

TU Dortmund
Fakultät Informatik

Ausarbeitung zum Seminarthema FlexRay im Rahmen des
Seminarteils der Projektgruppe AutoLab



Christian Horn (111100)

Betreuer:

Prof. Dr.-Ing. Olaf Spinczyk
Horst Schirmeier
Jochen Streicher

Inhaltsverzeichnis

Motivation	3
Kommunikation der Knoten	3
Aufbau eines Knotens	6
Verwendung des Busses	7
Synchronisation der Knoten	9
Fehlertoleranz	10
Fazit	11
Literaturverzeichnis	12
Abbildungsverzeichnis	13

Motivation

In der Automobilindustrie werden derzeit viele unterschiedliche Bussysteme eingesetzt. Zu den bekanntesten und verbreitetsten zählen CAN, LIN und MOST. Während der MOST-Bus insbesondere für Multimediaanwendungen konzipiert wurde, werden CAN-Busse und LIN-Busse für die Steuerung der Fahrzeugelektronik eingesetzt. Beim Entwurf neuer Fahrzeugmodelle kommen immer mehr Mikrocontroller zum Einsatz. CAN und LIN sind dieser immer weiter steigenden Zahl an Knoten innerhalb des Fahrzeugnetzes nur bedingt gewachsen. Einerseits ist die Bandbreite beider Busse (LIN 20 Kbit/s und CAN ≤ 1 Mbit/s) im Vergleich zu FlexRay (2x 10 Mbit/s) deutlich geringer, andererseits bieten sie nur eine begrenzte Echtzeitfähigkeit, die sie insbesondere für sicherheitskritische Aufgaben disqualifiziert. FlexRay wird vom FlexRay-Konsortium entwickelt, einem Zusammenschluss von mittlerweile über 125 Mitgliedern in und um die Automobilbranche herum. Das Ziel der Entwicklung ist ein leistungsfähiger, fehlertoleranter, echtzeitfähiger Bus für die Kommunikation in Automobilen.

Kommunikation der Knoten

Jeder Knoten an einem FlexRay Bus kann Daten senden und empfangen. Das Empfangen der Daten ist durch permanentes Lauschen auf dem Bus möglich. Der Knoten erhält alle Informationen, die über den Bus laufen und greift diejenigen ab, die er benötigt, um diese weiter zu verarbeiten oder andere Aktionen durchzuführen.

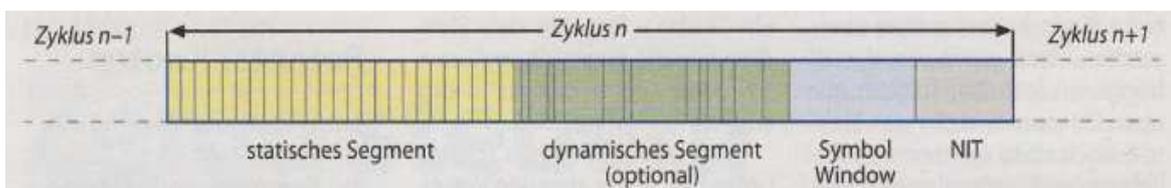


Abb.1: Aufbau eines Zyklus

Das Senden ist etwas komplizierter. Die Kommunikation beruht auf Zyklen. Jeder Zyklus gliedert sich in vier Segmente: das statische Segment, das dynamische Segment, Symbol Window und NIT (Network Idle Time).

Das statische Segment wird während des Entwurfs in eine bestimmte Anzahl an gleich großen Slots unterteilt. Diese Slots werden den Knoten zugewiesen. Während des statischen Segments kann nur derjenige Knoten etwas senden, dessen Slot-ID gerade gültig ist. Das statische Segment wirkt unflexibel, garantiert aber die Echtzeitfähigkeit, die FlexRay ausmacht. Durch diesen exklusiven Zugriff wird (weitgehend) Kollisionsfreiheit gewährt. Das statische Segment ist ausschließlich zeitgesteuert. Nicht genutzte Slots verfallen somit, ohne dass andere Knoten, die etwas senden möchten, dies können. Das statische Segment ist also insbesondere wichtig für echtzeitkritische Aspekte. So können Verzögerungen beim Bremsen den entscheidenden Unterschied zwischen Unfall und rechtzeitigem Anhalten ausmachen.

