LEHRSTUHL FÜR REALZEIT-COMPUTERSYSTEME
TECHNISCHE UNIVERSITÄT MÜNCHEN
UNIV.-PROF. DR.-ING. G. FÄRBER

Realzeit in der Praxis

Hauptseminar Realzeit-Computersysteme
Wintersemester 2003/2004

Seminarband

Realzeit in der Praxis

Seminarband

Ausgeführt am Lehrstuhl für Realzeit-Computersysteme
Technische Universität München
Prof. Dr.-Ing. Georg Färber

Hauptseminar Realzeit-Computersysteme
Wintersemester 2003/2004

Inhaltsverzeichnis

Einleitung

Dieser Seminarband entstand im Rahmen der Lehrveranstaltung „Hauptseminar Prozeßrechentechnik“ im Wintersemester 2004. Die Arbeiten erscheinen inhaltlich so, wie sie von den Studenten abgegeben worden sind.

Kapitel 1

LEGO-Mindstorms

Autor: Raphael Schopp

Betreuer: Dipl.-Ing. Jürgen Stohr

1.1 Einführung

Die LEGO-Gruppe enwickelte in Zusammenarbeit mit dem Massachusetts Institute of Technology (MIT) 1998 das Robotics Invention System (RIS). Der RIS-Baukasten wird für die Altersgruppe ab 12 Jahre ohne Programmiererfahrung empfohlen. Im Rahmen der LEGO-Technik Weiterentwicklung erschien zuerst der CyberMaster, der dem RCX nahe verwandt ist und über Funk kontrolliert wird. Im CyberMaster bilden Aktoren und Computer eine Einheit.
1999 folgte das Droid Developer Kit, mit dem MicroScout, einem Modul mit Motor, Licht-Sensor und Tongenerator, mit dem Star Wars Modelle gebaut werden sollen. Das Kit enthält eine CD mit einer multimedialen Einführung in das System, benötigt aber zum Betrieb keinen PC.
Ebenfalls 1999 erschien das Robotics Discovery Kit mit dem Scout, der für 9-jährige aufwärts gedacht ist und über eine Infrarot-Fernbedienung beeinflußt wird.Vor kurzem erschien bei LEGO im Internet ein Development Kit zum Scout, mit dem er vom PC aus im sogenannten LEGO Assembler programmiert werden kann. Der Scout verwendet eine ähnliche Peripherie wie der RCX, seine Software ist aber deutlich eingeschränkt. Zentraler Bestandteil des RIS ist ein Hitachi H8 Microcomputer in einem größeren LEGO-Baustein, der sogenannte Programmable Brick oder RCX, zu Beginn in der Version 1.0. Der RCX wird über eine serielle Schnittstelle und den sogenannten Infrared Tower in der Regel von einem PC aus kontrolliert und programmiert. Normalerweise lädt man Programme auf den RCX und betreibt sie dann autonom, deshalb eignet sich das System vorwiegend zum Bau mobiler LEGO-Roboter. Die Kosten für ein Anfangsset betragen ca. 200 Euro.

Abb.: Lego Mindstorms Robotics Invention System 2.0 [1]

[1]Produktbild eines Angebots bei www.ebay.de

1.2 Der RCX-Baustein als didaktisches Hilfsmittel

LEGO Mindstorms ist keineswegs wie der Hersteller vermuten lässt ausschließlich in Kinderzimmern zu finden. Auch Fachakademien und Universitäten nutzen den RCX-Baustein, mit dessen Hilfe sich Studenten Fachkenntnisse erschließen können. Neben dem Vermitteln der Programmiergrundlagen (Schleifen, Prozeduren, objektorientierte Methoden) sowie der Algorithmenoptimierung eignet sich LEGO Mindstorms zum Veranschaulichen von eingebettete Systemen. Das Verhalten konkurrierender Systemoperationen kann dank schneller Montage direkt am „echten“ Roboter getestet werden. In diesem Zusammenhang sei auch die Programmkorrektur erwähnt, die bei unerwünschtem Systemverhalten unabdingbar ist. Des Weiteren können Studenten mit dem RCX die netzzentrierte Steuerung (periphere und zentralisierte Steuerung) evaluieren und optimale Lösungen suchen. Da bei allen Praktika in Gruppen gearbeitet wird, gibt LEGO Mindstorm außerhalb der direkten fachlichen Qualifikation die Möglichkeit Teamfähigkeit zu erwerben und zu trainieren.

1.3 Technische Daten des RCX

Der RCX Baustein besitzt neben einem LCD-Display vier Taster mit den Funktionen Ein-/Ausschalten, Programmwahl, Programmansehen und Programmstart. Das Herzstück des Bausteins bildet ein Hitachi H8/3292 Mikrokontroller mit einer Taktfrequenz von 16 Mhz, 16 kB ROM und 28 kB RAM. Die Kommunikation mit einem PC oder zusätzlichen RCX ermöglicht eine serielle Infrarotschnittstelle. Betrieben wird der Baustein mit sechs in Reihe geschaltenen AA-Batterien, sodass die Betriebsspannung annähernd 9 V beträgt. Die Funktionsdauer beträgt bei Akkus ein bis zwei Stunden, bei konventionellen Batterien zwei bis drei Stunden.

Abb.: RCX Baustein

1.3.1 Hitachi H8/3292 Mikrokontroller

Die H8/300 CPU ist eine schnelle CPU mit acht 16-bit Registern, die ebenfalls als 16 achtbit Register konfiguriert werden können und besitzt einen übersichtlichen Befehlssatz für Hochgeschwindigkeitsoperationen. Dieser beinhaltet 57 Grundbefehle, Multiplikation und Division sowie Anweisungen für bitweise Verarbeitung inbegriffen. Die on-chip Supportingmodule implementieren periphere Funktionen, die in der Systemkonfiguration benötigt werden. Diese schließen neben den bereits erwähnten ROM und RAM drei Timerarten (16-bit free-running timer, 8-bit timer, watchdog timer), eine serielle Kommunikationsschnittstelle (SCI) einen A/D-Wandler sowie I/O ports ein.

1.3.2 Schnittstellen

Der RCX besitzt neben der bereits erwähnten Infrarotschnittstelle sechs weitere Schnittstellen für die Kommunikation mit Sensoren und Aktoren

Abb.: Der RCX 2.0 Baustein mit Schnittstellen

Eingänge

An die drei Eingänge können verschiedene Sensoren angeschlossen werden. LEGO bietet derzeit folgende Sensoren an: Tastsensor, Temperatursensor, Lichtsensor und Rotationssensor. Hierbei differenziert man zwischen aktiven und passiven Sensoren. Temperatursensor und Tastsensor sind beispielsweise passive Sensoren und benötigen daher keine Stromversorgung. D.h. es liegt nur eine kleine Spannung an, mit der überprüft wird, wie groß der temperaturabhängige Widerstand ist oder ob der Tastsensor gedrückt wird.

Ausgänge

Die Ausgänge A, B und C laufen in drei Modi: an (on), aus (off) und gleitend (float). Ein Motor, der auf „float“ gesetzt wurde, befindet sich im Leerlauf. „Off“ hingegen bewirkt, dass beim Motor eine konstante Bremswirkung vorhanden ist. Für „on“ kann natürlich auch die Drehrichtung festgelegt werden. In welche Richtung sich der Motor letztendlich dreht, hängt davon ab, wie er angeschlossen ist. Die Drehrichtung kann auch dann gesetzt werden, wenn der Motor läuft. Die Änderung wird sofort wirksam, sodass er Motor vorher nicht extra ausgeschaltet werden muss.

1.3.3 Sensormodi

Der RCX holt die erwünschten Sensordaten mittels polling von den Peripheriegeräten. Mit dem NQC-Befehl `setSensorType()` wird die Sensorart für den jeweiligen Eingang festglegt. Da es mehrere Möglichkeiten gibt die Daten, die der Sensor liefert, zu messen, muss auch der Sensor-Mode mit dem Befehl `setSensorMode()` definiert werden.
Es stehen dafür beim RCX acht Modi zur Verfügung:

Raw Der Sensor detektiert mit einer Zahl zwischen 0 und 1023. Danach wird der Rohwert je nach SensorMode umgesetzt. Will man mit dem Roh-Wert weiterarbeiten, muss der SensorMode auf Raw gesetzt werden.

Boolean Der boolsche Wert ist entweder null oder eins und eignet sich besonders dazu den Tastensesor zu betreiben. Wenn man andere Sensoren verwendet, muss der Grenzwert angegeben werden. Liegt der gemessene Wert darunter, ist der Sensorwert null, über dem Grenzwert beträgt er eins.

Impulszähler Dieser Modus zählt, wie oft der logische Wert gesetzt wurde. Dies vereinfacht das Programmieren erheblich, da nur noch angegeben werden muss wie oft die Flanke gesetzt sein muss um einen Prozess auszuführen.

Flankenzählmodus Hierbei werden auch Impulse gezählt. Allerdings wird nicht unterschieden, ob der logische Wert gesetzt oder rückgesetzt wurde. D.h. beim Flankenzählmodus wird im Gegensatz zum Impulszähler jede Flankenänderung gezählt.

Prozent Der Lichtsensor wird normalerweise mit diesem Modus betrieben. Dabei wird der Raw-Wert in Zahlen zwischen null und hundert verwandelt, wobei null Prozent einem hohen Raw-Wert entspricht.

Rotationsmodus Dieser Modus funktioniert nur mit dem Rotationssensor. Je nach Drehung des Rotationssensors wird der Wert erhöht bzw. erniedrigt. Eine Umdrehung des Sensors wird in 16 Schritte aufgeteilt, ein Schritt entspricht somit 22.5 Grad.

Celsius und Fahrenheit Diese beiden Modi werden beim Temperatursensor verwendet. Der Raw-Wert wird in beiden Fällen zuerst in Celsius umgerechnet, beim Letzteren wird der Celsiuswert im Anschluss nach Fahrenheit konvertiert.

1.3.4 Motorensteuerung

Mit dem Befehl `setpower()` in NQC kann die Geschwindigkeit eingestellt werden, mit der sich der Motor drehen soll. Allerdings ist es nicht möglich eine Drehzahl einzugeben. Für die Geschwindigkeitssteuerung stehen lediglich acht Stufen (numerisch null bis sieben). Mit voller Leistung läuft der Motor bei sieben. Die Ansteuerung funktioniert mittels Pulse Width Modulation (PWM), einem Verfahren, bei dem die unterschiedlichen Leistungsstufen über die Anliegedauer der Spannung, hier fünf Volt, definiert werden.

1.3.5 Wie können zwei RCX miteinander kommunizieren?

Die Roboter tauschen über den Infrarot- Kanal Informationen aus. Damit können mehrere Roboter zusammen arbeiten (oder gegeneinander kämpfen). Außerdem besteht die Möglichkeit größere Roboter mit zwei RCX so aufbauen, sodass sechs Motoren und sechs Sensoren angeschlossen werden können. Die Kommunikation zwischen den Robotern funktioniert folgendermaßen: Mit dem Befehl `SendMessage()` kann der RCX einen Wert , der zwischen 0 und 255 liegt, über den Infrarot- Kanal senden. Alle weiteren Roboter empfangen diese Meldung und speichern sie. Das Programm in den Empfänger- Robotern kann den Wert der letzten Meldung, die mit `Message()` empfangen wurde, abfragen. Mit Hilfe des übertragenen Wertes kann das Programm den Roboter bestimmte Tätigkeiten durchführen lassen.

1.4 Programmiersprachen

Zur Programmierung des RCX können unterschiedliche Programmiersprachen verwendet werden. Neben dem LEGO-MindStorms Robotics Invention System (RIS), einer iconische Programmiersprache für Kinder bietet LEGO dacta RoboLab die weitaus größeren Möglichkeiten in der Programmierung des RCX.

Für Schulen und Hochschulen eignen sich iconische Programmiersprachen weniger, da der schöpferische Freiraum nicht genutzt werden kann. Für diese Anforderungen eignen sich Ada, C, C++, Java, Not Quite C (NQC), Scheme und Smalltalk. Je nach Lernziel kann auf die einzelnen Sprachen zurückgegriffen werden.
An der „U.S. Air Force Academy“ lernen angehende Offiziere mit Hilfe von Mindstorms die Grundlagen der Ada- Programmierung. Sie sollen ihnen im Rahmen der Bachelor of Science Ausbildung das nötige Wissen für den späateren Einsatz vermitteln.

1.5 Entwicklungsablauf

Folgende Schritte sind zum Bau eines RCX-Roboters auszuführen:
Als erstes wird die fimware heruntergeladen. Dies ist nur dann notwendig, wenn die Batterien ausgewechselt wurden oder der RCX die firmware „verloren“ hat. Anschließend ist für ein gutes Verständis des Problems, das gelöst werden soll, zu sorgen. Nun wird eine Lösung ausgearbeitet, die die Schritte beinhaltet, die der Roboter tun muss. Dabei ist es sinnvoll in der Gruppe andere Ideen zu diskutieren und eventuell eine neue optimierte Version zu generieren. Als Nächstes schreibt man den NQC code um die Ergebnisse zu implementieren. Danach wird der Code kompiliert und falls Fehler auftreten muss das Programm so lange nachgebessert werden bis es läuft.
Im Anschluss lädt man das Programm auf einen der fünf Programmspeicherplätzen des Roboters. Selbstverständlich muss der RCX beim Download eingeschalten sein. Bevor man das Programm mit dem Run-Knopf am RCX startet, muss es über mode ausgewählt werden. Sollte nun der Roboter nicht das tun was gedacht war, sollte als erstes die Korrektheit des Codes sichergestellt werden. Wenn dies der Fall ist, muss man das Design nochmals überarbeitet. Hilft dies ebenfalls nicht weiter ist es sinnvoll sich nochmals über das Problem Gedanken zu machen.

1.6 Programmbeispiel

Hier soll nun ein Programm für einen Roverbot dargestellt werden. Beim Roverbot handelt es sich um ein einfaches Gefährt, das nicht sensorgesteurt ist und lediglich einen Links- und Rechtsantrieb besitzt. Die Aufgabe besteht darin, den Roverbot drei Quadrate fahren zu lassen.

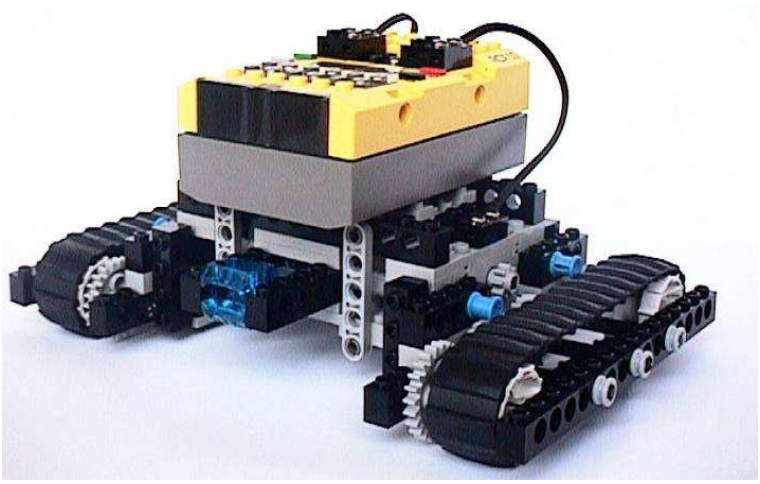


Abb.: Der Roverbot

```
#define FAHR_ZEIT 200
#define DREH_ZEIT 160

task main()
{
  SetPower(OUT_A+OUT_C,2);
  repeat(3)
  {
    repeat(4)
    {
      OnFwd(OUT_A+OUT_C);
      Wait(FAHR_ZEIT);
      OnRev(OUT_C);
      Wait(DREH_ZEIT);
    }
  }
  Off(OUT_A+OUT_C);
}
```

Die beötigten Variablen werden als erstes definiert. Bei diesem Programm sind dies die Fahrzeit und die Drehzeit, angegeben in hundertstel Sekunden. Die Fahzeit gibt an, wie lange der Roverbot gerade aus fährt. Die Drehzeit ist mit 1.6 sec so gewählt, dass der Roverbot eine Drehung von 90 Grad durchführt. Im Hauptprogramm – `task main()` – definiert man zunächst wie schnell die an Ausgang A und C angeschlossenen Motoren fahren sollen. In unserem Beispiel fahren die Motoren bei Aktivierung mit 37.5% der Maximalgeschwindigkeit, da Stufe 2 gewählt wurde.
In der inneren Schleife gibt man mit `OnFwd()` und `OnRev()` jeweils zwei Befehle, da beide sowohl Richtungsangabe als auch Aktivierung des Outputs enthalten. `OnFwd()` steht beispielsweise für die Befehle `SetDirektion(OUT_A+OUT_C, OUT_FWD)` und `SetOutput(OUT_A+OUT_C, OUT_ON)`. Die innere Schleife besagt, dass ein Quadrat gefahren wird. Da die Aufgabe allerdings drei Quadrate vorsieht, muss eine zweite Schleife programmiert werden. Nachdem der Roverbot die Aufgabe erfüllt hat, werden die Ausgänge wieder ausgestellt. Dafür steht der Befehl `Off(OUT_A+OUT_C)`.

1.6.1 Tasks

Ein NQC- Programm kann aus bis zu 10 Tasks bestehen. Das Hauptprogramm wird üblicherweise`main()` genannt und startet durch Betätigung der grünen Run- Taste des RCX. Die anderen Tasks werden nur ausgeführt, wenn eine laufende Task sie aufruft. Von diesem Moment an laufen die aktivierten Tasks gleichzeitig. Eine laufende Task kann eine andere laufende Task stoppen, indem sie den Befehl `stop taskname` verwendet. Bei Neustart beginnt sie allerdings wieder am Anfang und nicht an der Stelle, an der sie zuvor gestoppt wurde.
Der inzwischen mit einem Berührungssensor ausgestattete Roverbot soll nun die Verwen-

dung von Tasks an einem Beispiel verdeutlichen. Der Roverbot fährt nach wie vor seine Quadrate. Sollte er jedoch auf ein Hindernis treffen, startet ein Ausweichmanöver, um anschließend wieder Quadrate zu fahren.

```
task main()
{
 SetSensor(SENSOR_1,SENSOR_TOUCH);
 start check_sensor;
 start move_quadrat;
 }

task move_quadrat()
{
  while (true)
  {
    OnFwd(OUT_A+OUT_C); Wait(200);
    OnRev(OUT_C); Wait(160);
   }
  }

task check_sensor()
{
  while (true)
  {
    if (SENSOR_1 == 1)
    {
      stop move_quadrat;
      OnRev(OUT_A+OUT_C); Wait(50);
      OnFwd(OUT_A); Wait(80);
      start move_quadrat;
     }
   }
 }
```

Die task `main()` teilt dem RCX nur den Sensortyp mit und startet die beiden anderen Tasks. Danach wird die `main ()` beendet. Die task `move_quadrat ()` veranlasst den Roboter ständig im Quadrat zu fahren, `check_sensor` überprüft, ob der Berührungs- Sensor gedrückt ist. Ist dies der Fall, wird `move_quadrat` gestoppt. Dieses ist sehr wichtig, denn `check_sensor` übernimmt jetzt Steuerung über die Bewegungen des Roboters und fährt den Roboter zunächst ein Stück zurück. Im Anschluss fährt sie eine Drehung aus und startet `move_quadrat` wieder. Damit dem Motor keine inkonsistenten Befehle gegeben werden, muss `check_sensor` unbedingt `move_quadrat` anhalten.

1.7 Zusammenfassung

Wie man sieht ist die neue Generation der Spielsachen keineswegs nur für Kinder geeignet und lässt sich sehr gut in die Lehre integrieren. Es sei darauf hingewiesen, dass es neben LEGO auch andere Firmen gibt, die diese Art von Spielgeräten anbietet. Allen die wieder Lust aufs Spielen bekommen haben wünsche ich viel Spaß.

1.8 Quellenverzeichnis

- Klassner, F. LEGO Mindstorms: Not Just for K-12 Anymore; IEEE Robotics & Automation Magazine Juni 2003
- Overmars, M. Programmieren von LegoMindstorms-Robotern mit NQC; Oktober 1999
- Fröhlich, A.: LEGO RCX Hitachi H8/3292; März 2003
- http://www.vorlesungen.uni-osnabrueck.de/informatik/robot00/html/skript-1.html#88608
- http://mindrobots.lernnetz.de/hardware/mindstorms/mikrocomputer.shtml
- http://www.marioferrari.org/ lego_mindstorm.html
- http://www.brauer.in.tum.de/lehre/lego/prog_prakt_lego_links.shtml
- http://www.yoman.ch/mindstorms/LegoMazePathfinder.html
- http://insel.heim.at/mainau/330001/lego.htm
- http://www.vorlesungen.uni-osnabrueck.de/informatik/robot00/
- http://www.plazaearth.com/usr/gasperi/lego.htm
- http://www.cs.fh-aargau.ch/ robot/RobiHTML/node14.html
- http://www.informatikdidaktik.de/HyFISCH/WorkshopLehrerbildung2000/Papers/Hinz/sld007.htm
- http://mindrobots.lernnetz.de/hardware/mindstorms/mikrocomputer.shtml
- http://artdecom.mesh.de/projekte/werkzeuge/hardware/LEGO-hardware/rcx.html
- http://membres.lycos.fr/vcallede/lego/lienslego.htm
- http://moss.csc.ncsu.edu/ mueller/rt/

Kapitel 2

Zertifizierung in der Medizintechnik

Autor: Christian Schultz

Betreuer: Dipl.-Ing. Dipl.-Kfm. Johann Oswald

2.1 Einleitung

Die Medizintechnik stellt einen Bereich innerhalb der deutschen Industrie dar, der in den letzten Jahren zunehmend gewachsen ist und bei dem auch in Zukunft weiter ein Umsatzanstieg zu erwarten ist [15]. Für die Entwicklung werden Ingenieure benötigt, die dort interessante, anspruchsvolle Arbeitsplätze vorfinden, die ein hohes Maß an Sicherheitsbewusstsein erfordern.

Sinn und Ziel des Vortrages und dieser Ausarbeitung zum Hauptseminar ist es, einen kompakten Einblick in den Ablauf und die rechtliche Problematik der Zertifizierung von Medizinprodukten zu geben. Dabei wird besonders auf die Bereiche eingegangen, mit denen der Elektroingenieur vermutlich zu tun bekommt. Dem Aspekt Software als Medizinprodukt wurde dabei ein eigenes Kapitel gewidmet, da Software einen immer größeren Anteil des Entwicklungsaufwands darstellt.

Dabei können nur die Grundzusammenhänge vorgestellt werden und einige beispielhafte Ausnahmen kurz angesprochen werden. Auf umfassende Darstellung aller Ausnahmen wird bewusst verzichtet um den Überblick nicht zu verlieren.

2.2 Medizinprodukte

Zentrales Ziel des Gesetzgebers ist es, für Sicherheit im Umgang mit Medizinprodukten zu sorgen. Die Sicherheit des Patienten, des Anwenders und Dritter, sowie der Patientennutzen haben oberste Priorität MPG §1 [10].

2.2.1 Historisches

Schon früh wurde versucht durch Regelungen die Sicherheit von Medizingeräten zu garantieren. Die erste nationale Norm (VDE 0759) ist auf das Jahr 1928 datiert und beschäftigt sich mit der Regelung von HF-Heilgeräten [13].

Seit 1986 regelt die Verordnung über die Sicherheit medizinisch-technischer Geräte (Medizingeräteverordnung - MedGV) den Umgang mit Medizintechnik im Krankenhaus auf nationaler Ebene.

Die zunehmende europäische Harmonisierung führte dazu, dass am 2.8.1994 das Medizinproduktegesetz (MPG) in Kraft trat, das eine Umsetzung der Europäischen Direktive 93/42/EWG in deutsches Recht darstellt.

Aktuell gilt das Medizinproduktegesetz [10] in der 2. Änderungsfassung vom 7. August 2002, das zusätzlich zu den aktiven Implantaten nach 90/385/EWG und den Medizinprodukten auch die Regelungen zur In-vitro-Diagnostik nach 98/79/EWG enthält.

2.2.2 Anforderungen an Medizinprodukte

Definition Medizinprodukte

Nach Medizinproduktegesetz §3.1 [10] sind Medizinprodukte

- Instrumente, Apparate, Vorrichtungen
- Stoffe oder Gegenstände
- für das Medizinprodukt eingesetzte Software
- einschließlich Zubehör

zur Anwendung beim Menschen für

- Erkennung, Verhütung, Überwachung, Behandlung oder Linderung von Krankheiten
- Erkennung, Überwachung, Behandlung, Linderung oder Kompensation von Verletzungen oder Behinderungen
- Untersuchung, Ersatz oder Veränderung des anatomischen Aufbaus oder eines physiologischen Vorgangs
- Empfängnisverhütung

Das Medizinproduktegesetz beschäftigt sich ausschließlich mit Produkten für die Humanmedizin, eine Anwendung der Geräte und Vorschriften für die Tiermedizin ist möglich, aber durch das Gesetz nicht vorgesehen.

Grundlegende Anforderungen

In der Richtlinie 93/42/EWG Anhang I [9] werden die sog. Grundlegenden Anforderungen für Medizinprodukte definiert.

Produktsicherheit

Hier wird gefordert, dass besonderes Augenmerk auf die Sicherheit des Produktes gelegt werden soll. Dies muss schon bei der Auslegung und Konstruktion beginnen und dem allgemein anerkannten Stand der Technik entsprechen. Zudem sollte das Produkt eine ergonomische Bedienung ermöglichen, um eine sichere Anwendung zu gewährleisten.

Die Risikoreduzierung hat dabei immer zuerst bei der Beseitigung und Minimierung von Gefahren zu beginnen, erst wenn dieses Potential ausgeschöpft ist, sind für die restlichen Risiken Schutzmaßnahmen zu ergreifen. Die Risiken, die nicht durch Beseitigung und Schutzmassnahmen auszuschliessen sind, müssen unter Berücksichtigung der Leistungen vertretbar sein. Über diese noch verbleibenden Risiken muss der Benutzer umfassend aufgeklärt werden.

