indicated by the Opened Frame field in the frame layout. All nodes can use an opened transport frame to send data. This extension is called TDMA with open time slots (oTDMA). It provides a simple and cost-effective way to allocate bandwidth since the extension algorithm from eTDMA is not necessary for an i-to-j transfer. With oTDMA, the owner of a transport frame converts the transport frame to an opened transport frame if its send buffer is empty. Otherwise, if there is at least one data frame in the send buffer the node converts the opened transport frame back to a transport frame by resetting the Opened Frame bit. To send a data frame, it is not necessary that the node has to wait until its own transport frame arrives if an opened transport frame is available. In worst case, a node has to wait one TDMA cycle to use its own transport frame if its opened transport frame was already used by another node in the last cycle. The node has to reset the Opened Frame bit, i.e. to convert back the own opened transport frame to a transport frame, so that no other node can use it in the next circulation. Therefore, real-time is still maintained.

The oTDMA algorithm has the following advantages. First, it is compatible to eTDMA, so that oTDMA can be combined with eTDMA giving oeTDMA. Second, opened transport frames can be reused several times in one circulation. Third, synchronous data transfer can be completely reverted into asynchronous transfer mode by opening all transport frames. In completely asynchronous mode all bandwidth can be utilized by 1 sender for a high-speed transfer.

C. Register Insertion

With Register Insertion (RI) echo frames will be injected in the ring independently from transport frames. RI was implemented for performance reasons since echo frames have only a length of 5 bytes but transport, request, and response frames can have up to 76 bytes length. The usage of transport frames to send echo frames would result in an overhead of up to 71 bytes. RI guarantees an efficient bandwidth utilization since only as much data as needed is transmitted.

Echo frames are inserted into the ring by the Echo Insert/Remove unit that is a part of the node chip. It consists of an Echo Manager and a register that is implemented as circular buffer. The size of the circular buffer corresponds to the number of nodes in the ring since in worst case all nodes send data frames to one receiver and each data frame hast to be acknowledged by an echo. Therefore, each node can extend the ring n times the size of an echo frame where n is the number of nodes in the ring. An echo is temporarily inserted into the ring by waiting the shiftout of the next frame that has to be sent and inserting the echo. Therefore, the ring is extended by inserting register buffer space. It is inserted into ring to hold any other frame until the echo frame was sent out. The destination of an echo frame has to check if the acknowledge is positive or negative. After one circulation the echo frame is removed by not shifting it into the buffer and thus freeing the buffer space.

VII. Buffer Management and Reservation

Various buffers in the node chip requires a careful buffer management. Split send and receive buffers and different buffer classes for request and response frames are implemented for performance reasons and to avoid deadlocks. The node chip setup contains a receive buffer, a send buffer, a retry buffer, and an additional overflow receive buffer as depicted in Fig. 1.

A received request or response frame is stored in the receive buffer. After forwarding the stored request or response from receive buffer to the attachment the receive buffer place is released by the buffer management. Payload from the attachments is stored in the send buffer as request or response in order to send it to a dedicated receiver. Immediately after such a request or response was sent out, the send buffer place is released. Additionally, the sent frame is stored in the retry buffer for frame retransmission. After receiving a positive echo, the retry buffer place is released. The overflow receive buffer is used if the receive buffer is full. Its setup is identical to the receive buffer.

The buffer reservation mechanism was implemented to reduce buffer overflows in case of data bursts. To realize the reservation scheme an overflow receive buffer was implemented in CARRING II. There are two ways to use this buffer. First, it can be used by frames with high priority because such frames contain important payload that has to be processed as soon as possible. Second, it can be used by retry frames while such frames additionally occupy ring bandwidth. The utilization of the overflow receive buffer is only possible if the regular receive buffer is full.

All frames that belong to a transaction which have the MSB TR_Prio bit in the frame format set are high priority frames. According to the transaction concept, all frames of such transactions have a high priority, i.e. if the request frame of a transaction has a high priority the corresponding response frame has also a high priority. High priority frames are stored in the overflow receive buffer only if this buffer is not full or reserved but the regular buffer is already full. As soon as the regular buffer becomes free the overload receive buffer is no longer required and therefore no longer used.

Request and response frames with low priority are negatively acknowledged if the regular receive buffer is full. As a consequence, a negative echo is sent back to the sender. For each negative echo, the node tries to reserve a place in its overflow receive buffer for the subsequent retry. A reservation is possible if the regular receive buffer is full and the overflow receive buffer has a free place that is not already reserved by another transaction. Only retry frames have the possibility to reserve a buffer place in the overflow receive buffer. Then, an arriving retry frame first clears an existing reservation in the overflow receive buffer and afterwards it is stored in the overflow receive buffer only if the receive buffer is full. Otherwise it is stored in the receive buffer.