Nicht alle Knoten haben kritische Aufgaben. Um unregelmäßigen Sendewünschen der Knoten nachzukommen und eine begrenzte Ereignissteuerung zu ermöglichen, gibt es das dynamische Segment. Es ist zum Beispiel beim Verstellen des Sitzes unerheblich, ob dies unmittelbar oder mit einer Verzögerung im Millisekundenbereich geschieht. Das dynamische Segment ist für derartige Bereiche und Knoten, die nicht zyklisch abgefragt werden müssen, geeignet. Die Slots werden nahtlos weiter durchnummeriert. Ihre Größe, und damit die Länge der Nachricht, ist jedoch variabel. Die Gesamtlänge des dynamischen Segments hingegen muss aber beim Entwurf festgelegt werden, da sonst nicht garantiert werden kann, zu welchem Zeitpunkt das statische Segment wieder auftritt. Wäre eine garantierte Vorhersagbarkeit für das statische Segment nicht möglich, würde die Echtzeitfähigkeit verloren gehen. Nicht jeder Knoten kann in jedem dynamischen Segment eine Nachricht auf den Bus legen. Um Fairness zu garantieren, verwendet man so genanntes Cycle-Multiplexing. Beim Cycle-Multiplexing wird garantiert, dass jeder Knoten, der etwas im dynamischen Segment senden möchte, dies auch tun kann. Die Slot-Ids werden umverteilt, damit jeder Knoten die Möglichkeit hat, eine so kleine ID zu bekommen, dass sein Slot auch garantiert in dem dynamischen Segment in einem Zyklus vorkommt. Somit sind zwar Verzögerungen möglich, da ein Knoten auf seine Sendeerlaubnis einige Zyklen warten muss, jedoch ist kein unendliches Warten möglich.

Symbol Window ist ein festgelegter Abschnitt in jedem Zyklus, der zur Übertragung von internen Steuerinformationen dient. Nutzdaten werden hier nicht übertragen. Die Länge

muss beim Entwurf festgelegt werden, da sonst die Länge der Zyklen nicht vorhersagbar wäre. Damit wäre der Start des nächsten statischen Segments nicht gewährleistet und die Echtzeitfähigkeit nicht mehr gegeben.

NIT ist ein kleines Segment am Ende jedes Zykluses, das zur Anpassung der lokalen Uhr an die globale Zeit dient. Das Segment ist für die Synchronisation notwendig. Auch dieses Segment muss eine feste Länge haben, da ansonsten, wie auch beim dynamischen Segment und Symbol Window, die Länge jedes Zykluses abweichen könnte und somit die Echtzeitfähigkeit nicht mehr gewährleistet werden könnte.

Aufbau eines Knotens:

Jeder FlexRay-Knoten besitzt einen Host-Controller, einen FlexRay Communication Controller, zwei Bus Guardians und zwei Bus Driver, sowie die entsprechenden Leitungen zur Verbindung der Bausteine und die Anbindung an die Kanäle des FlexRay-Busses. Weiterhin besitzen die Knoten in der Regel zusätzlich noch einen Mikrocontrollerkern für Anwendungen unter einem Echtzeitbetriebssystem, wie zum Beispiel OSEK-Time.