Leistungs- und Merkmalkonstanz

Die vom Hersteller versprochenen Leistungen müssen über den ganzen Produktlebenszyklus erhalten bleiben, dabei sind alle möglichen erlaubten Umgebungseinflüsse mit einzubeziehen. In erster Linie muss die Sicherheit in jedem nur denkbaren Fall immer gewährleistet werden.

Bei Produkten mit Messfunktionen muss eine für den Anwendungszweck ausreichende Konstanz und Genauigkeit der Messwerte sichergestellt sein.

Bereitstellung von Produktinformationen

Jedem Produkt sind Informationen beizugeben, die eine sichere Anwendung des Produkts ermöglichen. Meist erfolgt dies in Form von Gebrauchsanweisungen, die ab Klasse II b Pflicht sind. Zusätzlich sind Kennzeichnungen am Produkt vorzunehmen, sofern dadurch die Sicherheit erhöht wird. Produkte zur klinischen Prüfung bzw. Sonderanfertigung sind als solche zu kennzeichnen.

Von besonderer Bedeutung ist der vorgesehene Anwenderkreis des Produktes. Dies muss von diesen Personen, aufgrund ihres Ausbildungs- und Kenntnisstandes unter zu Hilfenahme der Produktinformationen, sicher bedient werden können. Informationen sind je nach Vorwissen entsprechend umfangreich darzustellen.

Außerdem muss die Produktinformation enthalten, sofern diese zutreffend sind:

- Anwendungszweck, Leistungsdaten und Funktionsweise des Produkts
- Kombinierbarkeit mit anderen Medizinprodukten
- Zulässige und unzulässige Umgebungseinflüsse
- Restrisiken und Störungen die bei diesem Produkt bei speziellen Untersuchungen auftreten
- Vorsichtsmaßnahmen bei Änderung der Produktleistung
- Genauigkeit bei Produkten mit Messfunktion

2.2.3 Klasseneinteilung

Entsprechend den Gefahren, die von einem Medizinprodukt ausgehen, wird es in eine von 4 Hauptklassen eingeteilt: je höher die Klasse umso größer das Gefahrenpotential. Entsprechend der Klasse sind unterschiedlich aufwändige Verfahren erforderlich um eine Übereinstimmung mit den gesetzlichen Anforderungen nachzuweisen.

Die Klassifizierung hat dabei streng nach den 18 Regeln der RL 93/42/EWG Anhang IX entsprechend, ihrer Zweckbestimmung zu erfolgen. Sind verschiedene Anwendungszwecke erlaubt, so ist die Klassifizierung getrennt für jede Anwendung vorzunehmen und das Produkt in die höchste sich ergebende Klasse einzustufen. Gleiches gilt, sofern mehrere der Regeln anwendbar sind.

Elektrische Medizinprodukte lassen sich meist anhand ihrer Eigenschaften in die Klassen einteilen, ohne dass dabei Spezialfälle berücksichtigt werden können.

In **Klasse I** sind alle Produkte enthalten, die nicht- oder vorübergehend invasiv sind, also in der Regel für weniger als 60 Minuten in den Köper eindringen.

Die Klasse I enthält zusätzlich zwei Unterklassen: in die **Klasse** I_S sind Produkte einzuordnen die steril sein müssen, sowie in der **Klasse** I_M Produkte, die über eine Messfunktion verfügen, ebenso ist eine Kombination beider Klassen möglich.

Alle Produkte, die mehr als eine Stunde aber weniger als 30 Tage invasiv sind, werden der **Klasse II a** zugeordnet. Ebenso alle Produkte, die auf eine Stromquelle angewiesen sind und für Diagnose oder Therapie verwendet werden, sofern sie keiner höheren Klasse zuzuordnen sind.

Produkte, die Energie oder Stoffe an den Patienten abgeben oder zur Überwachung der vitalen Körperfunktionen dienen, falls deren Störung eine potentielle Gefährdung des Patienten darstellt, müssen der **Klasse II b** zugeordnet werden. Dasselbe gilt für Produkte, die mehr als 30 Tage invasiv sind.

Die **Klasse III** umfasst alle Produkte, die direkten Kontakt zum Herz oder dem zentralen Nervensystem haben, oder arzneimittelwirksame Stoffe enthalten bzw. eine ähnliche Wirkungsweise zeigen.

Sehr wichtig für die richtige Zuordnung zu einer Klasse ist die Definition des Anwendungszwecks, was hier am Beispiel eines Blutdruckmessgerätes verdeutlicht werden soll:

Wenn der Hersteller ausschließt, dass die Daten dieses Gerätes für eine Diagnose oder Therapie verwendet werden [5], kann ein Blutdruckmessgerät in Klasse I_M eingestuft werden. Für die sinnvolle Anwendung in einer Arztpraxis muss ein Gerät jedoch mindestens die Klasse II a aufweisen. Angenommen das durch das Gerät die Blutdruckwerte eines Intensiv-Patienten überwacht werden, um Änderungen seiner Herzfunktion zu erfassen, so muss das prinzipiell gleiche Gerät in Klasse II b eingeteilt werden und allen für diese Klasse notwendigen Prüfungen unterworfen werden. Welche dies sind, wird in folgenden Abschnitt beschrieben.

2.2.4 Konformitätsbewertung

Alle Medizinprodukte, die in Europa neu in Verkehr gebracht werden sollen, müssen ein CE-Kennzeichen als äußeres Merkmal aufweisen. Nur Medizingeräte in der klinischen Erprobung und Sonderanfertigung benötigen kein CE-Zeichen.

Dieses Zeichen bescheinigt die Konformität mit den EG-Richtlinien. Nachdem die Konformität festgestellt wurde, darf das Produkt frei in ganz Europa vertrieben werden.

Der Katalog der anzuwendenden europäischen Normen [8] umfasst zahlreiche Einzelnormen, von denen je nach Produkt und Anwendung unterschiedlich viele relevant sind.

Für jedes Produkt muss eine technische Dokumentation erstellt werden, die es ermöglicht die Einhaltung der EU-Richtlinien nachzuweisen. Insbesondere muss enthalten sein

- Produktbeschreibung, inkl. aller geplanter Varianten
- Konstruktions- und Fertigungszeichnung in einem Umfang, der ein Verständnis des Produktes ermöglicht
- Risikoanalyse
- Liste der ganz oder teilweise angewandten Normen
- Prüfberichte, evtl. klinische Daten
- Kennzeichnung und Gebrauchsanweisung

Bei Produkten der Klasse I (ohne Zusatz) kann Konformität unter alleiniger Verantwortung des Herstellers festgestellt werden. Durch die **EG-Konformitätserklärung** erklärt der Hersteller in diesem Fall die Konformität mit den Grundlegenden Anforderungen [9]. In allen anderen Fällen erfolgt die Prüfung unter Beteiligung von sogenannten Benannten Stellen, als unparteiischen Dritten, die je nach Klasse das Produkt unterschiedlich umfangreich prüfen. Die vom Hersteller beauftragte Benannte Stelle zertifiziert die Konformität eines Produktes, d. h. sie prüft, ob ein angemessenes Vertrauen besteht, dass ein Produkt in Übereinstimmung mit einer bestimmten Norm ist [3].

Die **Benannten Stellen** („notified bodies“) müssen den staatlichen Stellen nachweisen, dass sie die erforderlichen Fähigkeiten besitzen um eine Prüfung vorzunehmen. Sofern die Anforderungen erfüllt sind, wird dies der EU-Kommission gemeldet und die Stelle bekommt zu ihrer Identifizierung eine europaweit eindeutige Kennnummer [11, 12], die auch neben dem CE-Zeichen des geprüften Produkts dargestellt werden muss, wie in Bild 2.1 zu sehen.

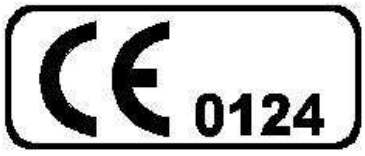


Abbildung 2.1: CE-Zeichen einer Benannten Stelle [6]

Es stehen je nach Klasse unterschiedliche Möglichkeiten der Konformitätsbewertung zur Verfügung, die vom Hersteller frei gewählt werden können. Der Hersteller hat einen der für sein Produkt zutreffenden Wege, in Bild 2.2, komplett zu durchlaufen.

Die in Bild 2.2 dargestellten Module haben im einzelnen folgende Bedeutung [3]:

Unter einem **vollständigen Qualitätssicherungssystem** (QSS) nach Anhang II wird dabei ein vom Hersteller unterhaltenes zugelassenes QSS für Auslegung, Fertigung und Endkontrolle verstanden. Die Benannte Stelle führt regelmäßige und unangemeldete Inspektionen des QSS durch und erstellt einen Bewertungsbericht darüber.

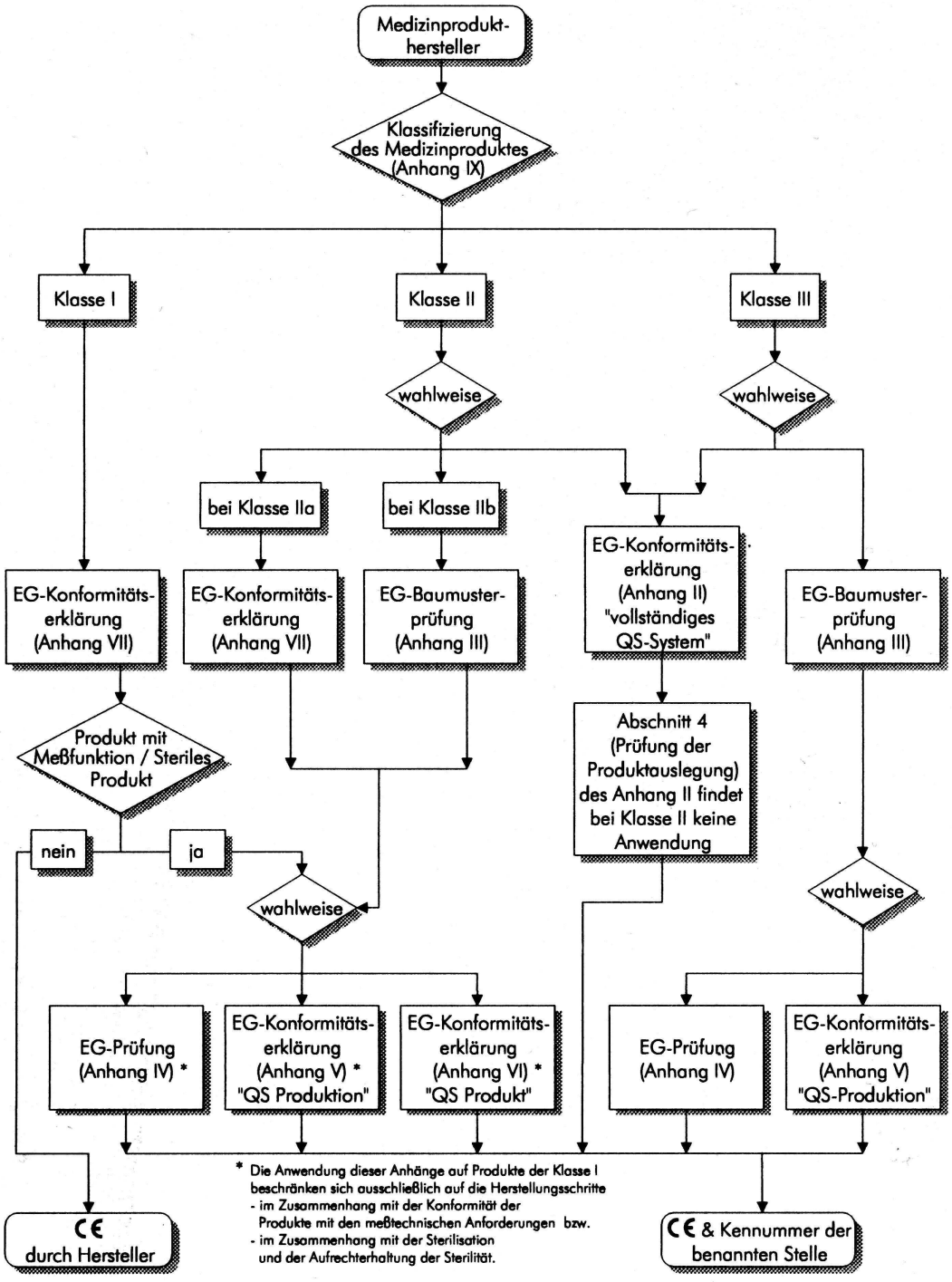


Abbildung 2.2: Möglichkeiten der Konformitätsbewertung [14]

Bei der **EG-Baumusterprüfung** stellt der Hersteller die technischen Unterlagen und ein repräsentatives Baumuster der Benannten Stelle zur Überprüfung zur Verfügung. Diese

prüft die Konformität mit den Grundlegenden Anforderungen und führt evtl. ergänzende Tests durch. Für die nach diesem geprüften Baumuster hergestellten Serienprodukte hat der Hersteller die Übereinstimmung durch eine Konformitätserklärung zu bescheinigen.

Um die Bedingungen der **EG-Prüfung** nach Anhang IV zu erfüllen, muss der Hersteller ein Überwachungssystem für den Vertrieb einführen, zusätzlich prüft die Benannte Stelle die Übereinstimmung des Produktes mit den Grundlegenden Anforderungen.

Bei der **QS Produktion** nach Anhang V muss der Hersteller ein zugelassenes QSS für die Produktion und Endkontrolle unterhalten, das bei der **QS Produkt** nach Anhang VII nur für die Überwachung der Endabnahme und Prüfung unterhalten werden muss. Die Benannte Stelle überwacht hier nur die QSS.

Falls der Hersteller nach der Auslieferung des Produkts Fehler- oder Schadensmeldungen erhält, muss er diese sorgfältig prüfen und, sofern diese neue bisher unbekannte Fehler oder Gefahren erkennen lassen, eine Sicherheitswarnung herausgeben. Diese muss die Geräteanwender über Art, Vermeidung und Verhalten beim Auftreten informieren. Dadurch soll größerer Schaden durch Störfälle, mittels einer möglichst frühzeitigen Entdeckung und Information der Benutzer über mögliche neue Gefahrenquellen, verhindert werden.

Sollte es trotz aller Überprüfungen zu einem Fehler kommen, der zu einem Schaden führt, muss der Hersteller anhand der Dokumentation beweisen, dass er den Richtlinien entsprechend vorgegangen ist und alle einschlägig bekannten Normen fachgerecht angewendet hat.

Laut EG-Richtlinie zur Produkthaftung haftet der Hersteller eines Produktes für den Schaden, der durch einen Fehler seines Produkts verursacht wurde. Seit 1989 ist diese Regelung durch das Produkthaftungsgesetz auch in Deutschland wirksam, auch für Medizinprodukte. Sollte es einem Hersteller nicht gelingen zu beweisen, dass er den Grundlegenden Anforderungen nachgekommen ist, so muss er für den Schaden haften. Angesichts der zu erwartenden Haftungssummen ist eine effektive Qualitätssicherung ein Muss um das Überleben der Firma zu sichern.

2.3 Zertifizierung von Software

Bei modernen Erzeugnissen übernimmt Software einen immer größeren Teil der Funktionalität. Medizinprodukte stellen hier keine Ausnahme dar, wenn auch der Einsatz von Software auf Grund der hohen Sicherheitsanforderung nicht unproblematisch ist.

Typische Einsatzzwecke [2] für Software in MP sind

- Software zur Steuerung, Regelung, Überwachung und Parameteranzeige eine aktiven Medizinprodukts
- Netzwerksoftware innerhalb eines abgeschlossenen Bereichs

- Software für Weiterleitung, Bearbeitung und Analyse von Patientendaten und medizinischem Bildmaterial

- Dosisberechnung von Medikamenten

2.3.1 Software als Medizinprodukt

Software wird nur an drei Stellen im MPG bzw. der RL 93/42/EWG explizit erwähnt [14]. So steht im MPG § 3.1 „*Medizinprodukte sind ... Instrumente, Apparate ... einschließlich der für ein einwandfreies Funktionieren des Medizinproduktes eingesetzten Software ...*“ und in § 3.9 „*Zubehör für Medizinprodukte sind Gegenstände, ... sowie Software, die selbst kein Medizinprodukt nach Nummer 1 sind, aber vom Hersteller dazu bestimmt sind, mit dem Medizinprodukt verwendet zu werden, damit dieses entsprechend der Zweckbestimmung des Medizinproduktes angewendet werden kann*“.

Weiter steht in der RL 93/42/EWG Anhang IX 2.3 „*Software, die ein Produkt steuert oder dessen Anwendung beeinflusst, wird automatisch derselben Klasse zugerechnet wie das Produkt*“.

Sofern Software einen Einfluss auf eine Therapie oder Diagnose haben könnte, ist davon auszugehen, dass sie wie ein Medizinprodukt behandelt werden muss, auch wenn sie nur Zubehör ist.

Software, die funktionaler Bestandteil eines Gerätes ist, wird bereits mit dem Medizinprodukt selbst klassifiziert und einer Konformitätsprüfung unterzogen. Die Klassifizierung von Software, die nicht an Hardware gebunden ist, stellt ein größeres Problem dar. Da Software für ihr Funktionieren immer auf Hardware angewiesen ist, die eine Stromquelle benötigt, muss diese als aktives Medizinprodukt angesehen werden und so klassifiziert und entsprechend geprüft werden.

Besonders kritisch ist das Verändern der ursprünglichen Software in einem bereits zertifizierten Gerät. Sofern durch das Update wesentliche Eigenschaften der Software verändert wurden, ist eine erneute Konformitätsbewertung von Nöten, was einen nicht unerheblichen Aufwand und entsprechende Kosten verursacht.

Wie Software zu prüfen und zu konzipieren ist, kann nicht aus den gesetzlichen Regelungen abgeleitet werden. Um sicherzustellen, dass die entwickelte Software den strengen Anforderungen des Medizinproduktegesetzes genügt, empfiehlt sich die Anwendung der harmonisierten Normen. Die Einhaltung dieser Normen ist zwar nicht vorgeschrieben, da aber die Normen den anerkannten Stand der Technik beschreiben, wird bei deren Gebrauch vom Gesetzgeber die Erfüllung der gesetzlichen Richtlinien vermutet [14].

2.3.2 Qualitätssicherung von Software

Die größte Hilfestellung für die Entwicklung von Software bieten die Normen zum Qualitätsmanagement von Software, wie die ISO 9000 Reihe und deren Ergänzung EN 46000 für die Bereiche Produktentwicklung und -fertigung.

Spezielle Ergänzungsnormen für Medizinprodukte wie die EN 46001, EN 46002 und die EN 46003 berücksichtigen die speziellen Anforderung dieses Bereiches. Besonders hilfreich ist die EN 60601-1-4, die auf programmierbare elektrische medizinische Systeme eingeht.

Da Software-Qualität nicht in den Normen definiert ist und auch sonst keine allgemeingültige Definition bekannt ist [16], kann die Qualität einer medizinischen Software nicht einfach geprüft werden.

In der Einleitung zur EN 60601-1-4 wird festgestellt, dass der Einsatz von Computertechnologie in medizinisch elektrischen Geräten einen Grad der Komplexität mit sich bringt, bei dem systematische Fehler bei einem in der Praxis akzeptablen Prüfaufwand eventuell nicht erkannt werden. Die Prüfung des fertigen Produkts alleine ist nicht geeignet, die Sicherheit von komplexen medizinischen elektrotechnischen Geräten sicherzustellen. Sie fordert einen Prozess und eine Prozessdokumentation zur Unterstützung der Sicherheit von medizinischen elektrischen Geräten mit programmierbaren elektrischen Subkomponenten.

Wichtigster Bestandteil des Prozesses ist das Risiko-Management, dessen Schritte in der Risiko-Management-Dokumentation protokolliert werden und dadurch nachvollziehbar sind.

Als erstes wird eine Gefährdungsanalyse erstellt, die alle relevanten Ursachen berücksichtigt, insbesondere Fehler, die durch bestimmungsgemäßen sowie unsachgemäßen Gebrauch durch den Anwender entstehen können. Diese werden in der Risiko-Abschätzung anhand ihrer Wahrscheinlichkeit und der sich ergebenden Gefährdung bewertet. Darauf aufbauend werden Verfahren zur Risiko-Beherrschung ermittelt, die durch Senkung der Auftrittswahrscheinlichkeit und des Schadensausmaßes, dieses so reduzieren, dass es unter dem maximal akzeptablen Risiko und so gering wie vernünftigerweise praktikabel ist.

Der gesamte Entwicklungszyklus wird in Phasen aufgeteilt, für die auch ein Verifizierungs- und Validierungs-Plan erstellt werden muss. Bei der Verifizierung wird geprüft, ob die am Anfang einer Entwicklungsphase gestellten Anforderungen erfüllt werden konnten. Durch die Validierung wird sichergestellt, ob das Produkt die Anforderungen für seine spätere Verwendung erfüllt. Die Leitung der Gruppe, die dies vornimmt, muss von der Entwicklergruppe unabhängig sein. Vor allem ist die Wirksamkeit und Zuverlässigkeit der Maßnahmen zur Risiko-Beherrschung in jeder Phase wieder kritisch zu hinterfragen.

Während des ganzen Entwicklungszyklus muss das Risiko-Management integraler Bestandteil sein. Die bei Entwicklungen immer auftretenden neuen Erkenntnisse sind mit in die Gefährdungsanalyse aufzunehmen und diese, sowie die resultierenden Änderungen, zu dokumentieren.

Dieses Verfahren ist auch für andere Entwicklungsprozesse in der Medizintechnik und auch außerhalb dieses Bereichs anwendbar und versucht dem Anforderungsprofil gerechte Produkte sicherzustellen. Der hohe Aufwand, den dieses Verfahren erfordert, wird bei komplexen Entwicklungen oft durch die Einsparung von Nachbesserungen aufgewogen.

2.4 Aufgetretene Fehler

Trotz des hohen Aufwandes, der bei der Entwicklung und Zertifizierung von Medizinprodukten betrieben wird, sind nie alle Risiken auszuschließen. Schäden, die trotz sorgfältiger Entwicklung entstehen, sind oft auf Bedienfehler zurückzuführen, wobei es hier besonders häufig zu schweren Unfällen kommt [4]. Darum ist es besonders wichtig, schon bei der Konstruktion auf einfache Bedienbarkeit zu achten, die selbst im oft hektischen medizinischen Alltag nicht falsch angewandt werden kann.

Wie „banal“ die Ursachen schwerwiegender Unfälle oft sind, sollen die nachfolgenden Beispiele verdeutlichen:

- Durch die Verwechslung einer sog. Kaltgeräteanschlussleitung mit dem Monitorkabel für die EKG-Überwachung wurden die nicht isolierten Elektroden-Anschlussstifte in die geräteseitige Anschlussbuchse gesteckt, so dass über den Thorax des angeschlossenen Patienten 220V Wechselspannung anlagen. [2]
- Der Anschluss des Luftschlauches eines nicht-invasiven Blutdruck-Monitors wurde mit einen intravenösen Zugang eines Patienten verwechselt. Wenn einem Patienten durch einen solchen Anschluss Luft zugeführt wird, kann dies eine ernsthafte Verletzung oder sogar den Tod zur Folge haben. Von vier bekannten derartigen Unfällen war einer tödlich. [7]

2.5 Zusammenfassung

Der gesetzliche Rahmen für Medizinprodukte in Deutschland ist das Medizinproduktegesetz. Medizingeräte benötigen ein CE-Zeichen um in Europa vertrieben zu werden. Die Geräte werden nach ihrem Risiko in Klassen eingeteilt, diese wiederum bestimmen welche Prüfungen durchgeführt werden müssen, wodurch schließlich Aufwand und Kosten einer Konformitätsprüfung bestimmt werden. Ein Produkt, das alle Prüfungen bestanden hat wird als konform mit den EU-Richtlinien angesehen. Außer bei Geräten mit geringem Risiko, erfolgt dies unter Beteiligung von sog. Benannten Stellen.

In den gesetzlichen Vorschriften finden sich keine genauen Angaben zu Software, aber Software, die wesentliche Eigenschaften eines Medizinprodukts hat oder beeinflusst, muss zertifiziert werden. Es wird empfohlen ein Risiko-Management für die Software-Entwicklungen zu etablieren. Sofern dieses den einschlägig bekannten Normen entspricht, wird davon ausgegangen, dass es auch den gesetzlichen Vorgaben entspricht.

Wenn ein Hersteller nicht beweisen kann, dass er den Grundlegenden Anforderungen nachgekommen ist, so muss er für den Schaden, den sein Produkt verursacht hat, haften.

Literaturverzeichnis

[1] Franz Josef Fergerl. *Seminar: Das Medizinproduktegesetz und die künftige Betreiberverordnung für Spitäler.* TÜV Östereich, Wien, 26.05.2003.

[2] Armin Gärtner. *Medizinproduktesicherheit: ein Leitfaden für den Betreiber.* TÜV-Verlag, Köln, 1. edition, 2001.

[3] Manfred; Menke M. Kindler. *Medizinproduktegesetz - MPG.* ecomed Verlag, Landsberg/Lech, 4. edition, 1998.

[4] W. Menke. Sichere anwendung von medizinprodukten. *Medizinmarkt fundamental, Anwender- und Betreiberpflichten für Medizinprodukte*, Sonderband 99-I:41–42, 1999.

[5] o.V. *Bedienungsanleitung: Digitales Blutdruckmessgerät zum Blutdruckmessen am Handgelenk M 251.* 4MBO International Elektronik AG, Polchingen, 04/2003.

[6] o.V. *AMD - Neues und Wichtiges.* AMG-GmbH, Berlin, 07.01.2004.

[7] o.V. *Vorfälle mit Blutdruck-Monitoren von Spacelabs.* TÜV Österreich, Wien, 12.11.2001.

[8] o.V. *Mitteilung der Kommission im Rahmen der Durchführung der Richtlinie des Rates 93/42/EWG.* Amtsblatt der Europäischen Union, 15.10.2003.

[9] o.V. *Richtlinie 93/42/EWG des Rates über Medizinprodukte zuletzt geändert durch 2000/70/EWG.* 2001.

[10] o.V. *Gesetz über Medizinprodukte (Medizinproduktegesetz - MPG). Neugefasst durch Bekantmachung vom 7. August 2002 (2. MPG-ÄndG) - Nichtoffizielle Lesefassung -.* Universität Magdeburg, Magdeburg, 2002.