With the proposed scheme the probability of acceptance of retry frames or high priority frames at the receiver is increased.

VIII. Timeout Concept

For each transaction, a node has to store and check 2 timeoutvalues. These values are relative to the node's local clock.

The first timeout detects lost echo frames, i.e. it checks if the acknowledge of a request or response is not present in time. The second timeout detects lost response frames that were sent as an answer of a request. If one of these timeouts are activated the transaction terminates, an error code in the status register of the node chip is set, and one of the attachments, e.g. the attached controller, is interrupted. Then, the status register is read by the attachment's interrupt service routine to investigate the reason for the interrupt. Subsequently, the attachment may put a frame with an error message onto the ring to inform other nodes. It is possible to read and write timer memory and status register via the attachment bus or via the ring. This supports local and global node chip handling. Additionally, dual access is allowed for nodes without local attachments. This is required for switches and hubs. Their error handling takes place via the ring. After detecting a chip error, the chip may be reinitialized locally or globally.
An additional TDMA timeout checks if the time slot, i.e. the transport frame, is not present in time. If a transport frame is lost the node has to generate a new transport frame.
Altogether, 33 timeout values are required per node because a node can have up to 16 pending transactions with 2 timeout values each plus one TDMA timeout value.

IX. Conclusion and Future Work

A reliable, real-time, high-speed local-area network called CARRING II was developed that is based on a ring topology. It is scalable and cost-effective. CARRING II is a further development of the older CARRING communication network. Applications for CARRING II are e.g. the connection of mechatronic systems in cars. For CARRING II, a new node setup and frame format, and a new buffer receiver reservation scheme together with proper transaction priority handling were introduced . The proposed medium access scheme is a combination of TDMA and Register Insertion that guarantees a fair and efficient bandwidth allocation. CARRING II is stable, predictable and real-time capable even under overload conditions. Simulation results are presented in future publications. Future works deal with performance comparisons and routing algorithms.

References

[1] Robert Bosch GmbH, *CAN bus specification*, ISO std. 11898/11519 http://www.can.bosch.com, 2005.

[2] LIN Consortium, *LIN bus specification*, http://www.lin-subbus.org, 2005.

[3] MOST Cooperation, *MOST bus specification*, http://www.mostnet.de, 2005.

[4] Flexray Consortium, *Flexray specification*, http://www.flexray-group.org, 2005.

[5] H. Richter, M. Wille, A. Guerrouat, C. Siemers, U. Konigorski and M. Vossiek, *A concept for a reliable, cost effective, real time Local Area Network for automobiles*, Proc. Joint conference Embedded in Munich and Embedded Systems, Munich, November 09-11, 2004.

[6] H. Richter and M. Wille, *Efficient Bandwidth Allocation in a Ring-Based Real- Time Network for Automobiles*, Poster Proc. 12th IEEE International Symposium on Modeling, Analysis and Simulation of Computer and Telecommunication Systems (MASCOTS), Volendam (NL), October 4-8, 2004.

[7] H. Richter and M. Wille, *A High-Performance Local Area Network for Cars*, Proc. 3rd IASTED International Conference on Communications, Internet, and Information Technology (CIIT), St. Thomas - US Virgin Islands, November 22-24, 2004.

[8] M. Wille and H. Richter, *Carrings Medium Access Methods in Comparison to FDDI, Proc. 2005 IEEE Intelligent Vehicles Symposium (IV'05),* Las Vegas (USA), June 6-8, 2005.

[9] Cypress Semiconductor Corporation, *CYP15G0101DXB Single-channel HOTLink II Transceiver (Data Sheet)*, San Jose, CA 95134 USA : http://www.cypress.com, 2005.

[10] Xilinx, *8b/10b Encoder/Decoder - product specification*, San Jose, CA 95124-3400 USA : http://www.xilinx.com, 2005.

[11] A. Tannebaum, *Computernetzwerke*, 4th ed., Munich: Pearson Studium, 2003.

[12] IEEE, *Standard for Scalable Coherent Interface (SCI)*, IEEE Std. 1596-1992, New York, 1993.

[13] W. Proebster, *Rechnernetze: Technik, Protokolle, Systeme, Anwendugen*, Munich: Oldenbourg Verlag, 1998.