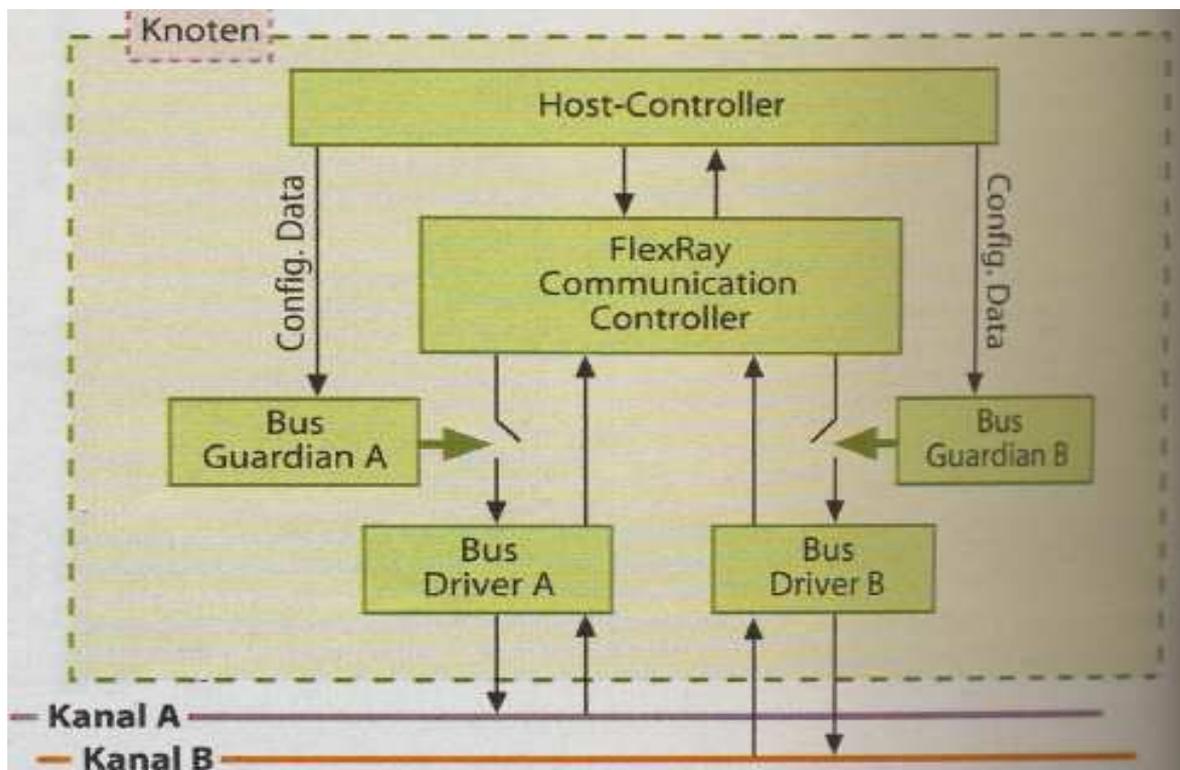


Abb.2: Aufbau eines Knotens

Der Hostcontroller verarbeitet, gegebenenfalls im Zusammenspiel mit dem Mikrocontrollerkern, Daten, die über den FlexRay Communication Controller übertragen werden.

Der FlexRay Communication Controller dient der Umsetzung des Busprotokolls in jedem Knoten. Er empfängt die Daten des Busses und der ausgeführten Berechnungen, um sie gegebenenfalls in entsprechende Datenpakete umzupacken und weiterzuleiten.

Die Bus Guardians garantieren, dass die FlexRay Communication Controller der Knoten nur dann Daten senden können, wenn ihre Slot-ID gültig ist, damit es nicht zu Kollisionen kommt. Ihre Informationen erhalten sie vom Hostcontroller.

Die Bus Driver lauschen permanent auf dem Bus, um Informationen, die der Knoten benötigt, mitzubekommen und weiterzureichen. Beim Senden von Daten ihres Knotens leiten sie diese an den Bus weiter.

Verwendung des Busses

FlexRay bietet verschiedene Möglichkeiten zur Verwendung des Busses. Einerseits ist es möglich, verschiedene Topologien zu erstellen und diese sogar zu kombinieren; andererseits können die beiden Kanäle verschiedene Aufgaben, je nach Erfordernissen, übernehmen.