[11] o.V. *LIST OF BODIES NOTIFIED UNDER DIRECTIVE 93/42/EEC Medical devices.* 2003.

[12] o.V. *Benannte Stellen in Deutschland.* DIMDI Deutsches Institut für Medizinische Dokumentation und Information, Köln, 28.08.2003.

[13] B. Schwarzzenberger. *Medizinische elektrische Geräte und Systeme.* Deutsche Elektrotechnische Kommission im DIN und VDE Fachbereich 8: Medizintechnik, Elektroakustik, Ultraschall, Laser, 1999.

[14] Gero Schütgen. *Qualitätsmanagement bei der Software-Entwicklung in Medizinischer Physik und Technik*. Berichte aus dem Institut für Medizinische Physik der Friederich-Alexander-Universität Erlangen-Nürnberg Band 3, Diss. Universität Erlangen-Nürnberg, 1. edition, 2000.

[15] Ilburga Tamer. *Medizintechnik-Industrie Kurzreport*. IG Metall, Frankfurt am Main, 04/2003.

[16] Ernest Wallmüller. *Software-Qualitätsmanagement in der Praxis*. Hanser Verlag, München, Wien, 2. edition, 2001.

[17] Erich Wintermantel. *Medizintechnik mit biokompatiblen Werkstoffen und Verfahren*. Springer Verlag, Berlin, 3. edition, 2002.

Kapitel 3

Realzeitanforderungen bei „Operationsrobotern“

Autor: Jing Yang

Betreuer: Dipl.-Ing. Robert Diemer

Über den Autor Jing Yang studiert zur Zeit an der TU München im 7. Semester Elekro- und Informationstechnik, Studienrichung Informations- und Komunikationstechnik.

3.1 Einleitung

In unseren täglichen Leben spielt Computer immer eine wichtige Rolle. Seit einigen Jahren hält er auch verstärkt Einzug in den klinischen Bereich. Das Einsatzgebiet reicht hierbei von der Datenbank zur Verwaltung der Patientendaten über die Computertomographie oder die Magnet-Resonanz-Tomographie bis hin zu Navigations- und chirurgischen Planungssystemen. Vereinzelt kommen auch schon Roboter bei chirurgischen Eingriffen zum Einsatz. Diese Systeme sind in der Lage dem Chirurgen im Verlauf einer Operation zu assistieren oder sogar selbständig einzelne Operationsschritte durchführen.
In diesem Artikel wird einen kurzen Überblick auf dem Operationsroboter (OP-Roboter) ausgezeigt.

Motivation Hier wird kurz vorstellt, warum ein OP-Roboter im Eingriff eingesetzt werden soll.

Typen der OP-Roboter Es gibt drei verschiedene Robotersysteme, nähmlich passive, semi-aktive und aktive Robotersysteme. Sie sind gemäß der Fähigkeit und funktionierender Weise im Eingriff unterteilt.

Technische Anforderungen Hier wird kurz Überblick auf technische Anforderungen des OP-Roboter-Systems, sowie: Multitasking-Unterstütung, Mensch-Maschine-Schnittstellen, Verbindungsstruktur und Fehlerbehandlungsfähigkeit, u.s.w., eingewurfen.

Weitere Entwicklungen Die Technik entwicklt immer, auch beim OP-Roboter.

3.2 Motivation

Jeder, der eine Untersuchung mit Gastroskope erlebt hat, kann sich gut erinnen, wie es ihm unangenehm ist. Der Speiseröhre leidet viel an der Bewegung und Umdrehung des Gastroskopes. Wenn es ein Mikroroboter wie „inchworm“ gäbe, das man als Tablett einnehmen kann, wäre es viel besser. Weil es automatisch und zärtlich im Magen bewegt ohne die Speiseröhre zu stören. Es ist immer der Wunsch der Chirugen, ihr operatives Ziel mit der geringsten körperlichen Belastung des Patienten zu erreichen. Beim Einsatz eines OP-Roboters wird die Belastung des Patienten deutlich reduziert.
Und eine andere Hoffnung von Mensch ist immer die Operationszeit zu verkürzen. Auch soll es erfolgreich erreicht wird wenn ein OP-Roboter richtig eingesetzt wird. Leider wegen vielen techischen Probleme ist es heute nur noch eine Hoffnung.
Aber deutlich verbessert ist die Genauigkeit des Eingriffs. Z. B.: mit der Hilfe des OP-Roboters kann man auf den Zentelmilimeter genau operieren. [3]
Auch andere Vorteile sind noch:

- Chirurg von köperlicher Arbeit befreien
- die Qualität und die Dokumentation medizinischer Diagnosen verbessern
- die Ergebnisse besser für die Ausbildung genutzt werden.

3.3 Typen der OP-Roboter

Nach dem Modus, darin ein OP-Roboter bei Operation das Task duchführen wird, kann man die OP-Roboter in folgenden Typen unterteilen: Passive Robotersysteme, semi-aktive und aktive Robotersysteme.

3.3.1 Passive Robotersysteme

Passive Robotersysteme können nur als passiver Arm zu Lokalisierungs und Positionierungszwecken verwendet werden. Der Roboter kann weder als „Führungshilfe“ sein, noch automatisch den Task durchführen. Was es machen kann ist nur durch mechanisches System die schweren Operationsgeräte (z.B.: Bestrahlungsgeräte) zu bewegen.

3.3.2 Semi-aktive Robotersysteme

Semi-aktive Robotersysteme ([1], Seite 15) basieren auf einer pysikalischen Beschränkung einer Aktion. Sie dienen gewissermaßen als „Führungshilfe“. Wegen der großen Anforderungen auf die Genauigkeit des Eingriffs (besonders in der Neurochirugie) kann man das chirugische Instrument nur bis einer bestimmten Begrentze führen. Ein typischer Beispiel davon ist das ACROBOT-System(siehe Bild. 3.1),der bei der Implantation künstlicher Kniegelenke eingesetzt wird.

Abbildung 3.1: ACROBOT mit passiver Positioner und Kontrollcomputer [5]

Es ist mit einem chirurgischen Planungssystem gekoppelt, mit dessen Hilfe, anhand eines aus CT-Daten erzeugten 3D-Modelles, die genaue Positierung der Implantate festgelegt wird. Das Modell wird anschließend in 2-dimensioalen Schichten zerlegt. In jeder dieser Schichten werden bei der Operation drei Sicherheitszonen difiniert(siehe Bild. 3.2), die das Verhalten des Roboters bestimmen.

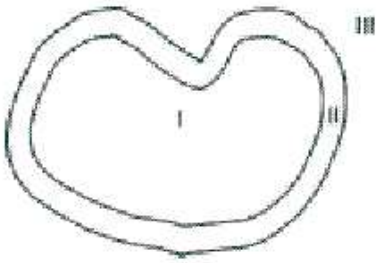


Abbildung 3.2: Die Sicherheitszonen bei ACROBOT ([1],Seite 18)

- Zone 1 ist sicherer Bereich, in dem das an dem Roboter montierte chirugische Instrument uneingeschränkt bewegt werden kann.
- Zone 3 ist verbotener Bereich. Eine Bewegung in diese Region ist verboten.
- Zone 2 Übergangszone. Eine Bewegung darin ist zwar erlaubt aber erschwert, damit der Operateur die falsche Richtung merken kann und eine intuitiven Richtungskorrektur machen kann.

Durch die Definition dieser Sicherheitszonen ist ACROBOT in der Lage, die Aktionen des Chirurgen, beim schichtweisen Sägen zur Anpassung des Implantats an

die Knochenoberfläche, zu begrenzen.

3.3.3 Aktive Robotersysteme

Aktive Robotersysteme ([1],Seite 13)können Teilaufgabe einer Operation (bzw. eine komplette Operation) autonom durchführen, allerdings unter voller Kontrolle des operierenden Arztes.
Z.B.: ROBODOC (siehe Bild. 3.3), bei dem wird mithilfe des zugehörigen Planungssystems ORTHODOC (siehe Bild. 3.3) präoperativ zunächst das geeignetste Implantat ausgewählt.
Intraoperativ wird dann von ROBODOC völlig autonom die Höhlung für das Implantat in den Oberschenkelknochen gefräst. Eine postoperative Betrachtung in der Unfallklinik Frankfurt zeigte, dass 2 Jahre nach dem Eingriff bei keinem der 580 mit diesem Verfahren operierten Patienten Komplikationen oder Verschiebungen des Implantats auftraten ([1], Seite 13).
Ein anderer Beispiel ist Radio Irradiation Robot (siehe Bild. 3.4).

Abbildung 3.3: ROBODOC und ORTHODOC [6]

Er wird in der Strahlenchirurgie verwendet, um schwere Bestrahlungsgeräte zu tragen

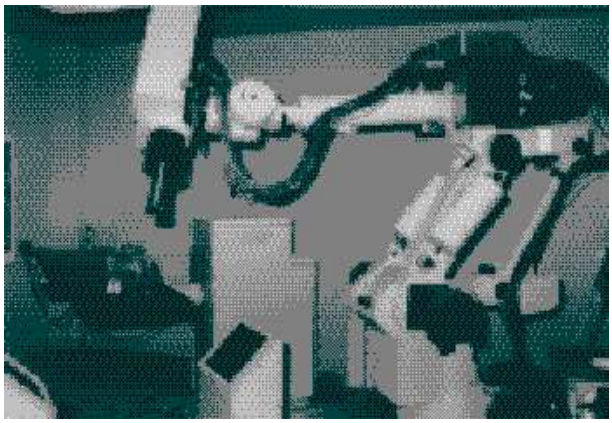


Abbildung 3.4: Radio Irradiation Robot ([1], Seite 14)

und zu führen. Außerdem kann er einen Tumor so zu bestrahlen, dass wenig umliegendes, gesundes Gewebe verletzt wird. Dies wird dadurch erreicht, dass in der Planungsphase der Operation eine optimale Folge von Punkten bestimmt wird, an die das Bestrahlungsgerät bewegt werden muss, um den Tumor am besten zu bestrahlen. Diese Punkte werden während der Operation durch den Roboter sukzessive angefahren([1], Seite 13).
Auch gehört voher genanntes Mikro-Roboter-System „inchworm“ zu diesem Typ. Er ist vergleichbar mit einer pneumatischen Raupe, die sich halbautonom forbewegen kann. Eine integrierte Kamera dient dem Arzt als Orientierungshilfe für die Steuerung. Falls Eingriffe vonnöten sein sollten, ist zusätzlich ein kleines chirurgisches Instrument vorhanden. Der inchworm wurde in die Klasse der aktiven Systeme eingeordnet, weil er von der Konzeption her keine pysikalischen Begrenzungen der Steuerung durch den Arzt aufweist ([1], Seite13).

3.4 Technische Anforderungen

Alle chirugische Roboter beim Eingriff, ob sie die Operation autonom durchführen können oder nicht, müssen unter voller Kontrolle des operierenden Arztes sein. Deswegen ist die Kommuniktion zwischen Mensch und Maschinen sogar zwischen Maschinen selbst ganz wichtig. Dabei gibt es viele technische Anforderungen. Z.B.:

Multitasking-Unterstützung

Multitasking ist im Prinzip mit jedem Rechner möglich, wobei man davon ausgeht, dass die Koordinierung der Tasks durch ein Realzeitbetriebssystem durchgeführt wird.
Bei der Operation müssen viele Aufgaben gleichzeitig durchgeführt werden. Z.B.:der Rechner des Roboters soll so snell wie möglich auf den Befehler des Arztes reagieren können während er die Daten der Sensoren bearbeitet.

3.4.1 Einsatz der Mensch-Maschine-Schnittstellen

Die Aufgabe einer Mensch-Maschine-Schinittstellen ist es, die Kommunikation des Benutzers mit dem System zu gewährleisten.

Ein-/Ausgabegeräte

Das typischste **Anzeigegeräte** ist der Bildschirm (Monitor). Da bei der Operation der Monitor meist einige Meter vom Chirurgen entfernt, empfehlt es sich, mindestens einen 21Bildschirm zu verwenden und die möglichen Optionen als große Buttons darzustellen ([1], Seite 24-25). Für wichtige Informationen, wie Systemstati oder die Position des chirurgischen Instrumentes, sollten getrennte Fenster, die nach dem Kachelprinzip angeordenet sind, vorgesehen werden. Ein zusätzliches Hilfefenster erleichtert die Behebung von Fehlern bei Problemfällen. Und die Verwendung eines **Sprachausgabesystems** erspart den ständigen Blick auf den Monitor. Das kann von Vorteil sein, wenn andere Aufgaben die volle Aufmerksamkeit des Benutzers erforden.

Einen Eingabevorgang über speziele **Eingabegeräte** mit Beteiligung des Benutzers bezeichnet man als Benutzereingabe. Typische Eingabegeräte sind Tastaturen und Maus. Neben dem Einsatz einer Maus gibt es zahlreiche andere Vorschläge und Realisierung für effective Eingabegeräte für den Einsatz in der Operation. Z.B.: ein Joystick, der ‚durch einen Einrastmechanismus, an beliebigen Stellen im OP-Feld, zum Beispiel an dem stereotaktischen Rahmen, befestigt werden kann. Er kann sowohl als Zeigegerät, wie auch zur Steuerung eines Roboters benutzt werden. Außerdem kann man auch Fußpedale verwenden. Der größte Vorteil davon ist dass der Arzt beide Hände frei haben kann. Aber auch ist die schwierige Koordination der Fußpedale der größte Nachteil.

Eine nicht benutzer gesteuerte Dateneingabe erfolgt neben dem Datenaustausch über Netzwerke oder dem Lesen von Disketten, Festplatten und CD-ROMs häufig durch das Auslesen von Sensorwerten. Sensoren dienen zur Ermittlung von Meßgrößen wie Temperatur oder Länge, zur Feststellung der Position von Objekten oder auch zur Bestimmung der Gelenkwinkel eines Roboters. Die ermittelten Daten können dann in System eingelesen und verarbeitet werden. Z.B.: Optische Sensoren werden meistens verwendet, um die Lage und Orientierung von Objekten zu bestimmen, Hindernisse zu erkennen oder Prozesse und Szenen zu überwachen.

Effectives Dialogmodell

Es ist natürlich dass im Eingriff viele Daten zwischen dem Benutzer und dem Rechener ausgetauscht werden müssen. Die Art und Weise und in welchen Schritten dieser Austausch stattfindet wird durch das Dialogmodell festgelegt. Es gibt drei Typen davon nähmlich: masken-, menu- und fensterbasierte Dialoge.

Maske ist wie ein Formular, durch Tastatureingaben werden die einzelnen Felder dieses Formulares gefüllt. Masken können Verzweigungen zu anderen Masken haben, zu denen mit einfachen Tastenkombinationen gewechselt werden kann.

Bei **Menus** kann man durch einfaches Mausklick die entsprechenden Befehler und Funktionen auswählen. Die Ablaufsteuerung geschieht in der Regel durch einen eindlichen Zustandsautomaten oder durch Dialogbäume

Im **fensterbasierten System** ist es möglich mehrere Anwendung(Tasks) gleichzeitig auf dem Bildschirm zu verwalten. Jede Anwendung läuft dabei in einem eigenen Fenster. Fenster lassen sich beliebig anordnen, beispilsweise nebeneinader (gekachelt) oder überlappend. Objekte werden in den Fenstern als Piktogramme, sogenannte Icons dargestellt, die durch Mausklick ausgewählt werden können. Gängige Fenstersysteme sind X-Windows, MS-Windows oder OS-2. Morderne Fenstersysteme sind auch eine große Unterstutzung des Teleroboticsurgerys. ([1], Seite 23)

Einfach und intuitiv handhabbare Schnittstelle zur Steuerung des Roboters

Bei der Gestaltung der Schnittstelle ist auch insbesondere darauf zu achten, dass sie einfach und intuitiv zu bedienen ist. Es soll ermöglicht werden, dass der Chirurg die an den Roboter montierte chirurgischen Instrumenten fast genauso leicht führen kann, wie er von Operationen ohen Roboterunterstützung gewohnt ist. Auch ist die Sychronisierung zwischen dem Gefühl des Arztes und der Bewegung des Roboters sehr wichtig, damit der Arzt richtig beurteilen kann.

3.4.2 Passende Verbindungsstruktur

Eine Bewegung der Patienten führt möglich zu einer ganzen Veränderung der geschpeicherten Daten. Der Rechner muss so schnell wie möglich daraufreagieren. Deswegen ist eine schnelle und erreichbare Verbindung zwischen Sensoren und Rechner sehr wichtig. Die Kommunikationsstrukturen zwischen den einzelnen Subsystemen wurden so gewählt, dass sich die größtmögliche Abtastrate erreichen läßt. Die Abtastrate gibt die Anzahl der Datenstrukturen an, die in einer Sekunde übertragen werden können. Sie sollte in echtzeitfähigen Systemen mindestens 500 Hertz betragen.
Aufgrund dieser Maßgabe und wegen der kurzen Distanz zwischen Sensor und Rechner ist der Einsatz des CAN-Buses sehr günstig. Der CAN-Bus wurde deshalb gewählt, weil, er nicht den Nachteil eines großen Protokolloverheads im Vergleich zu einer geringen Nutzdatenmenge hat, wie es bei anderen Feldbussystemen der Fall ist.
Nach CAN-Protokoll, erhählt die Nachrichten selbst einen sogenannten Identifier. Anhand dieses 11-Bits breiten Identifiers können alle Busteilnehmer auf einfache Weise überprüfen, welche der maximal 2048 möglichen Nachrichten für sie relevant ist. Zusätzlich ermöglicht diese Technik, dass eine Nachricht gleichzeitig von mehreren Teilnehmern übernommen werden kann. Außerdem ist CAN ein Multi-Master-System, das heißt jeder Busteilnehmer kann mit dem Senden einer Nachricht beginnen, ohne eine explizite Erlaubnis abzuwarten. Die Nutzdatenlänge einer einzelnen Nachricht beträgt 8 Byte. Mit diese Begrenzung können die meisten im Feldbereich anfallenden Daten mit hohem Datendurchsatz übertragen werden. Die maximal auftretende Latenzzeit in einem CAN-Netzwerk beträgt für die höchstpriore Nachricht 130 Bitzeiten. Ein weiterer Vorteil, der durch die Begrenzung der Datenlänge erreich wird, ist die Verringerung der Störanfälligkeit einer zu übertragenden Nachricht, die proportional zur Länge der Nachricht zu- oder abnimmt. Eine zusätzliche Sicherheit der Datenübertragung ist durch mehrere einander ergänzende Fehlererkennungsmechanismen (Bitmonitoring, Message Frame Check, Cyclic Redundancy Check) gewährleistet. Sie erkennen verfälschte Nachrichten mit einer sehr hohen Wahrscheinlichkeit, was zu einer automatischen Wiederholung der Transmission führt. ([1], Seite 43-45)

3.4.3 Fehlerbehandlungsfähigkeit

Die Sicherheitsanforderung ist immer ein wichtiger Punkt. Aufgrund der vielen komplexen Aufgaben, die ein Rechnersystem zu lösen hat besteht die Möglichkeit, dass es zu Fehlern in der Programm ausführung kommt. Auch fehlerhafte Eingaben oder Hardware-Defekte können Fehler auslösen, eventuell sogar das System zum Absturz bringen. Aus diesem Grunde sind umfassende Sicherheitsvorkehrungen einerseits für den stabilen Betrieb des Systems, andererseits für die körperliche Unversehrtheit des Bedienpersonals, beispilesweise beim Umgang mit Maschinen oder Robotern, zu treffen.

Um den korrekten Ablauf aller Funktionen zu sichern, muss man die Datenkosistenz ganrantieren und auch überflüssige Daten verrmeiden. Das heißt, man muss die **Daten-**

integritäts und **Datenrationalität** prüfen. Z.B.: die nomaler Temperatur des Körpers ist 37°C, wenn man jetzt ein Wert von 10° bekommen, bedeut es nicht dass man in einem Kühlschrank operieren, sondern ein Fehler.

Außerdem werden **Datentransfersicherungen** durch Paritätsbits und fehlererkennende Codes, beispielsweise cyclic redundancy checks (CRC) zur Sicherung der Kommunikation eingesetzt. ([1], Seite 23-24)
Ein potentielle Fehlerquelle sthellt in jedem Computersystem die Datenübertragung dar. Sowohl bei der inernen wie auch bei externen Kommuniktion müssen daher ausreichende Sicherheitsmaßnahmen ergriffen werden, um den Datentransfer zu sichern. Es gibt ja viele Fehlertypen, z.B.: Einzel-Bit-Fehler, Bündelfehler und Synchronisationsfehler.
Einzel-Bit-Fehler treten in der Regel als Folge von Rauschspitzen auf. Dabei überschreitet das Übertragungssignal kurzfristig die Bandbreite des Übertragungskanals, was zu einer fehlerhaften Detektion (eines einzelnen Bits) auf Empfängerseite führt.
Bündelfehler ist eine länger anhaltende Störung, zum Beispiel auf grund einer Überspannung hat diesen Fehlertyp zur Folge. Er manifestiert sich in mehreren Blöcken verfälschter Bits bei der Detektion.
Synchronisationsfehler: Sind Sender und Empfänger (auf Hardwareseite) nicht richtig synchronisiert, werden die Daten zwar korrekt Übertragen, aber nicht zu den korrekten Zeitpunkten empfangen. Dies führt dazu, dass alle Zeichen falsch erkannt werden. Fehlern diesen Typs kann nur durch einen Eingriff in die Hardware begenet werden (z.B.: durch stabilere Taktgeneratoren).
Fehlerhaft übertragen Daten können auf zweie Weise behandelt werden. Durch Verwendung von fehlerkorrigierenden Codes können verfälschte Daten „restauriert“ werden. Dies funktioniert allerdings nur bei Einzelbitfehlern. Wenn die Anzahl der veränderten Bits zu groß ist, versagen diese Codes. Aus diesem Grund verwendet man meistens fehlererkennende Codes. Sie sind in dr Lage zu entscheiden, ob die empfangenen Daten korrekt übertragen wurden. Falls dies nicht der Fall sein sollte, wird die Übertragung wiederholt. Ein häufig verwendete fehlererkennende Codes sind die Polynomcodes. Und einer der bekanntesten davon ist der CRC. Ein großer Vorteil des CRCs ist dass er in der Lage ist, alle Einzel- und Bündelfehler der Länge, die kleiner als oder gleich wie die Länge des Generatorpolynoms, zu erkennen. Die Länge der in der Praxis eingesetzten Generatorpolynome beträgt meistens 16 oder sogar 32 Bit. Damit ergibt sich eine Fehlerrate von $\leq 0.001\%$. Dies ist ein tolerierbares Restrisiko, das auch für medizinische Anwendungen ausreichende Sicherheit garantiert. ([1], Seite 93-96)

Zusätzlich zur Sicherung des Datentransfers, müssen auch zahlreiche Vorkehrungen getroffen werden, um das Risiko für die Personen, die sich im Arbeisraum des Roboters aufhalten, so gering wie möglich zu halten. Dazu kann man durch sogenannten **Watchdogs** den korrekten Betrieb der CPU's überwachen. Ein Ausfall einer CPU oder ein Fehler bei einem der laufenden Tasks führt zu einer sofortigen Beendigung aller Prozesse und zur Aktivierung der Bremsen des Roboters, so dass de momentan durchgeführte Bewegung beendet wird.

Die Kommunikation über das internen System, z.B.: VME-Bus, wird durch **Semaphore** gesichert. Auf dise Weise wird gewährleistet, dass zu einem Zeitpunkt immer nur ein Task Zugriff auf den Bus hat. Dadurch wird zum Beispiel vermienden, dass ein Prozess Daten liest, die gerade von einem anderen Prozess geschrieben oder geändert werden. Man spricht hier auch von einem gereglten Busbetrieb.

Die Transformationsmatrix, die die CT-Daten in Roboterkoordinaten umrechnet, wird durch eines Verwendung **Testmarkes** auf Korrektheit überprüft. Dies ist ein zusätzlicher Marker, der dem Patienten implantiert und bei der Referenzierung ebenfalls eingemessen wird. Ist die Differenz zwischen den eingemessen und den durch Transformation bestimmten Koordinaten zu groß, muss die Referenzierung widerholt werden. Die Sicherung der Korrektheit der transformierten Daten rechtfertigt hierbei die Mehrbelastung des Patienten.
Zur Vermeidung, dass sich der Roboter durch **Singularitäten** bewegt werden um diese Stellen Sicherheiszonen definiert, die nicht durchfahren werden dürfen.

Ein plöthlicher Konfigurationswechsel der Gelenkwinkel wird nicht erlaubt. Dadurch werden unkontrollierte Roboterbewegungen (die meist mit sehr hoher Geschwindigkeit ablaufen) vermieden.

Eingabefehler bei der Bedienung werden durch eine einfache Benutzeroberfläche vermindert. Außerdem kann das Risiko einer Fehlbedienung durch Schulung des OP-Personals vermindert werden. ([1], Seite 96-97)

3.5 Weitere Entwicklungen

Eine weitere wichtige Maßnahme zur Interaktionssicherheit ist die Verwendung eines **Totmannschalters**. Der Roboter kann nur dann bewegt werden, wenn dieser Schalter gedrückt ist. Ein Loslassen (wie es intuitiv in einer Notfallsituation geschieht) bewirkt ein sofortiges Beenden aller Roboteraktionen und eine Überführen desselben in einen sicheren Zustand. Aber leider im aktuellen Stadium des Systems ist der Totmannschlter allerdings noch nicht realisiert. ([1], Seite 97)
Des Weiteren muss mindestens ein Not-Aus-Schalter vorhanden sein, bei dessen Betätigung der Roboter sofort anhält und seine inernen Bremsen aktiviert.
Beim **Spracherkennungssystem** ist der große Vorteil, dass der Benutzer beide Hände frei hat. Allerdings sind die heutigen Systeme noch recht anfällig auf Hintergrundgeräusche, was einen stabilen Betrieb nicht garantieren kann. Desweiteren besteht das Problem der Mehrdeutigkeit - ein Spracherkennungssystem kann beispielsweise nicht zwischen dem Befehl „OKAY“ als Eingabe und der Antwort „OKAY“ eines Arztes auf die Frage eines Kollegen unterscheiden. Ein weiterer Nachteil besteht in dem Zeitaufwand und den Kosten für das Einlernen solcher Systeme. ([1], Seite 21)
Heutezutage wird immer mehr Bedeutung auf **Telerobotic Surgery** gelegen. Der Arzt

blickt gespannt auf den farbigen Bildschirm und kontrolliert den OP-Roboter durch einen Joystick, oder etwas ähnliches, ohne den Patienten im gleichen Zimmer zu sein zu brauchen. Man kann also ein Eingriff über eine Distantz mehr als 1000km durchführen.
Aber das größte Problem ist die Zeitverzögerung über lange Distanz. Nach Statistik darf eine Verzögerung zwischen der Eingabe eines Befehles des Arzts und dem Empfang der Quittung nicht mehr als 330ms sein, sonst kann der Arzt das Geführ der Sychronisierung zwischen seinen eingenen Bewegungen beim Joystick und den des Roboters verlieren, und nicht mehr richtig beurteilen [2]. Je kürzer diese Verzögerung dazwischen ist, desto besse ist es. Aber der aktuelle Datenübertragungsverzögerung im nomalen öffentlichen Netz (z.B.: Internet) zwischen großer Distanz ist meistens viel größer als 330ms, und verändert sich sehr stark wie ein Random. Ein spezielles Hochgeschwindigkeitsnetz für die Verkürzung der Verbindungszeit kostet einfach zu viel und sind auch nicht für die nomalen Menschen zu leisten. Wir hoffen durch die weitere Entwicklung der Technik, werden diese Probleme allmählich gelöst.

Fehler	Risiko	Behebung
CPU-Fehler	unkontrolliertes Systemverhalten	Watchdog
Resolver-Ausfall	Verlust der Gelenkwinkeldaten, fehlerhafte Positionsbestimmung	doppelte Resolver
Fehlerhafte Datenübertragung	falsche CT-Daten, falsche Sensordaten	CRC
Ungenauigkeit bei der Bestimmung der Referenzmarkerposition	zu großer numerischer Fehler	Wiederholung des Vorganges, Verwendung kegelförmiger Marker, automatische Segmentierung
Bewegung des Patienten	Verlust der Rferenzdaten	erneutes Referenzieren, Verwendung eines Navigationssystems
Marker löst sich	Verlust der Referenzierung, weitere Berechnungen fehlerhaft	Verwenden von 4 oder 5 Markern, anstatt 3
Singularitäten	unkontrollierte, sehr schnelle Bewegung des Roboters	Sicherheitskegel um Singularitätsstellen
Zu hohe Kraftausübung beim Führuen	zu große Geschwindigkeit	Überwachung der Meßdaten
Ungeübter Benutzer	fehlerhafte Bedienung	Schulung

Tabelle 3.1: Mögliche Risiken bei der Verwendung eines Robotersystems im klinischen Bereich ([1], Seite 92)

Literaturverzeichnis

[1] Keitel, Jochen: *Entwicklung einer intuitiven Mensch-Maschine-Schnittstelle für die Interaktion zwischen Chirurg und Roboter*;
Diplomarbeit bei der Institut für Prozeßrechnetechnik, Automation und Robotik, Universität Karlsruhe(TH), 09. 1998;
http://wwwipr.ira.uka.de/ keitel/mittle.html#download
Stand vom 19.07.1999.

[2] Sun, Lining;Xie, Xiaohui; Fu, Lixin; Du, Zhijiang: *Internet-based telerobotic surgery: problems and approaches*; Journal of Harbin Institute of Technology Vol35, No.2, Harbin Institute of Technology, Harbin, Feb. 2003.

[3] Fraunhofer Institut Biomedizinische Technik: *Roboter - der neue Assistent im Operationssaal*;
http://www.ibmt.fraunhofer.de/gruppe_c/papers/CS_romed99_de.pdf,
Stand vom 16.10.2002.

[4] Fraunhofer-Gesellschaft: *Roboter im Operationssaal der Zukunft*;
http://www.fraunhofer.de/german/publications/df/df1998/198-16.htm,
Stand vom 12.08.1999.

[5] *THE ACROBOT*;
http://www.me.ic.ac.uk/case/mim/projects/acrobot/index.html,
Stand vom 15.11.2002.

[6] Integrated Surgical Systems: *Integrated Surgical Systems provides computer-controlled, image-directed robotic products for surgical applications.*;
http://www.robodoc.com/eng/index.html,
Stand vom 08.06.2002.

Kapitel 4

Anwendung von RT-Linux in der biomedizinischen Forschung

Autor: Tobias Haberstroh

Betreuer: Dipl.-Ing. Alexander von Bülow

4.1 Einführung

In dieser Arbeit soll ein biomedizinisches Experiment zur Untersuchung von komplexen Herzschrittmacher-Algorithmen näher erläutert werden. Durchgeführt wurde es im Jahre 1998 am *Department of Medicine* der *Cornell University of New York*. Ziel war es, sog. *chaos-control-type pacing algorithms* auf ihre Eignung hin zur Unterdrückung von bestimmten Herzrhythmusstörungen zu untersuchen. Dazu war es erforderlich den Schrittmacheralgorithmus als Realzeitprozess in ein realzeitfähiges Computersystem zu implementieren. In der biomedizinischen Forschung sind Realzeitbedingungen allgegenwärtig. Eine vollständige Charakterisierung der wichtigsten biomedizinischen Signale ist mit einer Abtastrate von 10 kHz möglich. Bei dem hier beschriebenen Experiment ist eine Samplingrate von 2 kHz ausreichend [3], d.h. dass eine Periodendauer von 0,5 ms nicht überschritten werden darf. Somit stellt das Experiment *harte* Realzeitanforderungen an das Datenverarbeitungssystem. Im Gegensatz dazu stehen *weiche* Realzeitanforderungen, für die es ausreichend ist, wenn eine festgelegte statistische Mehrheit der Tasks die Deadline nicht überschreitet.

4.2 Betriebssysteme

4.2.1 Auswahl des Betriebssystems

Die Realzeitfähigkeit eines Systems ist abhängig von verschiedenen Faktoren (z.B. Leistungsfähigkeit der Hardware, Komplexität des Algorithmus, etc.) von ganz besonderer Bedeutung ist jedoch die Architektur des Betriebssystems. Man unterscheidet prinzipiell Allzweckbetriebssysteme sog. *general-purpose operating systems* (GPOS) von Echtzeitbetriebssystemen sog. *realtime operating systems* (RTOS). Das Ziel von GPOS ist die schnellst mögliche Abarbeitung möglichst vieler Tasks, es steht also der durchschnittliche Durchsatz im Vordergrund, wohingegen bei RTOS die Abarbeitung einzelner Echtzeitprozesse vor Erreichen ihrer Deadline die Hauptaufgabe darstellt. Es ist also einleuchtend das GPOS a priori nicht realzeitfähig sind. Durch Einsatz von Software-Interrupts ist es zwar möglich auch bei GPOS eine *weiche* Realzeitfähigkeit zu realisieren, allerdings nur für langsame Vorgänge, da die Interrupt-Auflösung relativ grob ist (z.B. 100 Hz für Intel-Prozessoren unter LINUX), vgl. [3].

Zur Visualisierung der gescannten Daten und für die Einstellung von Parametern über verschiedene Eingabegeräte (z.B Tastatur, Maus) wird ein sog. *graphical user interface* (GUI) verwendet. Ausserdem soll das System auch in der Lage sein über ein LAN mit anderen Rechnern zu kommunizieren. All dies sind Anforderungen, die eine GPOS-Funktionalität des Computersystems wünschenswert machen. Weitere Anforderungen sind die Adaptierbarkeit an zukünftige Experimente sowie eine möglichst kostengünstige Gesamtlösung. Anhand dieser Anforderungen erfolgte die Auswahl des Betriebssystems. Im folgenden möchte ich einige Vor- und Nachteile verschiedener Systeme auflisten.

DOS

Vorteile: - single-task Betriebssystem

Nachteile: - veraltetes System
- keine aktuelle Software erhältlich

Kommerzielle RTOS

Vorteile: - speziell für Realzeitapplikationen entwickelt

Nachteile: - hohe Anschaffungskosten
- urheberrechtlich geschützt
- keine GPOS-Funktionalität

Windows NT

Vorteile: - unterstützt einige wichtige RTOS-Eigenschaften

Nachteile: - zusätzliche kommerzielle Komponenten nötig
- unzureichende Testergebnisse
- Standardgerätetreiber nicht für RTOS-Erweiterung nutzbar

RT-Linux

Vorteile: - für mehrere Plattformen verfügbar
- tauglich für harte Realzeitanforderungen
- sehr geringer Kostenaufwand
- Linux Standard-Hardware-Treiber nutzbar
- GPOS Funktionalität

Nachteile: - keine bedeutenden Nachteile

4.2.2 RT-Linux

Aufgrund der Anforderungen des Experiments an das System und der Vorteile von RT-Linux haben sich die Wissenschaftler für dieses Betriebssystem entschieden. In diesem Abschnitt möchte ich näher auf die Funktionsweise und den Aufbau von RT-Linux eingehen. Bei RT-Linux handelt es sich nicht um ein eigenständiges Realzeitbetriebssystem,

sondern um eine Erweiterung von Linux, die jedoch grundlegende Realzeiteigenschaften implementiert. Die Hauptunterschiede zu Linux betreffen die Interruptverwaltung, die Steuerung der Prozesse durch den RT-Scheduler, und die Interprozeßkommunikation. Auf diese Aspekte wird weiter unten noch näher eingegangen.

Aufbau von RT-Linux

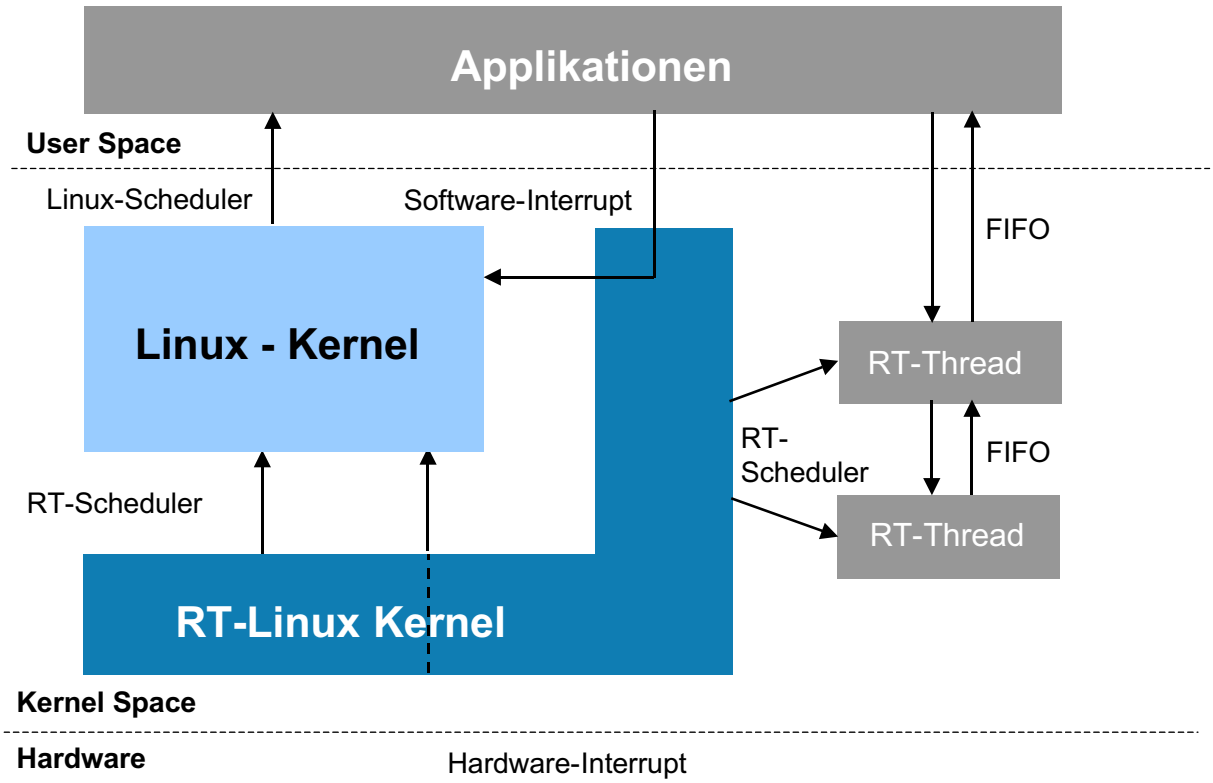


Abbildung 4.1: Schematischer Aufbau von RT-Linux aus [6]

Abbildung 4.1 stellt den Aufbau von RT-Linux in Form eines Blockdiagramms dar. Angedeutet ist auch die Interaktion zwischen den einzelnen Komponenten. Näheres darüber kann nachgelesen werden in [6].

Den Realzeitprozessen, auch als *Threads* bezeichnet (das sind leichtgewichtige Prozesse, die sich die gleichen Ressourcen teilen) wird vom RT-Linux Kernel Rechenzeit zugeteilt. Der Linux Kernel wird von RT-Linux ebenfalls als Thread angesehen, allerdings wird diesem die niedrigste Priorität zugewiesen.

Interruptverwaltung

Die Verwaltung der Hardware- als auch der Software-Interrupts wird fast vollständig von RT-Linux übernommen. Es wird in RT-Linux zwischen zwei Arten von Ereignissen un-

terschieden, die einen Interrupt auslösen. Zum einen sind das alle Ereignisse, die in fest vorgegebener Zeit behandelt werden müssen, dies sind Ereignisse die Realzeitprozessen zuzuordnen sind. Zum anderen gibt es Ereignisse, die zu einem späteren Zeitpunkt behandelt werden können, solche Ereignisse sind Prozessen zuzuordnen, die nicht in Echtzeit abgearbeitet werden müssen. RT-Linux sichert nun, dass die Ereignisse zuerst behandelt werden, die zum jeweils höchst priorem Realzeit Task gehören.

Scheduling

Man unterscheidet zwischen zwei Gruppen von Scheduling-Algorithmen: *dynamische* und *statische* Schedulingverfahren. Bei statischen Schedulingverfahren sind die Prioritäten der Threads von Anfang an festgelegt. Bei dynamischen Verfahren wird beim Aufruf des Schedulers anhand verschiedener Faktoren entschieden, welche Priorität dem Thread zugeordnet wird. In Realzeitsystemen muss gewährleistet werden, dass jeder Thread vor Erreichen der Deadline abgearbeitet ist. Bei periodischen Threads, wie im später beschriebenen Experiment, wird z.B. *Rate Monotonic Scheduling RMS* vgl. [5] eingesetzt.

RT-Linux ist modular aufgebaut. Module sind Dateien, die *Object-Code* enthalten (ausführbaren Binärcode) und zur Laufzeit geladen und wieder aus dem Speicher entfernt werden können, dazu stehen die Kommandos *insmod* und *rmmod* zur Verfügung. Aufgrund des modularen Aufbaus ist es möglich, je nach Erfordernissen verschiedene Arten von Schedulern zu implementieren (z.B. RMP, DMP)

Interprozeßkommunikation

Ein weiterer wichtiger Aspekt von RT-Linux ist die Interprozeßkommunikation. Normalerweise laufen Prozesse in strikt voneinander getrennten Bereichen ab. Daten eines Prozesses sind nicht zugänglich für andere Prozesse. Linux ermöglicht mehrere Arten der Interprozeßkommunikation, die jedoch RT-Linux nicht alle nutzen kann. In RT-Linux teilen sich die Threads selber den Speicher (*shared memory*) und können somit Daten sehr schnell austauschen. Ausserdem werden *FIFO*- Warteschalngen benutzt, um Daten zwischen Threads und Linux-Prozessen auszutauschen.

4.3 Das Herz

4.3.1 Anatomische Grundlagen

Das Herz hat im Vergleich zu anderen Organen des menschlichen Körpers einen relativ einfachen Aufbau, doch eine zuverlässige und korrekte Funktion des Herzens ist lebensnotwendig. Das Herz ist ein Hohlmuskel, der vierfach gekammert ist. Es besteht aus linkem und rechtem Vorhof, auch Atrium genannt, sowie der linken und rechten Herzkammer,

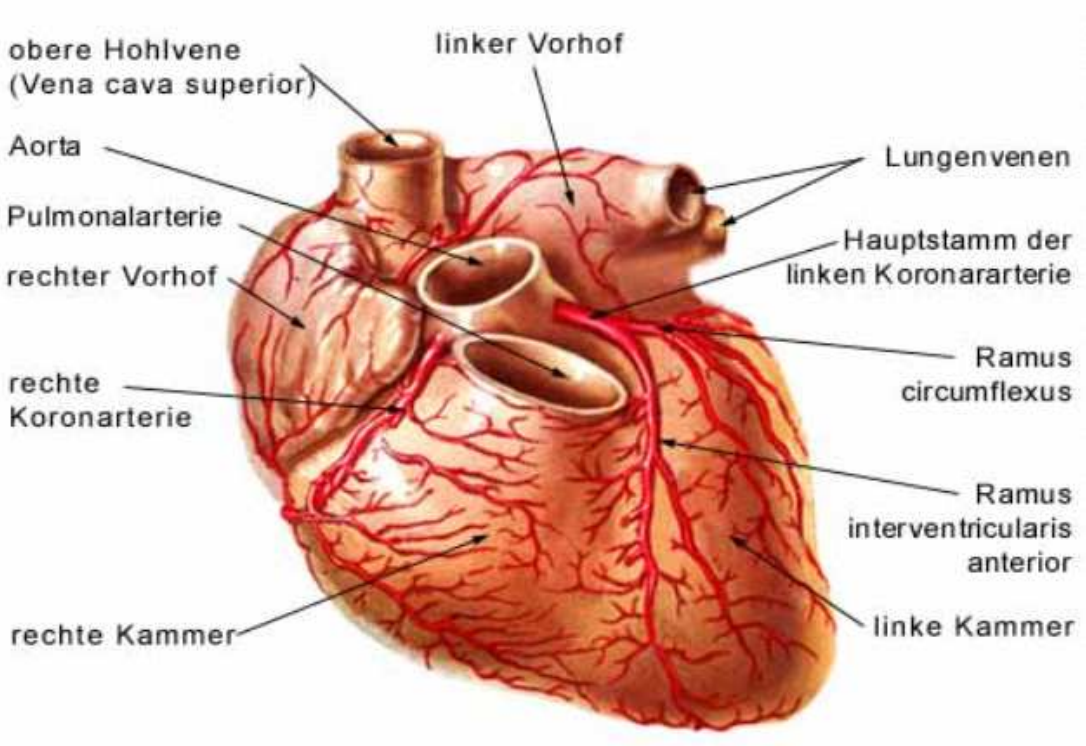


Abbildung 4.2: Das menschliche Herz

die Ventrikel. Mit jedem Schlag pumpt das Herz parallel Blut in zwei Kreisläufe. Man spricht vom Lungenkreislauf oder kleinen Kreislauf und dem Körperkreislauf, dem großen Kreislauf. Die linke und rechte Herzhälfte sind durch die Herzscheidewand getrennt. Die vier Herzklappen haben die Funktion von Ventilen und verhindern, dass Blut zurück in die Vorhöfe bzw. in die Kammern fliesst. Die *Mitralklappe* befindet sich zwischen linkem Atrium und linkem Ventrikel, die *Trikuspidalklappe* zwischen rechtem Atrium und rechtem Ventrikel. Die *Aortenklappe* ist das Auslassventil der linken Kammer hin zur Aorta und die *Pulmonalklappe* das Auslassventil der rechten Kammer zur Pulmonalaterie, vgl. [4].

Kurze Zeit vor der Kammerkontraktion kontrahieren die Vorhöfe, dadurch werden die Kammern prall mit Blut gefüllt, da das Kammergewebe elastisch ist spricht man auch von einer Vorspannung der Kammern. Erst jetzt tritt die Kontraktion der vorgespannten Kammern ein, mit deren Hilfe das sauerstoffreiche Blut in die Aorta hin zu den Blutgefäßen des Körpers und das sauerstoffarme Blut in die Pulmonalarterie hin zur Lunge gepumpt wird.

4.3.2 Elektrisches Reizleitungssystem

In dieser Arbeit ist das elektrische Reizleitungssystem des Herzen von besonderem Interesse. Die folgenden Ausführungen gelten für das Herz von Säugetieren und Menschen,

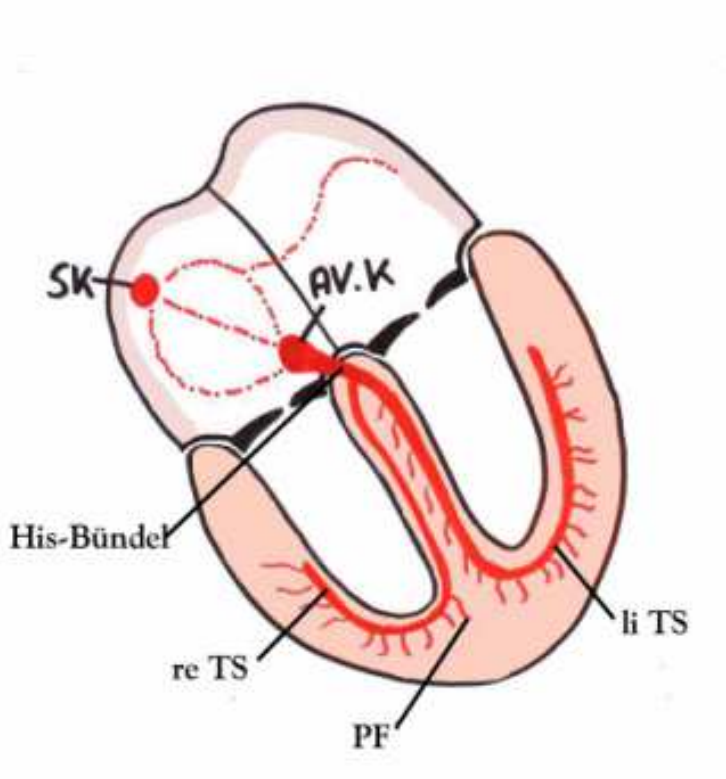


Abbildung 4.3: Elektrische Reizleitung im Herzen

aufgrund dieser Ähnlichkeit können Ergebnisse aus Tierversuchen auch auf den Menschen übertragen werden. Jede Art von Herzrhythmusstörung ist auf einen Defekt des elektrischen Reizleitungssystems zurückzuführen.

Die in Abbildung 4.3 gezeichneten Pfade veranschaulichen den Verlauf der elektrischen Signale im gesunden Herzen. Der Ursprung eines jeden natürlichen Schlagimpulses ist der *Sinusknoten*, dieser befindet sich im Bereich des rechten Atriums. Von dort breitet sich der Impuls über beide Atrien aus und gelangt zum *Atrioventrikulär-Knoten (AV-Knoten)*. Ausgehend vom AV-Knoten verläuft die Erregung über das *His-Bündel* in den linken und rechten Kammermuskel.

Die Kontraktion von Atrien und Ventrikeln wird direkt vom Verlauf der Erregung gesteuert. Um disen Verlauf grafisch darzustellen und evtl. vorhandene Defekte im Reizleistungssystem zu diagnostizieren wird ein EKG *(Elektro-Kardiogramm)* erstellt.

> ,,Das EKG ist die zeitliche Registrierung auf der Körperoberfläche des vom Herzen ausgehenden zeitlich veränderlichen elektrischen Vektors", [1].

Ein sogenanntes *Norm-EKG* wie es in Abbildung 4.4 zu sehen ist, beschreibt den Amplitudenverlauf der Erregung eines gesunden Herzens. Die verwendeten Bezeichnungen

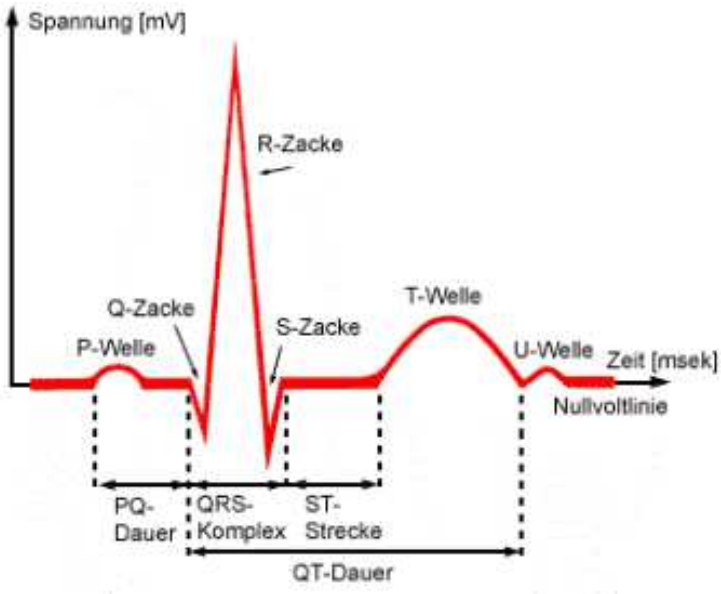


Abbildung 4.4: Norm-Elektro-Kardiogramm

haben dabei folgende Bedeutung, vgl. [1].

P-Welle: Erregung der Vorhöfe

Q-Zacke: Einleitung der Kammererregung

R-Zacke: Kammererregung

S-Zacke: Ausklang der Kammererregung

QRS-Komplex: Erregungsausbreitung (*Depolarisation*) der Kammern

T-Welle: Erregungsrückbildung (*Repolarisation*) der Kammern

U-Welle: kleine Nachschwankungen der Kammererregungsrückbildung, die nicht immer sichtbar sind

4.3.3 Kardiologische Arrhythmien

Wie schon weiter oben erwähnt, können Herzrhythmusstörungen mit Hilfe eines EKGs diagnostiziert werden. Als Referenz zum aufgezeichneten EKG dient das Norm-EKG. Abweichungen der Amplitudenwerte sowie Unterschiede im zeitlichen Verlauf der Erregung können auf eine bestimmte Arrhythmie zurückzuführen sein. Man unterscheidet Herzrhythmusstörungen nach dem *Ort ihrer Entstehung* sowie nach der *Art ihres Auftretens*. *Atriale Arrhytmien* entstehen im Vorhof und *ventrikuläre Arrhythmien* in der Herzkammer. Ist die Herzfrequenz stark erhöht, z.B. 300 Pulsschläge pro Min, so handelt es sich um eine *Tachykardie*, im Volksmund auch als ,,Flimmern" bezeichnet. Von einer *Bradykardie* spricht man bei einer sehr niedrigen Herzfrequnez. Bei einem Herzgesunden beträgt die

Herzfrequenz in Ruhe zwischen 60 und 80 Schläge pro Minute, Sportler haben teilweise einen sehr niederen Ruhepuls, z.B. 50 Schläge pro Min.