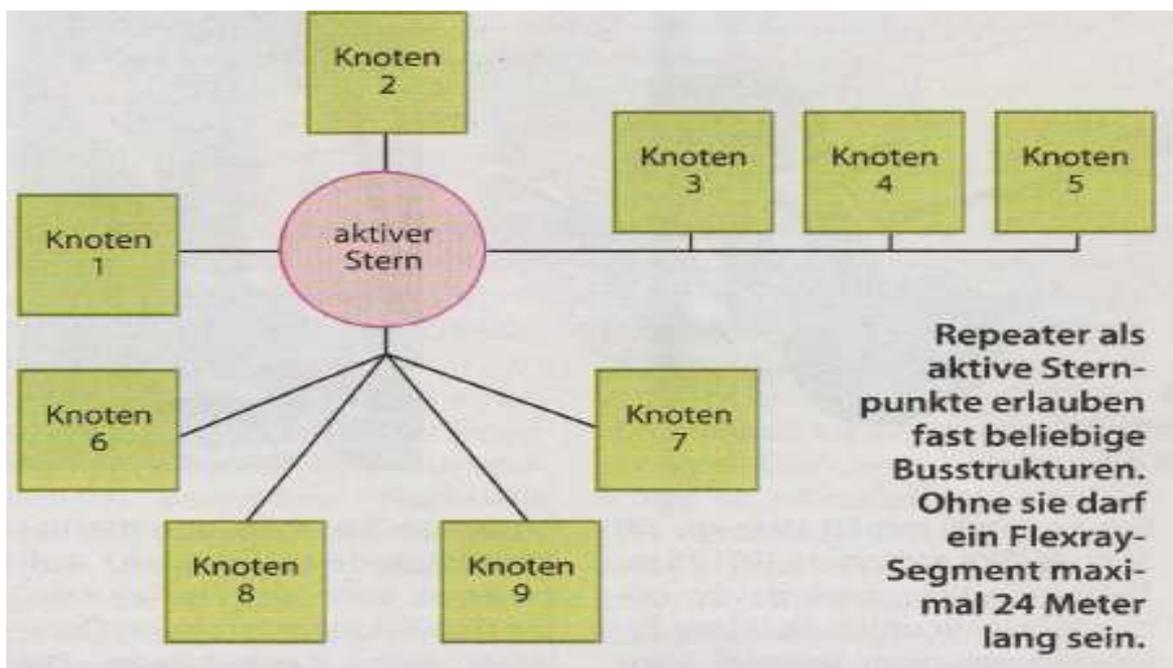


Abb.3: Topologien

In der Abbildung ist ein aktiver Stern (Repeater) zu sehen, von dem aus mehrere Stränge ausgehen. Diese Anordnung der Elemente heißt Sterntopologie. Weiterhin ist es auch möglich, wie im rechten Teil der Abbildung zu sehen, FlexRay als linearen Bus zu verwenden. FlexRay bietet somit die Möglichkeit, das Knotennetzwerk optimal an die Bedürfnisse des Designers und der vorgegebenen räumlichen Einschränkungen anzupassen.

Da FlexRay über zwei Kanäle verfügt, gibt es auch hier verschiedene Möglichkeiten diese zu nutzen. Dies ist jedoch nur für die Knoten möglich, die auch an beide Kanäle angeschlossen sind.

Besonders interessant für sicherheitskritische Systeme ist die redundante Datenübermittlung. Die entsprechenden Daten werden identisch durch beide Kanäle gleichzeitig gesendet. Durch dieses Vorgehen wird eine höhere Sicherheit der korrekten Datenübermittlung gewährleistet. Unterbrechungen und Störungen, die nur einen Kanal betreffen, können somit vermieden werden. Durch die doppelte Übertragung halbiert sich aber auch die Übertragungsrate von 2×10 Mbits/sec auf 1×10 Mbits/sec.

Die Gleichschaltung verfolgt genau das gegenteilige Ziel. Die Daten, die ein Knoten während seines ID-Slots verschickt, werden auf die beiden Kanäle aufgeteilt, um die maximale Bandbreite möglichst gut auszulasten. Dies ist insbesondere für Knoten mit hohem Datenaufkommen interessant. Hierbei wird jedoch die Sicherheit der redundanten Auslegung nicht gewährleistet. Weiterhin müssen auch alle Knoten, die die Daten benötigen, ebenfalls an beide Kanäle angeschlossen sein, da sie sonst nur unvollständige Datensätze erhalten.

Wie bereits in der Einführung erwähnt, bietet FlexRay gegenüber CAN und LIN eine deutlich höhere Bandbreite. Sofern die entsprechenden Knoten eine FlexRay-Anbindung unterstützen, ist es problemlos möglich, CAN und LIN Busse durch FlexRay zu ersetzen. Hierbei ist jedoch zu beachten, dass die Kosten von FlexRay und der dazu kompatiblen Hardware deutlich über denen von CAN und LIN liegen. Eine Ersetzung kann trotzdem sinnvoll sein, wenn entweder die Übertragungsraten der Knoten bereits an die Grenzen der alten Busse stoßen, oder aber mehrere Stränge alter Busse gleichzeitig durch einen FlexRay Bus ersetzbar sind.

Auf Grund der Möglichkeiten, Topologien flexibel anzupassen und zu kombinieren und der verhältnismäßig hohen Bandbreite, besteht zusätzlich die Option FlexRay als Backbone für andere Busse einzusetzen.