Von besonderer Bedeutung für nachfolgend beschriebenes Experiment ist die sog. *Reentry-Tachykardie*, die den AV-Knoten in sehr kurzen Zeitabständen durch retrograde Signale erregt. Dies führt wiederum zu einer zeitlichen Unregelmäßigkeit der Herzschlagimpulse, die durch den Atrioventrikulär-Knoten (AV-Knoten) geleitet werden. Man bezeichnet diese Erscheinung auch als *atrioventricular nodal conduction alternans (AVNCA)*, vgl. [2] und [3].

4.4 Das Experiment

In diesem Abschnitt soll das am *Department of Medicine* der *Cornell University of New York* im Jahre 1998 durchgeführte Experiment näher erläutert werden, vgl. [2]. Insbesondere sollen der Versuchsaufbau, die Motivation und die Implementierung mit RT-Linux beschrieben werden.

Motivation

In klinischen Laboratorien werden bei elektrophysiologischen Untersuchungen an Patienten elektrische Reize benutzt, um kardiologische Arrhytmien zu diagnostizieren. Dabei werden mittels eines Herzkatheters, der z.B. in die Vene des Oberschenkels eingeführt wird, Elektroden bis zum Herz transportiert. Diese Elektroden ermöglichen nun die direkte elektrische Stimulation des Herzens. Normalerweise wird diese Untersuchung von einer speziellen, urheberrechtlich geschützten Software überwacht und die elektrischen Impulse über einen relativ primitiven, manuellen Stimulator gesteuert. Diese Art der Steuerung der elektrischen Reize ist ausreichend für die Diagnose von typischen Herzrhythmusstörungen bei der keine komplexen Stimulationsmuster angewandt werden.

In dem beschriebenen Experiment werden jedoch komplexe sog. *chaos-control-type pacing algorithms* zur Unterdrückung von kardiologischen Arrhytmien untersucht. Diese speziellen Algorithmen müssen bestimmte Schrittmacher-Parameter in vorgegebener Zeit (beat-to-beat) modifizieren, d.h. sie stellen harte Realzeitanforderungen an das eingesetzte Computersystem. Weder die bei elektrophysiologischen Untersuchungen zum Einsatz kommende Standard-Software und Hardware sowie typische Steuerungssysteme wie LABVIEW, bieten die für diese Anwendung erforderliche Realzeitfähigkeit. Aus diesem Grund wurde von den Wissenschaftlern ein RT-Linux basiertes System entwickelt, daß in der Lage ist eine bestimmte Art von Arrhytmie zu steuern, die sog. *atrioventricular nodal conduction alternans (AVNCA)*, eine zeitliche Unregelmäßigkeit von Herzschlagimpulsen, die durch den Atrioventrikulär-Knoten (AV-Knoten) geleitet werden, vgl. [2] u. [3].

Tierversuche an Kaninchen haben ergeben, dass wenn die Zeit zwischen zwei aufeinanderfolgenden Erregungen des AV-Knotens sehr kurz ist, dies zu einer AVNCA führen kann.

Beim Menschen wurden AVNCAs beobachtet während sog. *Reentry-Tachykardien*, diese Art von Arrhythmie wird charakterisiert durch kreisende, retrograde Erregungen des Atriums, d.h. die Erregung wird über einen anormalen Pfad von den Kammern zurück in die Vorhöfe geleitet. Um AVNCA hervorzurufen, kann eine Reentry-Tachykardie induziert werden. Dies geschieht durch elektrische Erregung des Atriums innerhalb eines festgelegten Intervalls VA nach jeder Depolarisation der Ventrikel. Das VA-Intervall kennzeichnet den Zeitraum zwischen der Erregung der Ventrikel und der darauffolgenden Erregung der Atrien. Versuche an Herzen von Kaninchen haben ergeben, dass eine Variation des VA-Intervalls durch *chaos-control-type pacing algorithms* die Unterdrückung von AVNCA ermöglicht. Um zu Untersuchen, ob diese Technik zukünftig auch klinisch angewandt werden kann, wurde ein RT-Linux basiertes Realzeit-Rechnersystem entwickelt, das in der Lage ist das VA-Intervall in Echtzeit zu modifizieren.

4.4.1 Der Versuchsaufbau

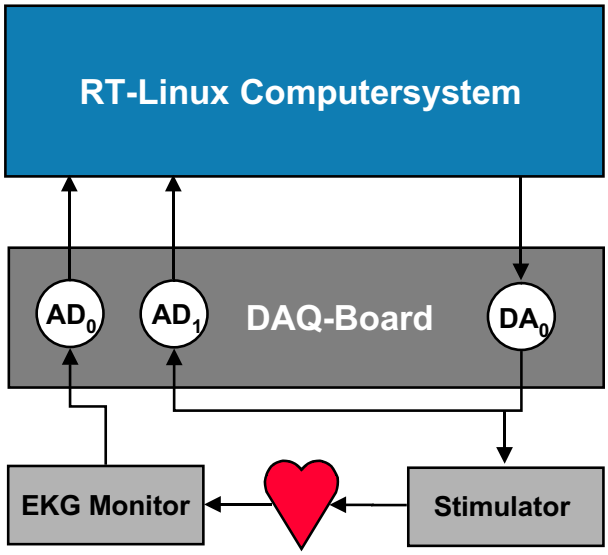


Abbildung 4.5: Versuchsaufbau

Abbildung 4.5 zeigt den prinzipiellen Versuchsaufbau, bestehend aus dem RT-Linux Computersystem, dem *Data Acquisition Board* DAQ Board sowie einer Stimulatorelektrode und dem EKG-Monitor.

Das RT-Linux Computersystem beinhaltet einen in C++ programmierten Realzeitprozess (**RT-Prozess**), der vom RT-Linux Kernel mit höchster Priorität betrieben wird und ein ebenfalls in C++ programmiertes *graphical user interface (GUI)* als Nicht-Realzeitprozess (**NRT-Prozess**) . Die Software, die frei erhältlich ist, wurde für einen Intel Pentium II Prozessor mit 266 MHz entwickelt. Das Rechnersystem hat 64 MB SDRAM und läuft unter Redhat Linux 5.0 und RT-Linux vers. 0.5a. Zur Datenerfassung wurde ein AT-MIO-16E-10 board der Firma National Instruments verwendet, im weiteren als *DAQ*

Board bezeichnet. Es werden drei Kanäle des DAQ Boards für das Experiment benötigt, zwei AD-Wandler und ein DA-Wandler.

4.4.2 Implementierung

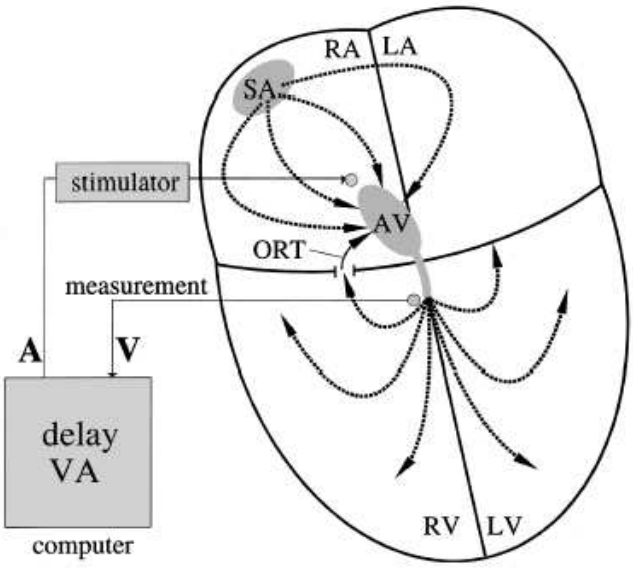


Abbildung 4.6: Schema

Abbildung 4.6 aus [3] zeigt die normalen Leitungspfade vom Sinusknoten über das rechte und linke Atrium zum AV-Knoten und zu den Ventrikeln. Ausserdem ist der anormale, retrograde Rückleitungspfad zu Erkennen (ORT) der eine Reentry-Tachykardie verursacht. Zu Experimentierzwecken kann diese Reentry-Tachykardie erzeugt werden, wenn das rechte Atrium zum Zeitpunkt A, nach einem festen Zeitintervall VA im Anschluss an die Ventrikelkontraktion zum Zeitpunkt V, mit einem elektrischen Impuls stimuliert wird. Funktioniert der pacing-Algorithmus korrekt, so wird das Intervall VA so berechnet, dass die AVNCA unterdrückt wird.

Zu diesem Zweck wird der pacing-Algorithmus in den Realzeitprozess, der mit einer Rate von 2 kHz aufgerufen wird, des RT-Linux Systems eingebettet. D.h. innerhalb von 0,5 ms müssen die Eingänge gescannt, Parameter übernommen und ggf. das VA-Intervall neu berechnet werden. Die grafische Benutzerschnittstelle (GUI) wird als herkömmlicher Linux Task implementiert. Dieser organisiert Zugriffe auf die Festplatte, die Kommunikation über das LAN, Tastatur- und Mauseingaben sowie die grafische Darstellung der EKG-Daten am Bildschirm. Die Kommunikation zwischen Realzeit- und Nicht-Realzeitprozess findet über zwei *FIFO* Warteschlangen und *shared memory* statt. Über die FIFOs gibt der RT-Prozess die gescannten Daten vom Datenerfassungsboard *data aquisition board (DAQ)* an den NRT-Prozess weiter. Globale Variablen die im shared memory abgelegt sind dienen einerseits zur Übergabe von im GUI eingestellten Parametern zum RT-Prozess

sowie zur Übergabe von Werten aus dem RT-Prozess zum NRT-Prozess. Abbildung 4.7 veranschaulicht die eben beschriebenen Zusammenhänge.

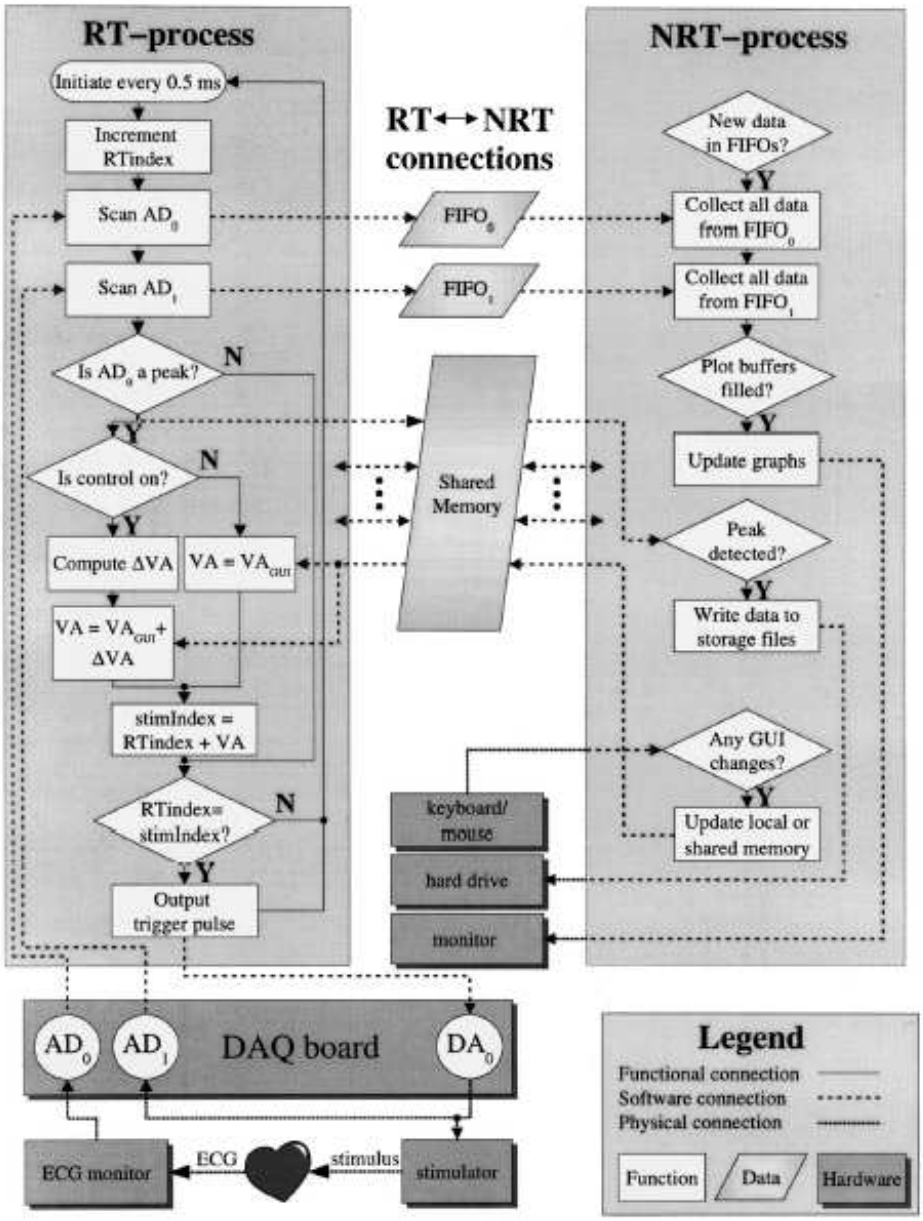


Abbildung 4.7: Ablaufdiagramm aus [3]

4.5 Schlussbemerkung

Die erfolgreiche Durchführung des Experiments zur Untersuchung von komplexen Schrittmacheralgorithmen beweist die Tauglichkeit des eingesetzten Systems in der biomedizinischen Forschung. Insbesondere die Entscheidung für Linux hat sich bewährt. Dies sollte auch Forscher auf anderen Gebieten motivieren die Verwendung von RT-Linux für ihre Forschungsprojekte in Erwägung zu ziehen. Da RT-Linux sehr effizient mit Rechenleistung umgeht, waren die Anforderungen an die CPU nicht sehr hoch, was weitere Kostenersparnisse gebracht hat.

Schließlich soll auch an dieser Stelle der Dank an die weltweite Gemeinde von Entwicklern und Nutzern von RT-Linux weitergegeben werden, die bei der Implementierung des Systems den Wissenschaftlern mit Rat und Tat zur Seite standen.

Literaturverzeichnis

[1] *Medicine World Wide*. Med-World AG, http://www.m-ww.de, 2003.

[2] J. Collins L. Glass D. Christini, M. Tremblay. *Dynamic control of cardiac alternans*. Biomedical Engineering Society, New York, erste edition, 1997.

[3] S. Markowitz B. Lerman D. Christini, K. Stein. *Practical Real-Time Computing System for Biomedical Experiment Interface*. Biomedical Engineering Society, New York, erste edition, 1998.

[4] Johannes Dr. med. Buschmann. *Physiologie und ihre Anwendung in Diagnostik und Therapie*. Lehrstuhl für Realzeitcomputersysteme, TU-München, München, erste edition, 2000.

[5] Georg Prof. Dr.-Ing. Färber. *Realzeitsysteme*. Lehrstuhl für Realzeit-Computersysteme, TU-München, München, erste edition, 2000.

[6] Alexander von Bülow. *Bestimmung der WCET auf AMD-Athlon Proz. unter Berücksicht. eines Realzeitbetriebssyst.* Lehrstuhl für Realzeit-Computersysteme, TU-München, München, erste edition, 2001.

Kapitel 5

Realzeitaspekte bei der Bergung der Kursk

Autor: Florian Rattei

Betreuer: Dipl.-Ing. Jürgen Stohr

Über den Autor

Florian Rattei studiert an der Technischen Universität München Elektro- und Informationstechnik mit dem Schwerpunkt Automatisierungstechnik.

Der vorliegende Artikel ist eine Zusammenfassung des Vortrags *Realzeitaspekte bei der Bergung der Kursk*, welcher im Rahmen eines Hauptseminars am Lehrstuhls für Realzeit-Computersysteme gehalten wurde.

5.1 Einleitung

Am 12. August 2000 sank das russische nukleargetriebene U-Boot Kursk [1] in der Barentssee nahe Murmansk. Die Unglücksursache war zu diesem Zeitpunkt noch unbekannt.

Nachdem alle Pläne verworfen worden waren, wie die 118 Mann Besatzung gegebenenfalls lebend gerettet werden könnten, beschlossen die verantwortlichen Stellen noch im selben Jahr, die Bergung des U-Boots voranzutreiben. Dazu wurde das Projekt international ausgeschrieben mit der Zielvorgabe, die Kursk möglichst als Ganzes spätestens bis Herbst 2001 zu bergen.

5.2 Konzeption

5.2.1 Projektierung

Die Koordination des Projekts oblag der Petersburger Firma Rubin, welche die Kursk auch konstruiert hatte. Im März 2001 erhielt das niederländische Konsortium Mammoet/Smit International den Zuschlag. Für die Automatisierung mittels Leit- und Prozessrechner wurde die ebenfalls niederländische Firma Raster Automation gewonnen. Die Simulationsmodelle für die Konzeption und später auch die Durchführung der Bergung entwickelte die hessische Ingenieurgemeinschaft IgH.

Sowohl die internationale Beteiligung als auch der enorme Zeitdruck bei dem Projekt hatten Einfluss auf die konkrete technische Realisierung:

- So wurde durch den Einsatz einer bereits aus Schwerlastprojekten an Land erprobten und vertrauten Hebetechnologie versucht, die Komplexität und damit auch die Entwicklungszeiten weitgehend in Grenzen zu halten. Die Folge waren sehr heterogene Strukturen mit eher einfachen Schnittstellen.
- Da für eine Entwicklung spezieller Produktionsanlagen keine Zeit blieb, musste man sich an bestehende Fertigungsmöglichkeiten halten.
- Teilweise verzichtete man auch ganz auf die Automatisierung technischer Prozesse und griff stattdessen auf Humanintelligenz zurück.

- Zudem schränkte der militärische Kontext der ganzen Unternehmung die Informationsfülle und Informationsflüsse ein und verstärkte dadurch den Zeitdruck.

5.2.2 Grundprinzip der Bergung

Obwohl es bis dato kein Bergeprojekt mit vergleichbaren Anforderungen gab, bestand weitgehend Einigkeit bei der Festlegung auf das Grundprinzip der Bergung. Die Basis bildete eine entsprechend dimensionierte schwimmende Bergeplattform, welche für den Einsatz bei diesem Projekt modifiziert werden sollte. Mammoet/Smit International hatte

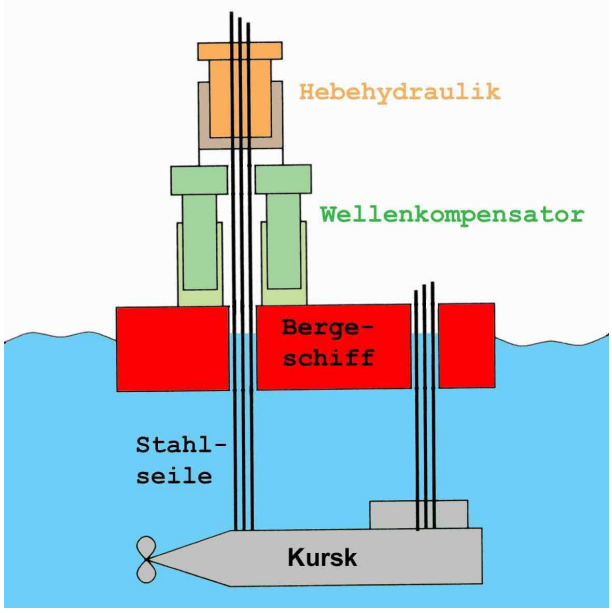


Abbildung 5.1: Grundprinzip der Bergung

sich bereits mit vielen Schwerstlasttransporten zu Land und Wasser einen Namen gemacht und bot mit der *Giant 4* eine geeignetes Ponton. Mittels dieser Plattform sollte die Kursk an ausreichend vielen Stahltrossen vertaut werden und anschließend bis unmittelbar unter das Bergeschiff gehoben und dort fixiert werden.
Da die Bergeplattform der Wellenbewegung der Barentssee ausgesetzt sein würde, wohingegen die Kursk – zumindest in der Anfangsphase der Bergung – starr am Meeresgrund liegt, bestand ein Schwerpunkt der Ingenieursarbeit in dem Entwurf und der Regelung der Wellenkompensation.
Diese wurde als ein an Deck örtlich verteiltes Array aus Stickstoff-Gasdruckfedern entworfen, welche die Wellenbewegung des Bergeschiffs relativ zur Lage der Kursk ausgleichen sollten. Auf den Kolben der Wellenkompensatoren waren hydraulische Hebevorrichtungen vorgesehen, über welche die Stahlseile und damit auch die Kursk verbunden wurden und letztlich gehoben werden sollten.

5.2.3 Problematik

Die physikalischen Randbedingungen des Projekts definierten die Anforderungen, welche bei Entwurf und Realisierung zu berücksichtigen waren:

- Problematisch bei der Umsetzung erwiesen sich hierbei die gewaltigen Dimensionen der Kursk. Je nach Restauftrieb ging man von einer aus 108 Meter Wassertiefe zu bergenden Masse von etwa 13000 Tonnen aus, welche sich auf 155 Meter Länge und 15 Meter Breite sowie 11 Meter Höhe inhomogen verteilten [4].
- Die Kursk wurde nicht hinsichtlich einer evtl. Bergung entworfen und gefertigt, weshalb die durch die Stahltrossen an die Außenhülle übertragenen Kräfte nur an definierten Stellen und genau dosiert appliziert werden durften. Konstruktionsbedingt erwiesen sich dabei 26 Punkte der Hülle als geeignet belastbar, woraus auch die entsprechende Anzahl an Wellenkompensatoren resultiert.
- Ebenfalls als problematisch erwies sich die Lage der Kursk im weichen, schlammigen Meeresgrund. Die Kräfte zum Anheben des U-Boots aus diesem Untergrund waren weitgehend unkalkulierbar, gleichsam erwies sich die Dynamik des Abhebevorgangs als schwer modellierbar. Geplant war, die Kursk erst mit dem Heck vom Bodenschlamm zu trennen und anschließend durch Kürzen der am Bug befestigten Trossen die waagerechte Lage wieder herzustellen. Je mehr sich die Kursk dann im weiteren Bergungsverlauf dem Bergeschiff nähert, desto ähnlicher verhalten sich beide.
- An Bord der Kursk befanden sich zwei atomare Reaktoren für die Energieversorgung und den Antrieb. Bei den Reaktoren dieser modernen Schiffsklasse ging man davon aus, dass sie selbst im Störfall noch definiert herunterfahren und passiv gekühlt werden können. Trotzdem galt es wegen der Empfindlichkeit des Innenlebens der Kursk, Neigungen größer als 5 Grad während des Hebevorgangs zu vermeiden.
- Das Wetter in Form von Wind, Seegang, Temperaturen etc. musste als externer, variabler aber nicht beeinflussbarer Parameter in die Simulationsmodelle mit einbezogen werden. Da der Bergevorgang einmal initiiert kaum wieder abgebrochen werden kann, mussten Wetteränderungen einkalkuliert und in der jeweiligen technischen Dimensionierung berücksichtigt werden. Das optimale Zeitfenster für die Bergung schloss sich mit dem Monat Oktober, da die Barentssee danach selbst kurzfristig nicht mehr verlässlich vorhersagbar ist.

5.2.4 Kritische Prozesse

Bei den technischen Prozessen an Bord des Bergeschiffs kann man Prozesse mit harten und weichen Zeitbedingungen differenzieren. Harte Zeitbedingungen sind unbedingt einzuhalten, ein Überschreiten führt unweigerlich zu massiven Schäden und/oder einer Gefahr für die Besatzung.

Das Nicht-Einhalten von weichen Zeitbedingungen hätte zwar zu suboptimalen Ergebnissen geführt, letztlich wären dadurch aber lediglich mehr Zeitbedarf oder Kosten entstanden.

Wellenkompensation

Als besonders zeitkritisch wurde in diesem Zusammenhang vor allem die Wellenkompensation identifiziert. Die Meereswellen laufen kontinuierlich - in Frequenz und Amplitude durchaus unterschiedlich - auf das Bergeschiff zu. Die Wellenbewegung der Wasseroberfläche lässt sich mathematisch mit dem Jonswap-Spektrum modellieren, wonach etwa jede tausendste Welle die doppelte signifikante (= normale) Wellenhöhe aufweist [3].
Sobald die Kursk mit dem Bergeschiff verbunden ist, mussten die Kompensatoren hochverfügbar die Wellenbewegung ausregeln, mit dem Ziel, die Kräfte an den einzelnen Trossen sowie die Kolbenwege beschränkt zu halten.

Die Schwachstelle lag dabei vor allem in den Befestigungen der Trossen an der Außenhülle der Kursk. Bei zu hohen Kräften drohten einige besonders kritische Befestigungspunkte den Belastungen nicht standhalten zu können. In diesem Fall wäre die Gegenkraft auf die entsprechenden Gasdruckfedern sprungartig weggefallen und die darin gespeicherte Energie frei geworden. Berechnungen zufolge hätte der maximale Arbeitsdruck einer einzelnen Gasfeder von 160 Bar ausgereicht, eine 50 Tonnen Masse 60 Meter hoch zu katapultieren [2].
Um die damit verbundene Gefahr für Schiff und Besatzung abzusichern, wurden die Kolbenschafte der Gasfedern konisch zulaufend gefertigt. Zwar wurde dadurch der Kolbenweg mechanisch nach oben begrenzt, dies allerdings zum Preis eines gegebenenfalls irreversibel festsitzenden Kolbens. In erster Konsequenz hätten dann die verbliebenen Kompensatoren die lokal gestiegenen Belastungen verteilen und ausregeln müssen. Ein Anschlagen des Kolbens am unteren Ende des Kolbenwegs hätte ebenfalls die Trossenkräfte sprungartig ansteigen lassen. Ziel war stets eine Lastverteilung, welche die größtmöglichen Sicherheitsreserven versprach. Daher wurden an die Verfügbarkeit und Realzeitfähigkeit der Wellenkompensatorregelungen hohe Anforderungen gestellt.

Resonanzvermeidung

In den Simulationsmodellen und später auch in der Realität erwies sich die Resonanzvermeidung als kritisch. Das komplette Bergesystem, im Wesentlichen bestehend aus den zwei großen Massen Kursk und Bergeschiff, war starr gekoppelt über die Stahltrossen und elastisch gekoppelt über die Gasdruckfedern der Wellenkompensation. Dieses System wurde von außen durch einlaufende Wellen periodisch angeregt und neigte dadurch zum Schwingen. Im Fall einer Resonanz hätten die auftretenden Seilkräfte bis weit über die systembedingt zulässigen Grenzen ansteigen können. Insbesondere die Kursk erwies sich in diesem Zusammenhang unter Wasser als nur schwach gedämpft, wohingegen das Bergeschiff an der Wasseroberfläche einen Teil der Schwingungsenergie in Form von Ge-

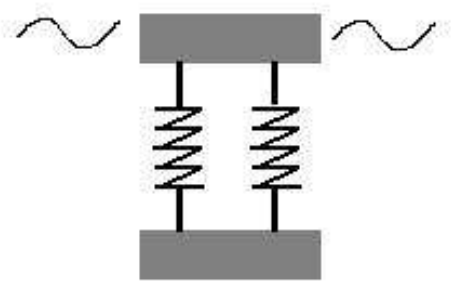


Abbildung 5.2: Feder - Masse - System

genwellen wieder abgab. Eine Zunahme der Schwingungsamplituden des Systems musste zuverlässig schnell erkannt und anschließend gegenphasig und synchronisiert ausgeregelt werden können, was besondere Anforderungen an die Realzeitfähigkeit der Kommunikationsprotokolle und den Übertragungsweg zwischen den einzelnen Wellenkompensatoren stellte.

Als weitgehend unkritisch erwiesen sich sämtliche technischen Prozesse im Zusammenhang mit der Hebehydraulik. So konnte der Bergevorgang beinahe jederzeit unterbrochen werden, um zum Beispiel alternative Parametersätze an den Simulationsrechnern des Leitstands zu testen. Ebenfalls unkritisch war die Kommunikation zwischen den einzelnen Rechnern des Leitstands sowie des Leitstands mit den einzelnen Prozessrechnern an Deck.

5.3 Technische Realisierung

5.3.1 Wellenkompensatoren

Jeder der 26 Wellenkompensatoren setzt sich aus 4 parallelen Gasdruckfedern zusammen. Auf den Kompensatoren waren hydraulische Hebevorrichtungen für Lasten bis zu je 900 Tonnen montiert, über welche das Wrack mittels verflochtener Stahlseile verbunden war. Während die Kompensatoren mit einer signifikanten Periode von 7 Sekunden eine Welle ausgleichen mussten, bewegten sich die dynamischen Anforderungen für den Hebemechanismus in Bereichen von etwa 10 Meter pro Stunde.
Bei bestimmter Druck- und Temperaturverteilung konnte über das Gasvolumen die Federhärte und damit letztlich die Dynamik jedes Kompensators geregelt werden. Der Arbeitsdruck in den Federn wurde durch Zu- und Abstrom des Gases über automatische Ventile gesteuert. Über zusätzliche manuelle Ventile konnten 32 Gasflaschen pro Kompensator zu Batterien konfiguriert werden, um letztlich damit die Elastizität der Kompensatoren einstellen zu können. Der Druck korrespondiert demzufolge mit der aufzunehmenden Ge-

wichtskraft, die Elastizität wirkt auf den Kolbenhub[3]. Sensorisch erfasst wurden unter

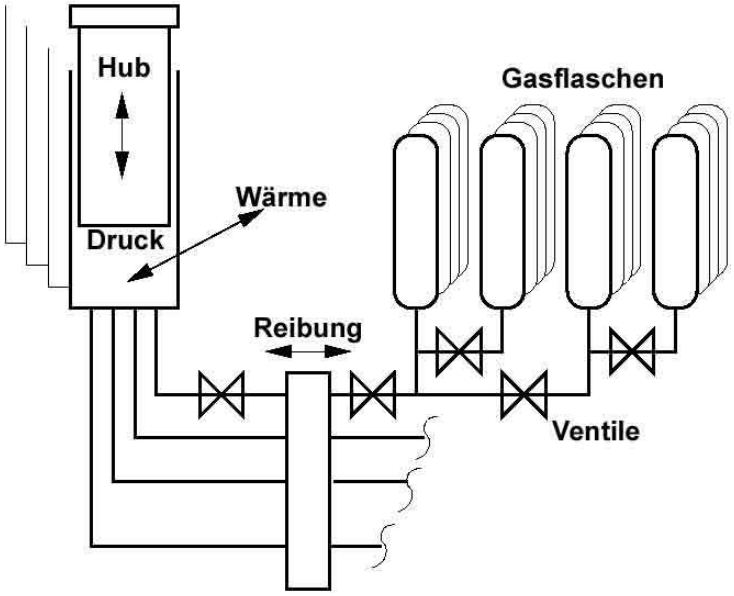


Abbildung 5.3: Wellenkompensatoren

anderem der Druck und die Temperaturen an verschieden Orten sowie indirekt das Gasvolumen über den jeweiligen Kolbenhub. Damit waren sämtliche Größen für die allgemeine Gasgleichung (Druck, Volumen und Temperatur) bekannt. Die Verhalten der realen Gasfedern war allerdings durch zahlreiche Nichtlinearitäten gekennzeichnet, was eine gute Modellbildung erschwerte. Sämtliche Sensoren und Aktoren wurden mindestens doppelt ausgelegt. Zusätzlich wurden voneinander abhängige physikalische Größen getrennt ermittelt, um die Plausibilität der Messwerte zusätzlich zu erhöhen. In Summe mussten etwa 700 Ventile betätigt und 424 Sensorsignale verarbeitet werden [2].

5.3.2 SPS als Prozessrechner

Bei einer derartigen hohen Anzahl an Sensor- und Aktorsignalen bietet sich eine dezentrale Lösung zur Verarbeitung an. Bei einer zentralen Lösung zur Regelung aller Kompensatoren wäre die Signaldichte bei einem einzelnen Prozessrechner sehr hoch, so dass Störungen und Übersprechen zusätzliche Probleme bereitet hätten. Zur Erhöhung der Verfügbarkeit hätten diese Rechner mit ihrer Vielzahl an Schnittstellen zudem mindestens doppelt ausgelegt werden müssen, wobei die Komplexität des Gesamtsystems weiter erhöht worden wäre.
Daher griff man auf eine dezentrale Lösung mittels 26 identischer speicherprogrammierbaren Steuerungen (SPSen) zurück. Auf jeder dieser Steuerungen wurden identische, verteilte Mehrgrößenregelungen implementiert. Die SPSen kommunizierten dabei über einen Feldbus, um die Daten benachbarter Wellenkompensatoren mit den eigenen Sensorwerten zu einem Satz Stelldaten verrechnet an die Prozessperipherie auszugeben. Veränderungen

z.B. bei der Parametrierung der Regelalgorithmen erfolgten zentral vom Leitstand aus. Alternativ konnte das Personal an Deck des Bergeschiffs direkt über die an den SPSen angeschlossenen Touchscreens erreicht werden. Die manuelle Ausführung einer Aktion wurde dem Leitstand auf entgegen gesetztem Weg quittiert.

Verfügbarkeit

Für eine hohe Verfügbarkeit des Gesamtsystems wurde bei der Auswahl der Einzelkomponenten auf eine hohe MTBF (Mean Time Between Failure) geachtet. Sollte dennoch eine einzelne SPS ausfallen, so war die Übernahme der Funktionalität durch die verbliebenen 25 Wellenkompensatoren vorgesehen.
Alle Sensoren waren mindestens doppelt ausgelegt, jeder Prozessrechner hatte zur Spannungsversorgung einen eigenen Dieselgenerator, der als Rückfalllösung noch mit einer Batterie gepuffert war. Die Kommunikation wurde durch einen redundanten Feldbus sichergestellt.

SPS in Realzeitsystemen

Grundsätzlich sind SPSen klassische Realzeitsysteme und als solche auch für harte Realzeitsysteme geeignet. Dies wird unter anderem durch feste Zykluszeiten gewährleistet. Programmiert wird meist mit inzwischen weitgehend standardisierten Programmen nach der DIN IEC 61131-3 Norm. Dabei handelt es sich um formale Vorgaben, auf denen die konkreten Entwicklungsoberflächen aufbauen. In diesem Fall beschränkt sich der Nachweis der Korrektheit der Software meist auf die Überprüfung der Parameter und Verbindungen. Alternativ können auch hohe Realzeitspezifikationssprachen zum Einsatz kommen, wobei die konkrete Umsetzung in Maschinencode semantikerhaltend über hardwarespezifische Tools erfolgt.

5.3.3 Leitstand

Der Leitstand auf der Brücke des Bergeschiffs bildeten 5 identische Industrie- PCs (Intel Pentium 3 Prozessoren - 1 GHz), welche untereinander über ein LAN verbunden waren. Zusätzlich hatte jeder Rechner noch eine Feldbuskarte installiert und konnte darüber mit den Prozessrechnern kommunizieren. Alle Rechner waren mit einer Dual- Boot Option ausgestattet und konnten je nach Bedarf unter Windows 2000 oder Suse Linux 6.4 betrieben werden.

Leitrechner

Unter Windows betrieben fungierten die Rechner als Leitrechner. Hauptaufgabe war in diesem Fall die Visualisierung der Prozessdaten von Deck des Bergeschiffs sowie die Pa-

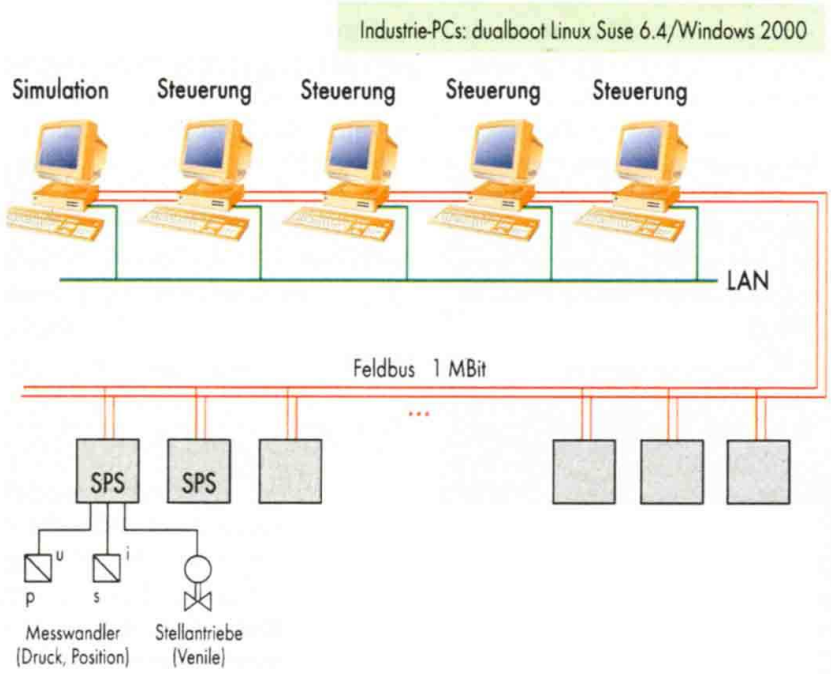


Abbildung 5.4: Gesamtmodell im Überblick

rametrierung der Prozessrechner über den Feldbus. Eine grafische Oberfläche reduzierte dabei die Datenfülle in eine für die Operatoren übersichtliche Form. Sollwertänderungen wurden softwareseitig erst einer Plausibilitätsprüfung unterzogen, um Fehleingaben durch den Bediener abzufangen.

Simulationsrechner

Unter Linux konnte eine Simulationsumgebung auf den Rechnern gestartet werden. Vor allem die Robustheit und herausragenden Netzwerkeigenschaften von Linux gaben für diese Systemwahl den Ausschlag. Darin wurde ein globales Modell von Bergeschiff und Kursk unter Berücksichtung aller relevanten externen Einflussfaktoren wie zum Beispiel einem kompletten Wettermodell abgebildet. Ebenfalls implementiert waren thermodynamische Modelle der Gasfedern bis hin zu Modellen für die SPSen und Prozessperipherie. Ein Abgleich der Simulationsdaten mit Versuchen an realen U-Boot-Modellen ergab eine sehr gute Übereinstimmung.
Die Realisierung erfolgte mit Matlab - Simulink. Um die Rechengeschwindigkeit der Modelle mit bis zu 300 Integratoren auf mehrfache Echtzeitgeschwindigkeit zu optimieren, wurde in den einzelnen Simulink Funktionsblöcken optimierter C-Code eingesetzt. Dadurch war es möglich, mehrere Vorgehensalternativen mit erträglichen Rechenzeiten zu simulieren. Dies war einerseits hilfreich während der Entwurfsphase, andererseits verkürzte es auch die Reaktionszeiten auf unvorhergesehene Situationen während der Bergung.
Über einen Softwareschalter konnten die Leitrechner wahlweise mit den realen Prozessrechnern oder mit den Simulationsrechnern verbunden werden. Als Startwerte für eine Simu-

Abbildung 5.5: Screenshot Leitstand (Ausschnitt)

lation konnten die von realen Prozessen an Deck erfassten Daten verwendet werden. Dies ermöglichte es, die zu erwartenden Auswirkungen einer veränderten Parametrierung der Prozessrechner bereits im Vorfeld zeitnah und trotzdem gefahrlos zu evaluieren.

5.3.4 Feldbus

Die Verbindung der einzelnen Prozessrechner untereinander und mit dem Leitstand erfolgt über einen redundanten Tokenbus. Physikalisch in Bus-Topologie ausgelegt bilden die angeschlossenen SPSen logisch aber eine Ringstruktur.
Dabei ist jede Station eindeutig über eine Identifikationsnummer (ID) ansprechbar. Sendeberechtigung hat jeweils diejenige Station, welche im Besitz des Tokens ist. Dabei handelt es sich um ein spezielles Datenpaket, welches von Station zu Station in der absteigenden Reihenfolge der IDs zyklisch weitergegeben wird. Die Station mit der niedrigsten ID schließt letztlich den logischen Ring durch Weitergabe das Tokens an die Station mit der höchsten ID.
Der Tokenbus wurde als deterministischer und somit für Realzeitsysteme geeigneter Bus ausgelegt. Im *non- exhaustive*[1] Modus betrieben darf dabei jede Station eines Tokenbuses nur eine genau definierte maximale Anzahl an Datenpaketen senden. Zudem wird ein Timer mit einer Token Holding Time initialisiert, nach dessen Ablauf die Sendeberechtigung spätestens weitergegeben werden muss [6].

[1]Den Gegensatz dazu stellt der *exhaustive* Modus dar, bei dem jede Station gemäß ihrem jeweiligen Datenaufkommen eine unbegrenzte Anzahl an Datenpaketen senden darf

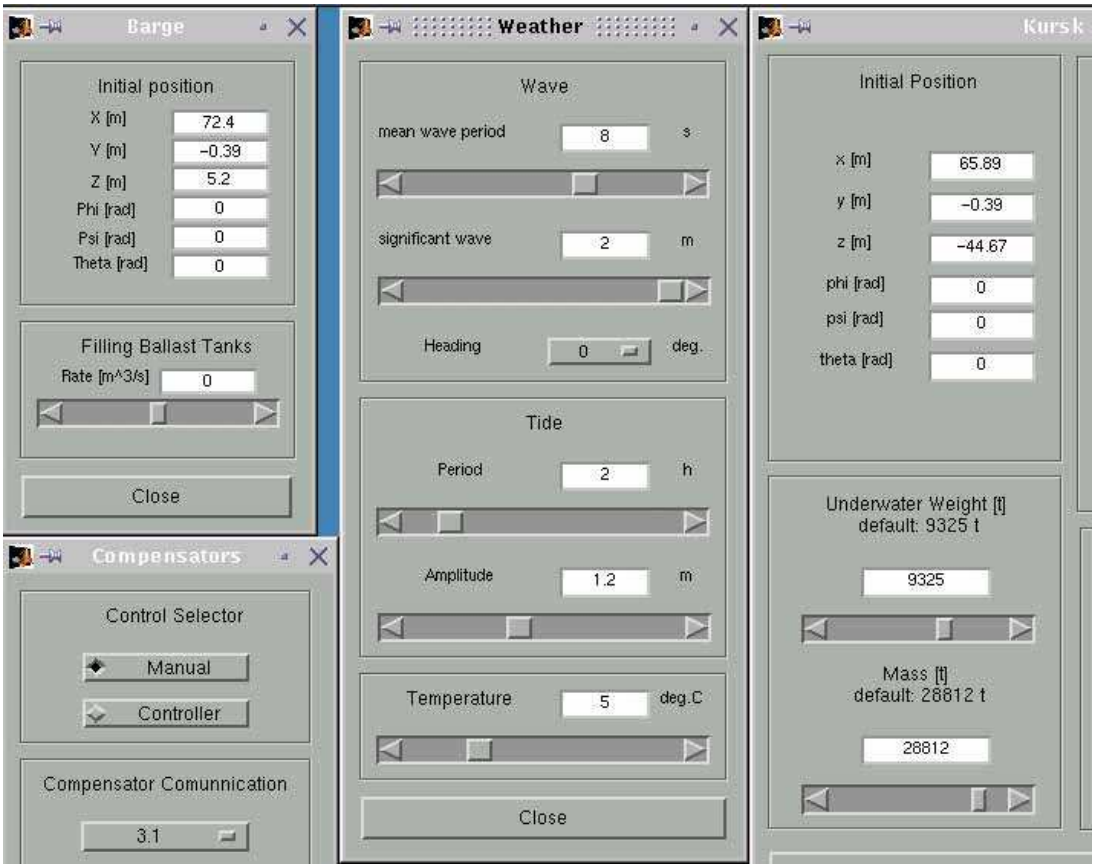


Abbildung 5.6: Screenshot Simulationsrechner (Ausschnitt)

Protokoll

Als Protokoll wurde *Open Modbus/TCP* verwendet, eine in TCP/IP eingebettete und gut dokumentierte Variante des Modbus- Protokolls der Firma Modicon. Somit konnten die Leitstände der Firma Raster bereits frühzeitig und ortsunabhängig mit der Simulationsumgebung des Ingenieurbüros IGH via Internet gekoppelt und erprobt werden.
Komplexität bewusst vermeidend wurden lediglich einfache Lese und Schreibefunktionen implementiert. Der Datenaustausch mit den einzelnen SPSen erfolgte mit Hilfe von Registern zu je 1024 mal 2 Byte. Die einzelnen Datenpakete enthielten neben einem Header und der jeweiligen Zieladresse noch einen Funktionscode für Lese- oder Schreibzugriffe, die eigentlichen Nutzdaten sowie zwei Bytes, welche die Länge des Pakets codieren [4].

Durch die bei Steuerungen üblicherweise relativ geringen Datenmengen erwies sich die für den Feldbus mögliche Datenrate von 1 Mbit pro Sekunde als ausreichend dimensioniert.

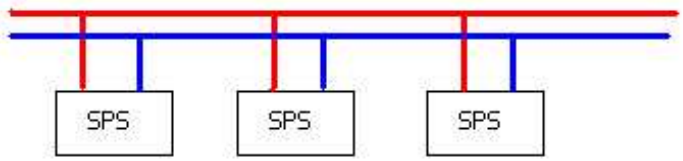


Abbildung 5.7: Topologie

5.4 Bergungsverlauf

Im Vorfeld der Bergung wurde die Bugsektion der Kursk vom Rest des Wracks mittels geeigneter Unterwassersägen abgetrennt. Dieser Abschnitt der Kursk, in dem Torpedos und Raketen lagerten, sollte in einer getrennten Aktion im Sommer des Folgejahres geborgen werden. Aus den Teilen der Bughülle und der Wasserstofftanks erhoffte man sich aufschlussreiche Informationen zur Unglücksursache.

Vom Absenken der Stahlseile bis zum Anheben des Wracks vergingen mehrere Tage. Aufgrund technischer Probleme bei der Befestigung der Haken für die Stahltrossen an der Kursk konnte erst Anfang Oktober 2001 und damit fast zum spätest möglichen Zeitpunkt mit der Bergung begonnen werden.
Bei optimalen Wetterbedingungen wurden die Trossen unter Spannung gesetzt und die Kursk angehoben. Dabei erwiesen sich die real auftretenden Kräfte als deutlich niedriger als befürchtet. Begünstigt wurde dies vor allem durch den geringen Wellengang. Nach 15 Stunden war der Bergevorgang ohne Komplikationen beendet und die Kursk unter dem Bergeschiff vertaut.
Während der gesamten Operation wurde die radioaktive Strahlenbelastung erfasst, die Messwerte bewegten sich allerdings immer im Bereich der natürlichen Radioaktivität. Zur Sicherheit hatte man sämtliche Begleitschiffe in einige Meilen Entfernung zum Bergeschiff gebracht. Eines dieser Schiffe stand über eine WLAN-Strecke ständig mit den Rechnern des Leitstands in Verbindung.

In einem Trockendock nähe Murmansk wurde die Kursk wieder abgesenkt und der russischen Marine übergeben.

5.5 Schlussbetrachtung

Das unter massiven Zeitdruck international entwickelte und realisierte Konzept hat die gestellte Aufgabe präzise erfüllt. Bemerkenswert ist dabei, dass trotz der Sicherheitsproblematik Hard- und Software „von der Stange“ zum Einsatz kam. Die Funktionalität des Gesamtsystems auch bei Grenzbelastungen hat die gewählte Realisierung während eines Sturms kurz vor dem eigentlichen Bergevorgang ebenfalls unter Beweis gestellt.

Abbildung 5.8: Kursk im Trockendock (weitgehend Unterwasser)

Die Gesamtkosten[2] für die russische Marine beliefen sich auf etwa 65 Mio. US-Dollar.

Als offizielle Unglücksursache teilte der russische Generalstaatsanwalt zum Abschluß der Ermittlungen mit, dass der Wasserstoffperoxid-Antrieb eines der Torpedos an Bord eine Explosion ausgelöst hatte, durch welche die Kursk Leck geschlagen und letztlich gesunken sei. Bei dem Unglück verloren alle 118 Mann der Besatzung ihr Leben.

Bildquellen

Die Abbildungen wurden aus den im Abschnitt *Literatur* angegebenen Quellen weitgehend unverändert übernommen.

[2]Die Summe variiert je nach verwendeter Berechnungsmethode und Quelle

Literaturverzeichnis

[1] Weiterführende Links zum Thema Kursk:
http://www.navy.ru
http://kursk.strana.ru/english/
http://www.kursksalvage.com

[2] Dr.-Ing. Torsten Finke, *Wie hebt man ein U-Boot - Computer helfen bei der Bergung der Kursk*, Zeitschrift „c't - magazin für computer und technik“, Heise Verlag, Heft 24/2001

[3] Dr. Wilhelm Hagemeister, Thomas Andraczek, *Hebung der Kursk mit MATLAB und Simulink*, Zeitschrift „MATLAB select“, The MATHWORKS GmbH, Heft 01/2002

[4] Dr.-Ing. Torsten Finke, *Die Bergung der Kursk*, Zeitschrift „Linux-Magazin“, Linux New Media AG, Heft 01/2002

[5] Dr.-Ing. Torsten Finke, *How Linux helped to Salvage the Kursk - Paper for the NLUUG - Autumn Conference 2002*, Ingenieurgemeinschaft IgH, Essen

[6] Prof. Jürgen Plate, Skriptum: *Grundlagen Computernetze*, Fachhochschule München

Kapitel 6

ACC - Adaptive Cruise Control

Autor: Christophe Achard

Betreuer: Dipl.-Ing. Matthias Goebl

Danksagung

Vielen Dank geht an meinen Betreuer Matthias Goebl, der mich sowohl bei meinen Recherchen, als auch bei der Vorbereitung und Probe des Hauptseminar-Vortrags tatkräftig unterstützt hat....

Zusammenfassung

Diese Hauptseminar-Arbeit befasst sich mit ACC - Adaptive Cruise Control, einem „intelligenten“ Abstandsregeltempomat im Automobil.
Diese Arbeit befasst sich näher mit dem allgemeinen Funktionsprinzip und dessen technische Implementierung.
Dabei wird ein Überblick über die vernetzten Bestandteile des ACC gegeben, und Beispielhaft einige physikalische Effekte näher erläutert, deren Eigenschaften sich das ACC zu nutzen macht.
Zum Schluß sind noch einige Gedanken zur Realzeitfähigkeit der eingesetzten Systeme aufgeführt.

6.1 Einleitung

In der Automobilindustrie gewinnt zunehmend der Anteil der elektronischen Komponenten an Wichtigkeit. Er dient nicht nur dazu, z.B. den Motor besser zu regeln, Komfortfunktionen zu ermöglichen (wie z.B. den Tempomaten), für die Auslösung der Air-Bags zu sorgen, sondern ist inzwischen auch symbolträchtig für die Innovationskraft des Automobilherstellers. Weitere Fahrerassistenzsysteme, wie z.B. ABS, ESP sollen hier stellvertretend genannt werden. In diesem Zusammehang soll ein kleiner Überlick über das ACC - Adaptive Cruise Control gegeben werden...

6.1.1 Was ist ACC?

ACC - Adaptive Cruise Control ist ein Fahrerassistenzsystem, das, zumindest in diesen Jahren, verstärkt in Autos der Oberklasse zu finden ist. Es ist eine Art intelligenter „Abstandsregeltempomat“, das zum Ziel hat, den Fahrer zu entlasten, als auch die Sicherheit zu erhöhen, auch wenn dies zugegebenermaßen nicht auf den “ersten Blick” ersichtlich ist.