Synchronisation der Knoten

Eine der grundlegendsten Eigenschaften von FlexRay ist die Echtzeitfähigkeit. Damit das gesamte System jedoch ein Echtzeitsystem bildet, benötigen die Knoten ein gemeinsames Verständnis der globalen Zeit. Jeder Knoten besitzt einen Oszillator. Zu Beginn, zum Beispiel bei Hochfahren des Systems, kennt jeder Knoten nur seine lokale Uhr. Auf Grund der fertigungsbedingten Abweichungen der Nominalfrequenz ist es nicht möglich, dass alle Knoten unmittelbar ein Verständnis der globalen Zeit haben und dieses auch halten können. Weiterhin unterliegen die Knoten äußeren Einflüssen, wie beispielsweise der Temperatur, die sich unterschiedlich auf die Knoten auswirkt. Um trotzdem Echtzeitfähigkeit zu gewährleisten, ist eine Synchronisation der Knoten notwendig.

Start-Up-Phase

Während des Hochfahrens des Systems wecken die Knoten einander durch das Senden bestimmter Zeichenketten (Symbole) auf. Sind alle funktionsfähigen Knoten aufgeweckt, senden die sogenannten Sync-Knoten Sync-Frames in ihren Slots. Die Sync-Knoten sind normale Knoten, die zusätzlich zu ihren normalen Daten jedes Mal in den ihnen zugewiesenen Slots ein Sync-Frame mitschicken. Alle Knoten, insbesondere auch andere Sync-Knoten, vergleichen daraufhin die erwartete Ankunft des Frames mit seiner tatsächlichen Ankunft, um gegebenenfalls Zeitunterschiede ausgleichen zu können. Die Einpendelung auf eine gemeinsame Zeit erfolgt während weniger Zyklen.

Für das gemeinsame Verständnis der globalen Zeit sind zwei Aspekte entscheidend: Zum einen müssen die lokalen Zeiten aller Knoten zu einem bestimmten Zeitpunkt (nahezu) identisch sein, zum anderen muss gewährleistet werden, dass die Knoten nicht unterschiedlich schnell fortschreiten in der Zeit. Nach der Start-Up-Phase haben alle Knoten (kurzzeitig) eine gemeinsame Zeit. Auf Grund der bereits erwähnten fertigungsbedingten Unterschiede der Frequenz können sie diese gemeinsame Zeit jedoch nicht beibehalten. Da die Knoten nicht in der Lage sind, die Frequenz ihrer Oszillatoren zu ändern, muss die Korrektur indirekt erfolgen. Man unterscheidet zwei Arten der Synchronisation: Offset Correction und Rate Correction.

Unter der Offset Correction versteht man das Einstellen einer bestimmten Zeit. Dies geschieht, indem die Sync-Knoten weiterhin in jedem ihrer Slots das Sync-Frame mitschicken. Wie auch in der anfänglichen Start-Up-Phase wird jeweils die tatsächliche mit der erwarteten Ankunft der Sync-Frames verglichen. Anhand der Werte der Abweichungen wird der FTM (Fault Tolerant Midpoint) – Algorithmus gestartet. Je nach Anzahl der Sync-Knoten innerhalb des Systems werden gegebenenfalls Extremwerte verworfen, um nicht vom Abdriften einzelner Knoten bzw. Teilsysteme betroffen zu sein. Sind mehr als zwei Sync-Knoten aktiv, werden die Extremwerte verworfen, bei mehr als sieben Sync-Knoten zusätzlich auch die zweitgrößten/-kleinsten Werte. Die verbliebenen Werte werden gemittelt und bilden die Abweichung der lokalen Zeit zur globalen Zeit. Der entsprechende Wert wird dann übernommen, so dass die Knoten wieder kurzzeitig in einen Zustand des Konsenses kommen. Die Anpassung wird während der NIT eines Segmentes vorgenommen. Somit kommt es nicht zu Sprüngen während der statischen oder dynamischen Segmente.

Um die Korrekturen zu minimieren, dass die Knoten weitgehend im Gleichschritt laufen, müsste man die Frequenzabweichung korrigieren. Dies ist, wie bereits erwähnt, nicht unmittelbar möglich. Die Korrektur erfolgt indirekt, in dem jeder Knoten die Zeitstempel zweier aufeinander folgender Zyklen misst und wieder mittels FTM-Algorithmuses mittelt. Dies nennt man Rate Correction. Durch das Ergebnis weiß jeder Knoten, ob sein Oszillator zu schnell oder langsam im Vergleich zu den anderen Knoten läuft. Dadurch ist er in der Lage, in künftigen Zyklen gleichverteilt Microticks hinzuzufügen bzw. wegzulassen, um seine Zyklenlänge anzupassen.

Beide Arten von Synchronisation sind immer wieder erforderlich. Es ist nicht ausreichend, ein synchron laufendes System einmal zu erreichen, da dieser Zustand nicht stabil wäre.

Fehlertoleranz

Die Fehlertoleranz, die FlexRay bietet, teilt sich in zwei Gebiete auf. Zum einen besitzt FlexRay eine hohe Verfügbarkeit, andererseits einen speziellen Start-Up-Mechanismus.

Fällt bei redundanter Auslegung ein Kanal aus, bleibt das System vollständig verfügbar, da der andere Kanal ausreichend ist, um alle Informationen unter Berücksichtigung der oben genannten Einschränkungen, sicher weiter zu leiten. Fallen Knoten oder ganze Teilsysteme aus, so bleiben alle weiteren Teilsysteme, die keine Informationen von den defekten Knoten benötigen, vollständig intakt. Besonders wichtig für FlexRay ist die Echtzeitfähigkeit. Dafür müssen, wie bereits erwähnt, die Knoten ein gemeinsames Verständnis der globalen Zeit haben. Die globale Zeit kann jedoch durchaus von der realen Zeit abweichen und das ganze System wegdriften, ohne Funktionalität einzubüßen. Weichen einzelne Knoten oder Teilsysteme jedoch durch Defekte ihrer Oszillatoren stark von der globalen Zeit innerhalb des Systems ab, verpassen diese Knoten oder Teilsysteme ihre Slots. Man spricht hierbei von einem Wegdriften eines Teilsystems. In diesem Fall bleiben andere Teilsysteme unberührt von der Drift und sind in der Lage, weiterhin ordnungsgemäß zu kommunizieren und zu arbeiten. Der andere Aspekt ist der Start-Up-Mechanismus, bei dem sich die Knoten gegenseitig aufwecken und somit aktivieren. Alle Knoten, auch die anderen Sync-Knoten, orientieren sich an diesen Frames zur Erlangung eines gemeinsamen Verständnisses der globalen Zeit. Durch die Festlegung der Knoten während des Entwurfs wird Kollisionsfreiheit für die Sync-Frames garantiert.

Fazit

FlexRay ist ein innovatives Bussystem, das sich in der Zukunft mehr und mehr durchsetzen wird. Trotz all seiner Vorzüge wird es die älteren Bussysteme jedoch nicht in absehbarer Zeit ablösen, da die Kosten für Steuergeräte und Leitungen deutlich höher sind. Im Bereich der heute nur theoretisch realisierbaren By-wire-systems, beispielsweise break-by-wire oder steer-by-wire, wird zukünftig nur FlexRay in der Lage sein, das hohe Maß an Sicherheit und Bandbreite zu bieten, welches für diese sensiblen Bereiche erforderlich ist. Gemäß der FlexRay-Spezifikation ist FlexRay so konzipiert, dass auch der Einsatz von Lichtwellenleitern statt der derzeit verwendeten Zweidrahtleitung vorgesehen ist. Da heute jedoch noch keine entsprechenden Lichtwellenleiter verfügbar sind, lässt sich nur schwer prognostizieren, in wie weit ihr Einsatz eine Veränderung bei den Kosten und der Leistung bewirken wird.

Literaturverzeichnis

<http://flexray.com/>

<http://www.tzm.de/flexray>

<http://de.wikipedia.org/wiki/flexray>

M. Jochim, „Zeitig steuern – sichere Datenübertragung im Automobil“, c't 2/2007, S. 190 - 195

Abbildungsverzeichnis

Abb.1: Aufbau eines Zyklus → M. Jochim, „Zeitig steuern – sichere Datenübertragung im Automobil“, c't 2/2007, S. 192

Abb.2: Aufbau eines Knotens → M. Jochim, „Zeitig steuern – sichere Datenübertragung im Automobil“, c't 2/2007, S. 194

Abb.3: Topologien → M. Jochim, „Zeitig steuern – sichere Datenübertragung im Automobil“, c't 2/2007, S. 194