Dieses System soll den Fahrer darin unterstützen, mit einer vorgewählten Geschwindigkeit zu fahren, so wie dies ein Tempomat macht. Sollte dies nicht möglich sein, z.B. weil ein vorausfahrendes Fahrzeug mit einer geringeren Geschwindigkeit fährt, passt das ACC die Fahrzeuggeschwindigkeit automatisch an, so daß ein zuvor ausgewählter Mindestab-

stand eingehalten wird. Dadurch wird der Fahrer durch die Abnahme von Routinearbeit entlastet.

Der Sicherheitsgewinn mit diesem System wird dadurch erreicht, daß während der „Schrecksekunde“ das ACC schon ein Bremsmanöver durchführt, und beim Eingriff des Fahrers das Bemssystem sich in einem vorkonditionierten Zustand befindet. Dadurch kann der Anhalteweg erheblich verkürzt werden, sowie die Aufprallgeschwindigkeit/Energie verringert werden.

ACC firmiert bei einigen Herstellern unter anderen Namen. Bei *Mercedes* ist dieses System unter den Namen *Distronic* bekannt, bei *BMW* unter den Begriff *Aktive Geschwindigkeitsregelung* (siehe [5]).

6.1.2 Beispielsituation

Die Funktionsweise des ACC-Systems soll an folgender Situation beispielhaft erklärt werden:

In diesem Fallbeispiel hat der Fahrer des grünen Autos sein ACC auf eine Geschwindigkeit von 140km/h eingestellt, sowie einen nicht näher beschriebenen Mindestabstand.

In der ersten Situation (oberes Drittel Bild 6.1) hat der Fahrer des grünen Fahrzeugs freie Fahrt, das ACC regelt die Fahrzeuggeschwindigkeit auf die gewünschten 140km/h.

In der zweiten Situation (mittleres Drittel Bild 6.1) schließt der Fahrer des grünen Autos zum langsameren blauen Auto auf. Das ACC erkennt, daß der Sicherheitsabstand bei Beibehaltung der Wunschgeschwindigkeit unterschritten würde, und passt deßhalb die Geschwindigkeit des grünen an die des blauen Autos an. Dies geschieht durch eine „Komfortbremsung“, die das ACC ausführt.

In der dritten Situation (unteres Drittel Bild 6.1) verläßt das langsam fahrende Fahrzeug die Straße, das ACC stellt fest, daß das grüne Auto freie Fahrt hat. Es beschleunigt selbständig das Fahrzeug, bis die eingestellte Wunschgeschwindigkeit von 140km/h wieder erreicht ist.

6.1.3 Bedienung

Die Bedienung ist, je nach Hersteller, teilweise ein wenig abweichend. Im großen und ganzen sind aber folgende Bedienelemente vorhanden bzw. Möglichkeiten auf das System Einfluß zu nehmen:

- Wird das Bremspedal betätigt, so wird das ACC abgeschaltet.
- Wird das Gaspedal betätigt, wird dadurch die Geschwindigkeitsregelung durch das ACC „überschrieben“. Sobald das Gaspedal aber wieder losgelassen wird, über-

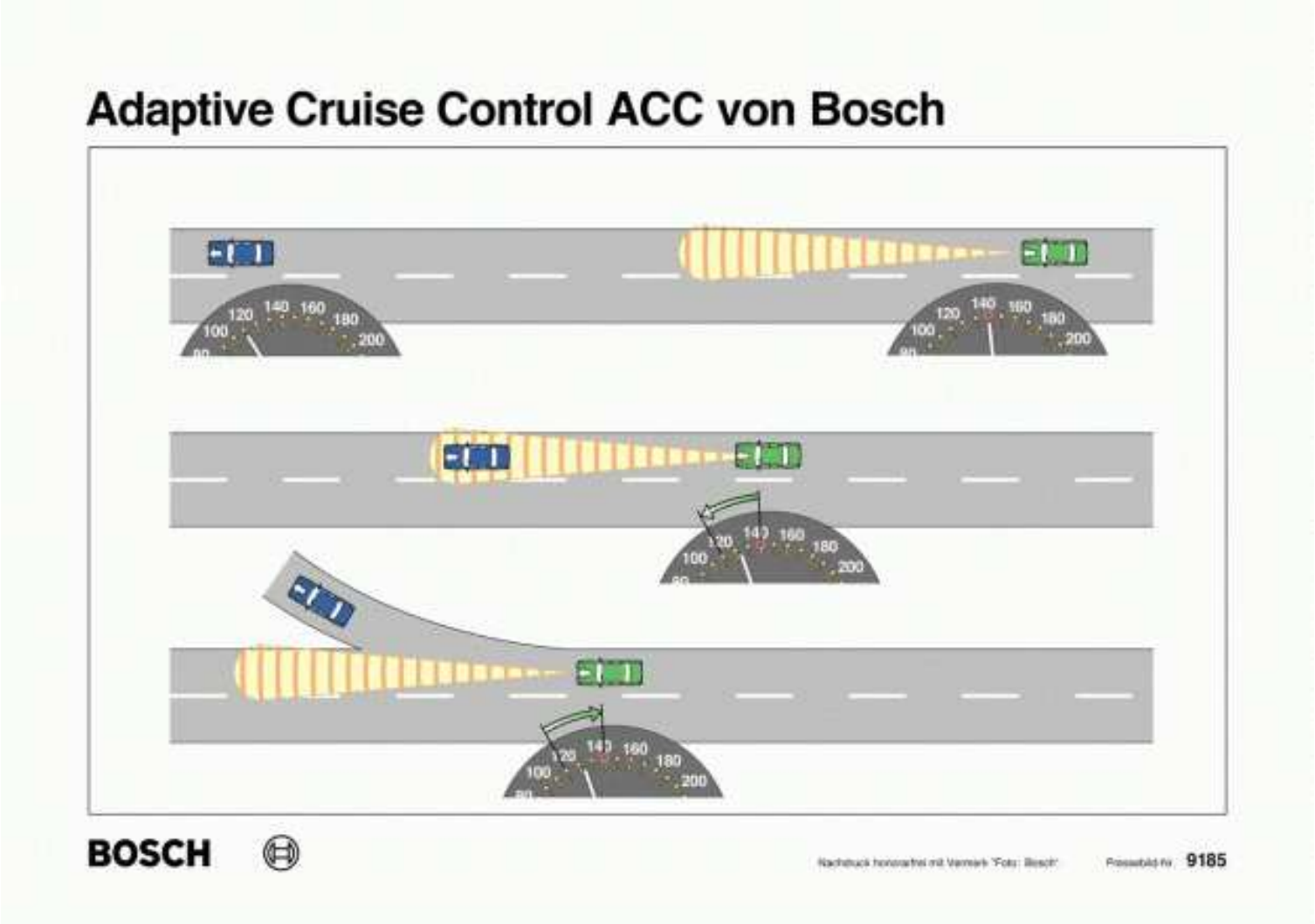


Abbildung 6.1: Beispielsituation

nimmt das ACC wieder mit den Parametern, die vor dem Betätigen des Gaspedals eingestellt waren.

- Die Wahl der Parameter erfolgt zumeist mit einem Lenkradhebel (ähnlich dem Blinkerhebel). Damit wird die Wunschgeschwindigkeit und der Wunschabstand eingestellt, der zumeist neben dem aktuellen Abstand im Cockpit angezeigt wird.

6.1.4 Grenzen des Systems

Der Einsatz von ACC ist momentan nur auf Autobahnen und Landstraßen möglich. Dies liegt zum einen daran, daß die zur Abstandsmessung verwendeten Sensoren nicht für den Einsatz im Nahbereich gedacht sind, d.h. für Situation, die typischerweise in der Stadt auftreten. Zum anderen ist die Komplexität der Situationen im Nahbereich viel größer,

gerade im Stop-and-Go Verkehr, so daß noch weitergehende Entwicklungsarbeit nötig ist. Entsprechende Systeme befinden sich bei den meisten Herstellern in der Entwicklung, ein Einführungstermin ist allerdings noch nicht abzusehen.

Der Geschwindigkeitsbereich, in dem das ACC arbeiten kann, erstreckt sich von ca. 30km/h bis ca. 180km/h, je nach Hersteller.

Die untere Geschwindigkeitsbegrenzung liegt wie oben erwähnt daran, daß die zu messenden Abstände den „Arbeitsbereich“ des Sensors verlassen.

Die obere Geschwindigkeitsbegrenzung ist eher aus psychologischen Gründen eingeführt, das System wäre auch jenseits der Grenze von 180 km/h funktionsfähig. Man will dem Fahrer auf diese Weise darbringen, daß er letztendlich die Verantwortung und Kontrolle über das Fahrzeug nicht vollständig abgeben kann.

Eine weitere Beschränkung liegt in der maximalen Verzögerung, die das ACC ausführen kann. Die inzwischen typische Grenze liegt im Bereich von $2-3\ m/s^2$, in dem sogenannte „Komfortbremsungen“ durchgeführt werden. Würde eine größere Verzögerung benötigt, warnt das ACC den Fahrer akustisch, damit er in dieser Situation wieder die Kontrolle über das Fahrzeug übernimmt.

Weitere Beschränkungen ergeben sich auch daraus, daß das ACC nicht alle Situationen richtig interpretieren kann. Dies kann zum Beispiel der Fall sein, wenn das ACC in der Spurmitte stehende Hindernisse geringer Abmessungen nicht erkennt oder bei Kurvenfahrten Objekte außerhalb der Fahrbahn als Hindernisse erkennt.

6.2 Technische Implementierung

Die technische Realisierung des ACC-Systems ist natürlich vom jeweiligen Hersteller abhängig. Hier wird näher auf die Realisierung von *Bosch* eingegangen, weil zum einen darüber etwas ausführlichere Informationen vorliegen (siehe [4]) und zum anderen die Systeme anderer Hersteller im Prinzip recht ähnlich aufgebaut sind.

6.2.1 Überblick

In der moderneren Generation der Fahrzeuge sind mehrere Steuereinheiten eingebaut, die untereinander zumeist mit dem CAN-Bus (Controller-Area-Network) vernetzt sind. Üblicherweise sind im Fahrzeug eine Motorsteuereinheit zu finden, sowie eine Steuereinheit, die für das ABS/ESP verantwortlich zeichnet. Die Steuereinheit für das ABS/ESP bezieht Daten von Sensoren, die die Geschwindigkeit, Gierrate (Winkelgeschwindigkeit) und die einzelnen Raddrehzahlen des Fahrzeugs mißt.

In dieses „Standardgerüst“ integrieren die Automobilhersteller eine (größtenteils von Zulieferern eingekaufte) ACC-Steuereinheit, die durch den CAN-Bus mit den restlichen Steuereinheiten vernetzt ist, um so auf deren Daten bzw. Funktionen zugreifen zu können.

6.2.2 Beispiel von Bosch

Hier ist auf Bild 6.2 nochmal ein Gesamtsystem aufgezeigt.

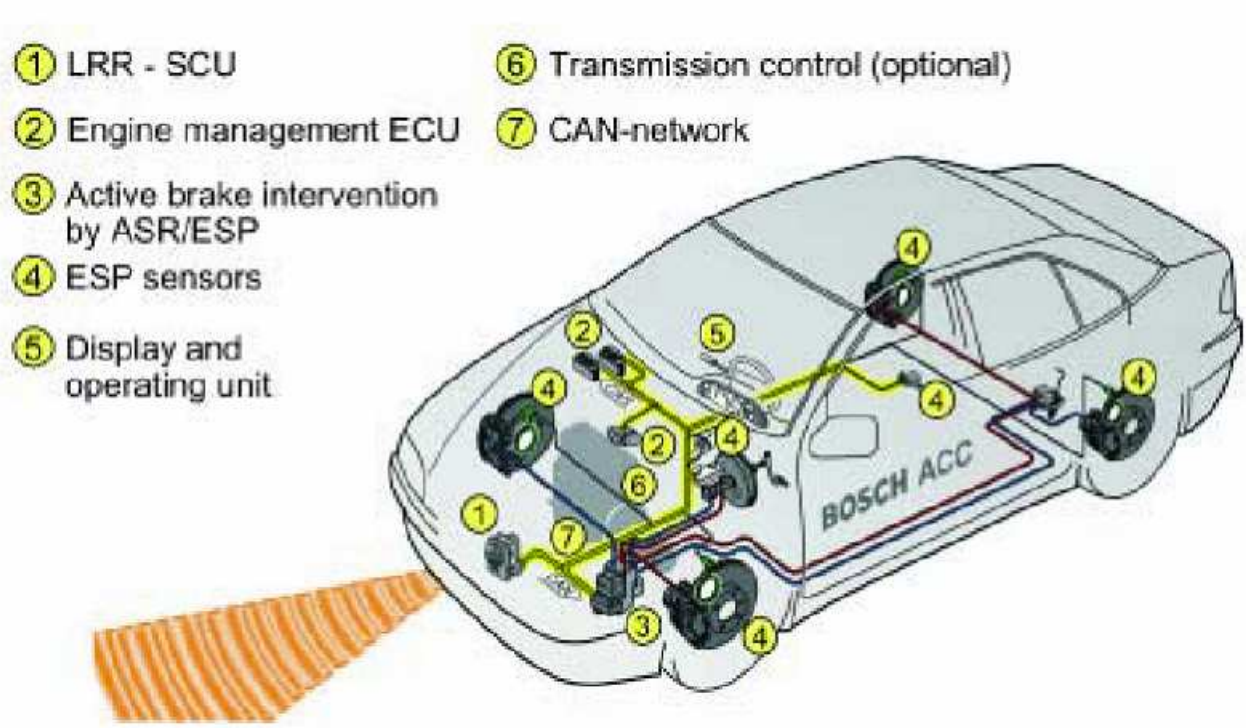


Abbildung 6.2: ACC von Bosch

Darin ist der CAN-Bus gelb eingezeichnet, der die unterschiedlichen Steuergeräte miteinander vernetzt:

- 1 LLR-SCU: Dies ist die ACC-Steuereinheit mit dem Sensor zur Erfassung des Abstandes und der Relativgeschwindigkeit zum voraus liegenden „Hindernis“ (LRR - SCU: Long Range RADAR - Sensor and Control Unit)
- 2 ECU: Dies ist die Motorsteuereinheit (Engine Control Unit)
- 3 ASR/ESP: Mit der Bremsanlage verkoppelte ESP-Steuereinheit (ASR/ESP: Anti Schlupf Regelung, Elektronisches Stabilitäts Programm)
- 4 ESP-Sensoren (Geschwindigkeit, Gierrate, Raddrehzahlen,...)
- 5 Dies ist die „Tachoanzeige“ im Cockpit des Autos
- 6 Getriebesteuerung
- 7 CAN-Netzwerk

Bei anderen Herstellern ist das Design ein wenig unterschiedlich, teilweise befinden sich Motorsteuereinheit und ESP in einer Einheit, teilweise sind Sensoren direkt am CAN-Bus

angeschlossen oder teilweise auch über einen eigenen getrennten Bus.

6.2.3 Eingesetzte Sensoren

Die Sensoren, die das ACC zur eigenen Geschwindigkeitsbestimmung verwendet, entnimmt es der ESP-Steuereinheit.

Zur Bestimmung des Abstandes sowie der Relativgeschwindigkeit des vorausfahrenden Fahrzeugs, werden üblicherweise LIDAR- und RADAR-Sensoren verwendet, auf die in den folgenden zwei Abschnitten näher eingegangen wird.

LIDAR

LIDAR steht für *Light Detection And Ranging*, Sensoren, die Licht im infraroten Bereich aussenden und das reflektierte Licht messen.

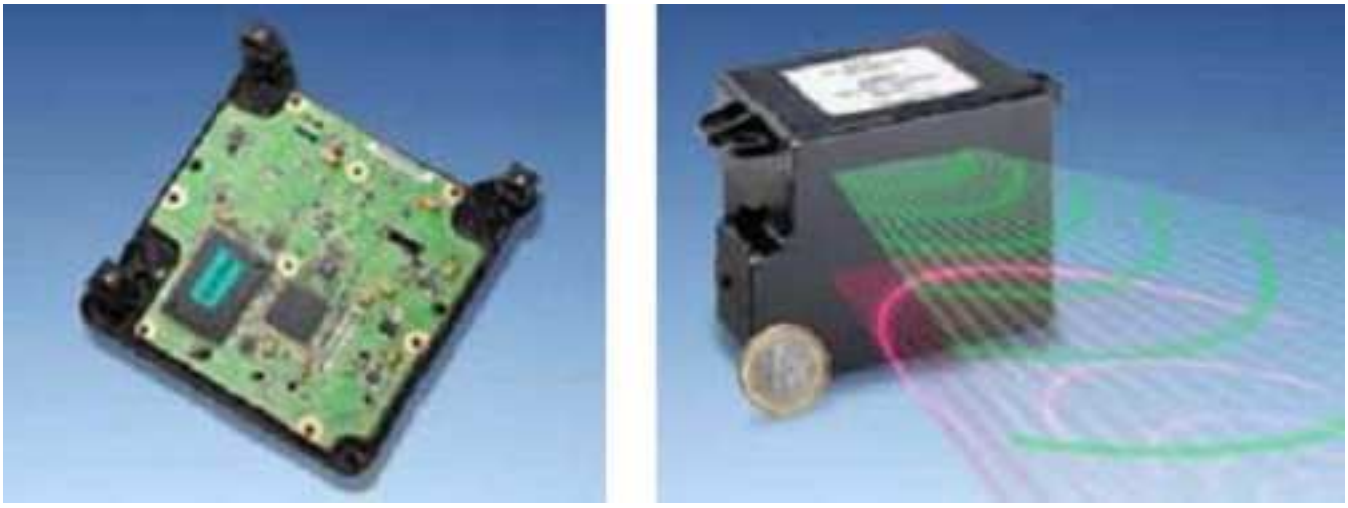


Abbildung 6.3: LIDAR-Sensor von Continental

Diese Geräte sind sehr kostengünstig herzustellen und für den Einbau in Mittelklasse-Wagen und Kleinfahrzeuge gedacht. Sie sind allerdings sehr empfindlich gegenüber Verschmutzungen (z.B. Dreck auf dem Lichtauslaß), als auch gegenüber dem Wetter (Regen, Nebel, Schnee). Diese Empfindlichkeit dem Wetter gegenüber soll allerdings nicht nur als Nachteil aufgefaßt werden! Es ist dadurch möglich, dem Fahrer eine den Sichtverhältnissen angepaßte Geschwindigkeit nahezulegen.

RADAR

RADAR steht für *Radio Detection And Ranging*. Diese Sensoren senden eine Funkwelle im Bereich von 77GHz aus und nehmen die von den Objekten reflektierten Signale über

eine Antenne auf.

Abbildung 6.4: RADAR-Sensor von Bosch

Diese Geräte sind sehr genau und sind im Vergleich zu den LIDAR-Sensoren viel weniger abhängig von Umwelteinflüssen wie Regen, Nebel, Schnee, aber auch gegen Verschmutzung der Sensoren. Sie können im Gegensatz zu den LIDAR-Sensoren „versteckt“ eingebaut werden. Üblicherweise werden gleich mehrere Sender und Antennen verbaut, die es ermöglichen den Winkel zum reflektierenden Objekt zu bestimmen.

Nachdem diese Geräte im Vergleich zu den LIDAR-Geräten recht teuer sind, werden sie vorerst nur in Autos der Oberklasse eingesetzt.

Bei den RADAR-Geräten ist außerdem ein Trend zur Miniaturisierung festzustellen.

6.2.4 Eingesetzte Aktoren

Das ACC hat eigentlich keine „eigenen“ Aktoren, sondern greift letztendlich über Funktionen der Motor-, Getriebe- und ESP-Steuereinheit auf die Aktoren zu, wie z.B. die Drosselklappensteuerung, sowie die Bremsen.

6.2.5 Meßverfahren

Um Geschwindigkeit und Abstand von Objekten zum eigenen Fahrzeug bestimmen zu können, sind zwei Verfahren maßgeblich: Das Puls-Doppler-Prinzip und das FMCW-

Verfahren (Frequency Modulated Continuous Wave).

Beim Puls-Doppler-Verfahren wird vom LIDAR/RADAR ein kurzer Impuls mit einer konstanten Frequenz gesendet. Durch die Laufzeit des Impulses, der von einem Objekt reflektiert wird und vom Fahrzeugsensor aufgenommen wird, kann auf den Abstand zum Objekt geschlossen werden ($x = \frac{1}{2}ct$, c Ausbreitungsgeschwindigkeit im Medium der EM-Welle, t Laufzeit des Impulses). Bewegt sich das Objekt relativ zum Fahrzeug, so ist durch den Doppler-Effekt die Frequenz des empfangenen Impulses verschoben. Diese Frequenzverschiebung ist direkt proportional zu der Relativgeschwindigkeit des Objekts ($\frac{\triangle f}{f_0} = -2\frac{v_{rel}}{c}$, $\triangle f$ Frequenzverschiebung, f_0 Frequenz des Senders, v_{rel} Relativgeschwindigkeit zum gemessenen Objekt, c Ausbreitungsgeschwindigkeit der Welle im Medium). Nachdem die Geschwindigkeit des eigenen Fahrzeugs bekannt ist, läßt sich dadurch die „Absolutgeschwindigkeit“ des Objektes bestimmen.

Die andere gebräuchliche Methode ist das FMCW-Verfahren. Dabei wird eine frequenzmodulierte Welle wie in Bild 6.5 dargestellt ausgesendet.

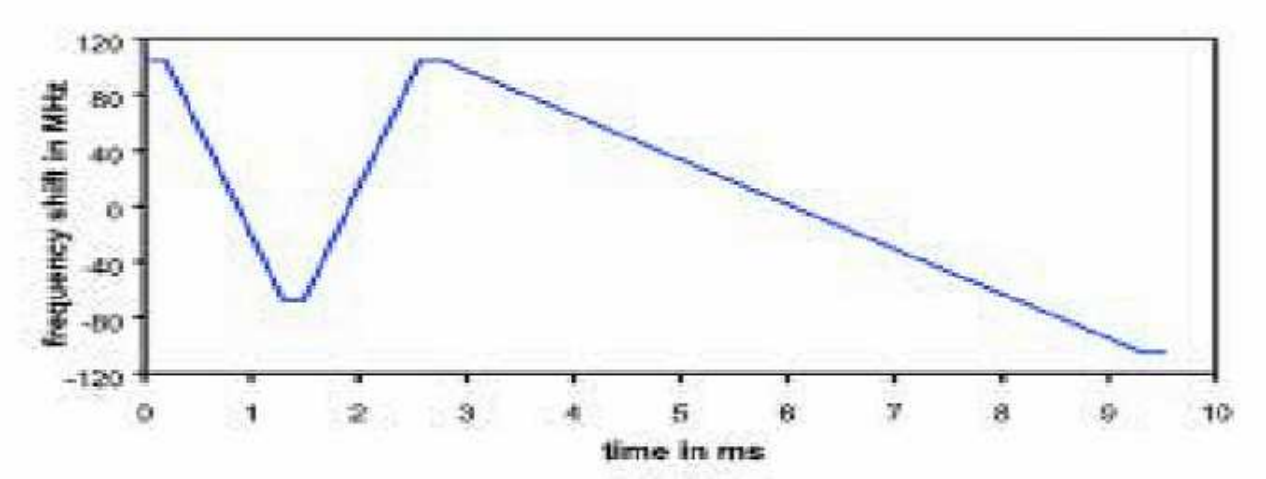


Abbildung 6.5: Beispiel eines FMCW-Signals

Die Auswertung des Signals findet nur im Frequenzbereich statt.

Als erstes soll der Fall 1 (siehe Bild 6.6) betrachtet werden, in der die Frequenz des gesendeten Signals steigt. In diesem Fall ist die Differenzfrequenz f_{delta} zwischen der gerade gesendeten Frequenz $f_{Sender}(t)$ und der Frequenz $f_{Objekt1}(t)$ eines stehenden Objekts direkt proportional zur Signallaufzeit, also auch zum Abstand. Ist das Objekt bewegt, ist die empfangene Frequenz um die Dopplerfrequenz f_d verschoben. Man erhält also folgende Beziehung:

$$f_{Objekt1}(t) = f_{Sender}(t) - f_{delta} - f_d$$

In Fall 2 (siehe Bild 6.7) fällt die Frequenz des gesendeten Signals mit der gleichen Steigung wie im Fall 1. In diesem Fall ist die Differenzfrequenz f_{delta} zwischen der gerade gesendeten Frequenz $f_{Sender}(t)$ und der Frequenz $f_{Objekt2}(t)$ eines stehenden Objekts direkt proportional zur Signallaufzeit, also auch zum Abstand. Ist das Objekt bewegt, ist

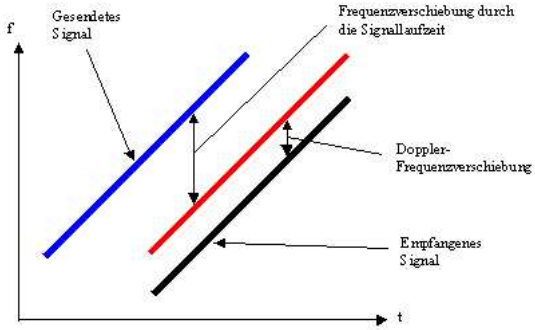


Abbildung 6.6: Fall 1

die empfangene Frequenz um die Dopplerfrequenz f_d verschoben. Man erhält also folgende Beziehung:

$$f_{Objekt2}(t) = f_{Sender}(t) + f_{delta} - f_d$$

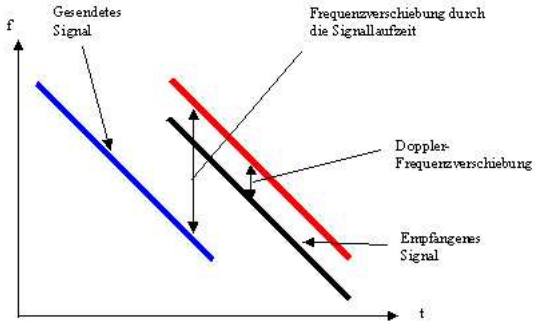


Abbildung 6.7: Fall 2

Addiert bzw. subtrahiert man die Gleichungen aus Fall 1 und Fall 2 voneinander erhält man:

$$f_{Objekt2} - f_{Objekt1} = 2f_{delta}$$

$$f_{Objekt2} + f_{Objekt1} = 2f_{Sender} - 2f_d$$

Um diese Berechnungen durchführen zu können, werden die ins Basisband transformierten Antennensignale FFT-transformiert. Es findet außerdem ein Vergleich der komplexen Amplituden des Spektrums statt, um den Winkel zu bestimmen, aus dem die Wellen reflektiert wurden.

6.2.6 Eingesetzte Hardware

Nachdem hier ausführlicher auf das System von Bosch eingegangen worden ist, soll zuerst ein Überblick über deren verwendete Hardware gegeben werden.

Die Hardware ist dabei in einen HF-Block und einen NF-Block unterteilt.

Der HF-Block enthält den Frequenzgenerator für die RADAR-Sender, einen PLL-ASIC zum Ansteuern des Frequenzgenerators, sowie einen Vorverstärker für die basisbandtransformierten Antennensignale.

Der NF-Block enthält einen ADC-ASIC (15MHz Sampling-Frequenz) zum digitalisieren der Antennensignale. Diese werden von einer TI-Centaurus-IC verarbeitet. In diesem Centaurus-IC sind ein DSP zur Signalverarbeitung enthalten (ein TMS320, 40MHz, 48kB RAM), sowie ein TMS470 Mikroprozessor, der die eigentlich ACC-Steuereinheit ist. Dieser Mikroprozessor enthält einen mit 40MHz getakteten ARM7 Kern, sowie 64kB RAM. Ebenfalls auf diesem Centaurus-IC enthalten sind CAN-Controller, serielle Schnittstellen und ein 768kB großes EEPROM.

Eine Vielzahl nahmhafter Halbleiterhersteller bieten spezielle auf den automobilen Bereich ausgerichtete Plattformen an, insbesondere für ACC bzw. den Einsatz mit Kameras.

Von Motorola wird z.B. die PowerPC-Familie mpc500 speziell für dieses Einsatzgebiet beworben, eine Beispielkonfiguration ist in Bild 6.8 zu sehen.

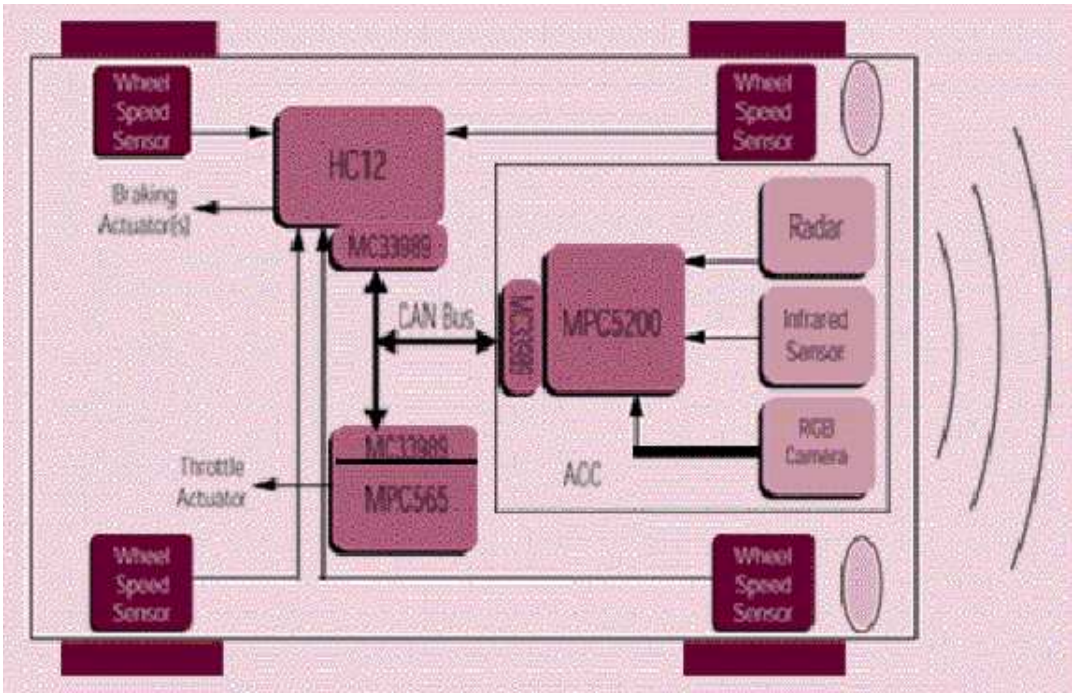


Abbildung 6.8: MPC500-Plattform von Motorola

6.3 Überlegungen zur Realzeitfähigkeit

In diesem Kapitel werden einige Gedankengänge zur Realzeitfähigkeit der ACC-Systeme aufgeführt. Dies hat den Hintergrund, daß sowohl die Automobilhersteller, als auch die Zulieferer keine oder nur wenige konkrete Informationen diesbezüglich herausgeben.

Folgende Größen können als kritisch für die Echtzeitanforderungen betrachtet werden:

- die maximale Geschwindigkeit des Fahrzeugs
- das räumliche Auflösungsvermögen des RADARS
- die Periodendauer eines FMCW-Signals
- die Samplig-Frequenz des ADC
- die Berechnungszeit für die Signalvorverarbeitung
- die Ausführungszeit des ACC-Programmes
- die Zeit, um über den CAN-Bus eine Bremse „anziehen“ zu lassen

Bis auf die ersten drei Punkte sind von den Herstellern keine Informationen erhältlich.

6.3.1 Abschätzung für das BOSCH-ACC

Man könnte also eine Durchlaufzeit der Daten durch den ADC inclusive FFT mit ungefähr 870ns schätzen (Periodendauer des ADC + 32 DSP-Takte). Wenn man die Ausführungszeit des ACC-Programmes auf 1000 Taktzzyklen schätzt, kommt man auf 25μs. Man könnte die Gesamtdauer also „guten Gewissens“ auf 11ms schätzen, bis ein Bremsbefehl vorliegt. In diesen 11ms würde man bei 200km/h gerade mal 6m zurücklegen, was in etwa in der Größenordnung des Auflösungsvermögen gängiger RADAR-Systeme liegt. Auch ansonsten ist die Reaktionsdauer des Systems vergleichsweise gering, wenn man sie mit der „Schrecksekunde“ des Menschen vergleicht...

6.3.2 Vergleich mit einer Kamera-Lösung

Zum Schluß soll noch ein Vergleich mit einer funktional gleichen Lösung gemacht werden, die ein Stereo-Kamerasystem statt RADAR/LIDAR-Sensoren verwendet. Diese wurde ausführlicher in einer Master-Arbeit von Jason R. Bergendahl erläutert (siehe [1]). Die gewonnenen Resultate sind allerdings mit Vorsicht zu genießen, nachdem diese Lösung eine frühe Studie ist.

Die Verarbeitungszeiten auf einen PC, sowie einem vorgeschalteten DSP liegen zusammen bei 29ms, wobei sehr einfache Berechnungen auf einem 512x80 Pixel großen Bild ausgeführt werden. In diesem Zusammenhang ist eine andere Größe maßgeblich, nämlich die Bildrate der Kamera! Diese würde bei einer guten Bildrate von 50Hz eine „Totzeit“

von 20ms induzieren. Es würden also ca. 50ms vergehen, bis ein Bremsbefehl vorliegt, also gut fünfmal so lang wie ein RADAR-System benötigt!

Literaturverzeichnis

[1] Jason R. Bergendahl. A computationally efficient stereo vision algorithm for adaptive cruise control. Master's thesis, Massachusets Institute of Technology, 1997.

[2] Björn Filzeck. *Abstandsverhalten auf Autobahnen - Fahrer und ACC im Vergleich*, volume 12 of *VDI-Fortschritts-Bericht*. VDI-Verlag, 2003.

[3] Dipl.-Ing. Michael Klotz. *An Automotive Short Range High Resolution Pulse Radar Network*. PhD thesis, Technische Universität Hamburg-Harburg, 2001.

[4] Dr. Götz Kühnle; Dipl.-Ing. (FH) Hermann Mayer; Dr. Herbert Olbrich; Dr. Wolf Steffens und Dipl.-Ing. Hans-Christian Swoboda; Robert Bosch GmbH. Low-cost long-range radar for future driver assistance systems. *AutoTechnology*, 4 2003.

[5] BMW / Audi / DaimlerChrysler; Delphi; Bosch; Continental; Visteon. Produktbeschreibungen. Internet.

[6] Markus Weinberger. *Der Einfluss von Adaptive-cruise-control-Systemen auf das Fahrerverhalten*. PhD thesis, Technische Universität München, 2000.

Kapitel 7

Fly-By-Wire

Autor: Thomas Heise

Betreuer: Dipl.-Ing. Sebastian Drössler

7.1 Einleitung

In den Anfängen der Luftfahrt wurden die Steuerbewegungen des Piloten mit Hilfe von Seilzügen und Umlenkrollen direkt an die Steuerflächen des Flugzeuges weitergegeben. Die Piloten mussten dabei direkt gegen die Windkräfte an den Rudern arbeiten. Bei größeren Flugzeugen reichten die Steuerkräfte des Piloten nicht mehr aus, um die Steuerruder und somit das Flugzeug zu kontrollieren. Daher sind Flugzeuge mit hydraulischen Kraftverstärkern ausgestattet, die die Bewegungen vom Steuerhorn und von den Pedalen direktproportional an die Ruder weitergeben. In den frühen 1980´ern, wurden die ersten Flugzeuge mit Fly-by-Wire (FBW) Systemen in Dienst gestellt. Durch ein FBW System ist der Pilot nicht mehr länger mechanisch mit den Rudern verbunden. Vielmehr liegt zwischen dem Steuerknüppel und den Steuerflächen das Electronic Flight Control System (EFCS), welches die Steuerdrücke des Piloten gemäß der aktuellen Fluglage und -geschwindigkeit interpretiert und anschließend entsprechende Signale an die mechatronischen Aktuatoren in den Steuerflächen weitergibt.

Die amerikanische F-16 war das erste Flugzeug, bei dem die FBW-Technik erfolgreich eingesetzt wurde. Gerade im militärischen Bereich eröffnet FBW neue Möglichkeiten: So mussten bis zur Einführung von FBW, Flugzeuge aerodynamische stabil sein, damit sie für den Piloten steuerbahr sind. Moderne Kampfjetzt wie die F-16 oder der Eurofighter sind dagegen aerodynamisch instabil ausgelegt, was ihnen eine größere Dynamik verleit. Aerodynamisch instabil bedeutet, dass das Flugzeug ohne ständige Steuereingriffe in einen unstabilen Flugzustand übergehen kann. Es ist aber für einen Piloten nahezu unmöglich, derartige Steuereingriffe schnell genug durch zuführen, diese Aufgabe übernimmt das EFCS.

Zivilflugzeuge wie der Airbus A-320, die per FBW geflogen werden, sind aerodynamisch stabile Flugzeuge. Beim A-320 als das erste zivile Flugzeug mit FBW sprechen andere Gründe wie Gewichtsreduzierung durch den Entfall der Steuerstangen und Hydraulikleitungen, für die Markteinführung als Fly-by-Wire Flugzeug. Weill sich da durch ein besseres Verhältnis von Nutzlast zu Treibstoff ergibt, wodurch diese Flugzeuge wirtschaftlicher sind.

7.2 Flugregelung

Aufgabe der Flugregelung ist es, die Stabilisierung und Steuerung des Flugzeuges zu gewährleisten. Dabei bewegt sich das Flugzeug im Raum. Um die Bewegungen des Flugzeuges im Raum beschreiben zu können, wird ein flugzeugfestes Koordinatensystem definiert (Abbildung 7.1)[5]. Man kann dabei grob sagen, dass das Flugzeug mit den Querrudern um die Längsachse gerollt wird, mit dem Seitenruder um die Hochachse giert und mit den Höhenrudern um die Querachse nickt. Die Flugregelegung hat sich schrittweise entwickelt von der rein manuellen Steuerung unter Verwendung von mechanischen Übertragungselementen bis zum weitgehend automatischen Flug mit einem EFCS.

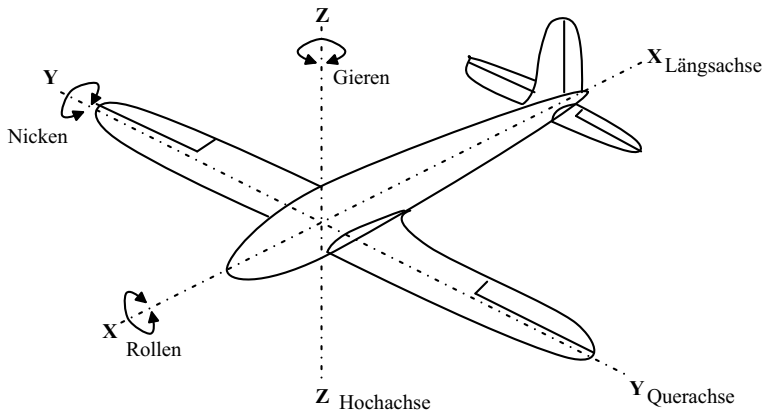


Abbildung 7.1: Flugzeugfestes Koordinatensystem

7.2.1 Manueller Flug mit teilweiser Reglerunterstützung

Bei manueller Steuerung (ohne hydraulischer Kraftverstärkung) hat der Pilot direkt die Ruderkraft gegen Massen- und Luftkräfte aufzubringen. Diese ist vom Staudruck und von der Schwerpunktlage abhängig und liefert insofern eine für den Piloten wichtige Rückführgröße zur Beurteilung des aktuellen Flugverhaltens. Um aber ständigen Kraftaufwand zu vermeiden, kann der Pilot über ein selbsthemmendes Getriebe (Trimmeinrichtung) die Höhenflosse rückwirkungsfrei verstellen und damit das Momentengleichgewicht herstellen, d.h. das Flugzeug am Arbeitspunk der aktuellen Fluggeschwindigkeit austrimmen [2].

Abbildung 7.2 zeigt eine häufig anzutreffende Konfiguration, bei der die Steuerung mit dem Höhenruder erfolgt und die längerfristige Trimmung über die Leitwerksflosse. Bei ausgetrimmtem Flugzeug ist das Ruder in Nullstellung kraftfrei und der Steuerknüppel in Mittelstellung.

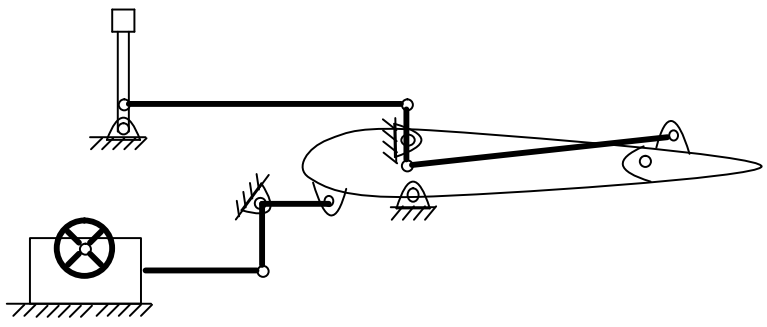


Abbildung 7.2: Manuelle Nicksteuerung und Nicktrimmung.

Mit der Flugzeuggröße und der Fluggeschwindigkeit wächst auch die Ruderkraft, die Knüppelkraft muss durch eine hydraulische Kraftverstärkung reduziert werden. Die jetzt

entfallende Ruderrückwirkung wird durch eine Feder ersetzt, die es erlaubt, am Steuerknüppel eine sinnvolle Zuordnung von Knüppelausschlag und -kraft zu bekommen und somit dem Piloten eine Abschätzung der Wirksamkeit seiner Streuerausschläge ermöglicht. Abbildung 7.3 zeigt diese Anordnung.

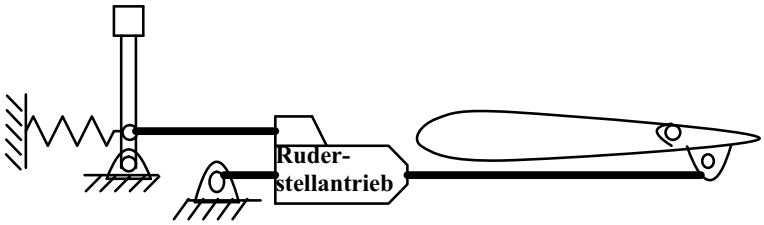


Abbildung 7.3: Hydraulischer Ruderstellantrieb

Um den Piloten vom ständigen Ausregeln von Böenstörungen zu entlasten und gleichzeitig die Dämpfung der Flugzeug-Eigenbewegung zu verbessern, wird das Stability Augmentation System (SAS) zur Dämpfung eingesetzt [2]. Da das Dämpfungsmoment proportional zur Geschwindigkeit ist, lässt sich über regelungstechnische Maßnahmen eine Dämpfungserhöhung erzielen, indem die Geschwindigkeit gemessen und auf eine Stellgröße zurückgeführt wird. Somit wird ein künstliches Moment erzeugt und der natürlichen Dämpfung überlagert. Dies soll am Beispiel eines Nickdämpfers dargestellt werden. Der Dämpfungskoeffizient der Anstellwinkelschwingung ist proportional der Nickdämpfung. Führt man die Nickgeschwindigkeit q derart auf das Höhenruder zurück, dass ein Aufnicken des Flugzeuges (q>0) zu einem positiven Höhenruderausschlag n führt, ergibt sich ein rückführendes Moment und damit eine zusätzliche künstliche Dämpfung der Nickbewegung.

Wie sich ein Dämpfungs-Regelkreis in der Zusammenwirkung mit dem Piloten beim Führen eines Flugzeugs auswirkt, wird in (Abbildung 7.4) dargestellt. Das SAS hat nur untergeordnete Aufgaben. Lagestabilisierung und Bahnführung bleiben weiterhin Aufgabe des Piloten. Dazu muss eine direkte Verbindung zwischen Steuerknüppel und Höhenruder bzw. Ruderstellantrieb bestehen bleiben.

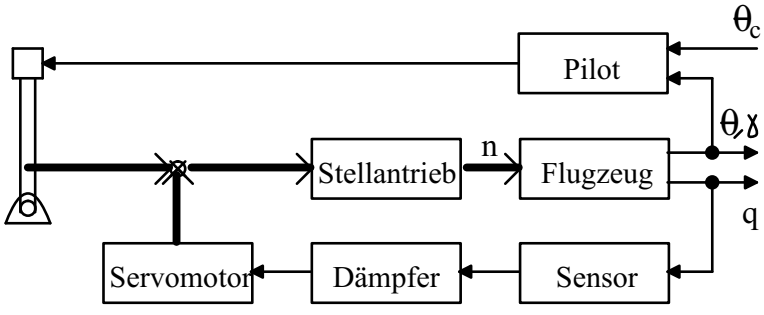


Abbildung 7.4: Dämpfer-Regelkreis

Abbildung 7.5 zeigt ein mögliche Ausführung der Dämpfung, bei der der Stellantrieb für

die Dämpfung in das Gestänge mit einbezogen wird. Hierbei wird das Stellsignal in eine Längenänderung des Gestänges umgeformt.

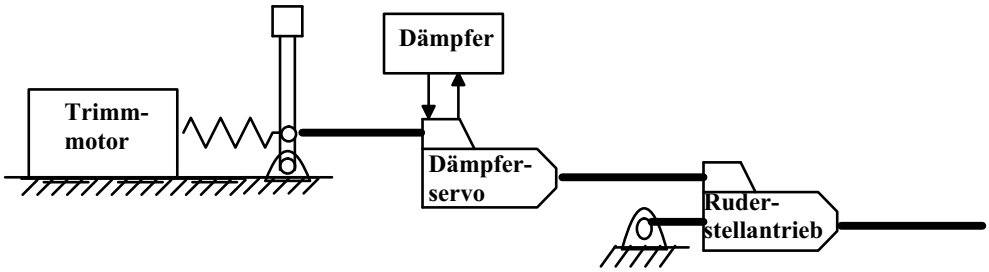


Abbildung 7.5: Dämpferservo in Reihenanordnung

7.2.2 Steuerung bei Autopilotenbetrieb

Die Funktionsweise eines Autopiloten soll beispielhaft an einer Änderung der Flughöhe erläutert werden. Wird eine neue Sollhöhe vorgegeben, wird in dem Führungsgrößengenerator der Sollbahnverlauf in einem mathematischen Modell generiert. Diese Sollbahn besteht aus einer Übergangsphase mit begrenztem Lastvielfachen.

Der Autopilot erhält als Führungssignal den jeweiligen Höhensollwert und regelt Abweichungen entsprechend aus (Abbildung 7.6). Der Pilot gibt lediglich den neuen Höhensollwert und Zeitpunkt des Manövers ein.

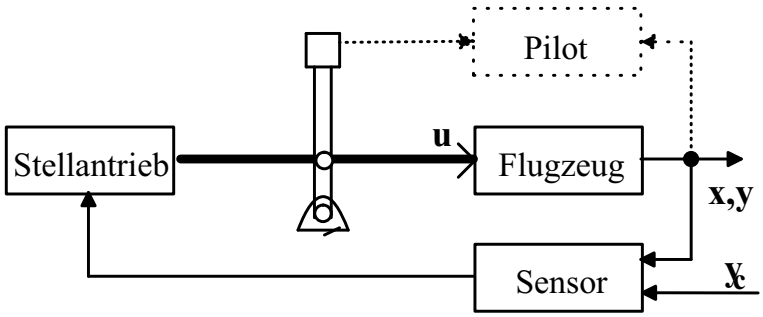


Abbildung 7.6: Autopilot-Regelkreis

Der Autopilot gibt seine Signale über eine Kupplung an das Steuergestänge weiter und steuert somit das Flugzeug eigenständig. Das Bedienelement des Piloten bewegt sich mit, so dass der Pilot die Steuerung überwachen kann.

In den Anfängen der Flugregelung wurden Autopiloten nachträglich in ein fertiges Flugzeug eingebaut. Die einfachste und sicherste Art der Nachrüstung bestand in einer Verbindung des Stellmotorausgangs an bestehenden Seilzügen oder Gestängen über eine elek-

trisch betätigte Kupplung, die im Notfall vom Steuerknüppel her überbrückt werden konnte. Die Forderung, dass die mechanische Verbindung zwischen Steuerknüppel und Ruder stets aufrechterhalten wird, hatte zur Folge, dass der Steuerknüppel vom Autopilotenstellmotor mitbewegt wird (Abbildung 7.6). Diese bis heute übliche Anordnung macht eine Mischung von Piloten- und Autopilotenkommandos unmöglich und der Pilot muss die Steuerung in den betreffenden Freiheitsgraden vollständig an den Autopiloten abgeben. Dem steht der große Vorteil gegenüber, dass der Pilot die Tätigkeiten des Autopiloten jederzeit am Steuerknüppel ablesen kann und bei Fehlverhalten des Reglers sofort und unmittelbar eingreifen kann, indem er den Autopiloten ausschaltet und selbst das Fluzeug übernimt.

Der Autopilot wird ergänzt durch einen Trimmregler, der aus Sicherheitsgründen meist als unabhängiges Gerät ausgeführt wird. Dieser erhält vom Autopiloten und/oder Piloten Eingangssignale. Die Trimmung hat auch hier die stationäre Ruderlast aufzunehmen und dafür zu sorgen, dass stets der volle Arbeitsbereich des Ruders für den Autopiloten zur Verfügung steht, so dass der Autopilotenservo keine stationäre Last aufzunehmen hat und bei Abschalten oder Ausfall des Autopiloten keine großen Ruderausschläge entstehen [2].

Auch bei automatischer Trimmung läuft das Trimmrad im Cockpit zwecks direkter Überwachung mit. Diese Ausführungen beziehen sich in der Hauptsache auf die Trimmung in der Nickachse. Es gibt auch Vorrichtungen zur Trimmung von Seiten- und Querruder.

7.2.3 Fly-by-Wire

Die Regelungsaufgaben der Flugführung können automatisch oder manuell erfüllt werden. Anzustreben ist eine Aufteilung, die den jeweiligen Fähigkeiten von Mensch und Maschine angemessen ist. Die größte Freiheit bei dieser Aufteilung hat man, wenn die mechanischen Verbindungen zwischen den Bedienelementen und den Steuerflächen durch eine FBW-System ersetzt wird (Abbildung 7.7). Eine FBW-Steuerung ist auch als Vorgaberegelung zu bezeichnen. Für eine Vorgabereglung kann man Laut Literatur [2] zwei

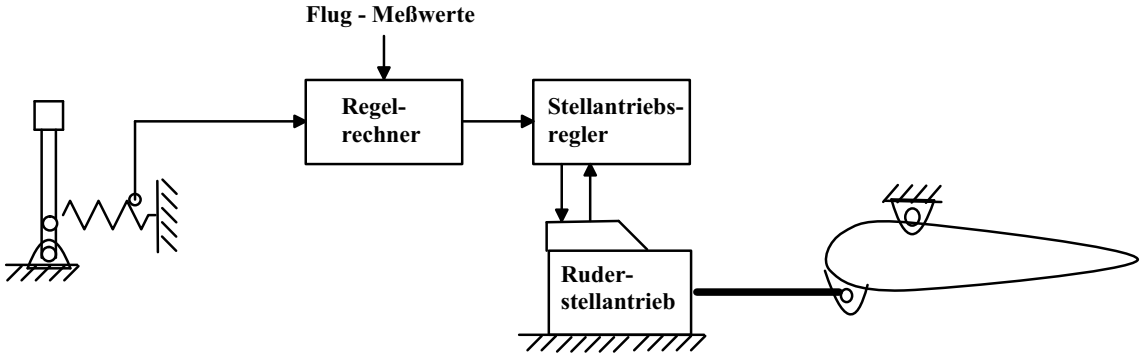


Abbildung 7.7: Fly-by-Wire Regelkreis

Aufgabenschwerpunkte definieren:

- Flugeigenschafts-Vorgaberegler modifizieren die höherfrequenten (rotatorischen) Bewegungsanteile und dienen der Entkopplung und Störunterdrückung beim Manövrieren. Typische Regelgrößen sind Drehgeschwindigkeit und Beschleunigungen. Diese Art der Vorgabereglung ist manuelle Basisbetriebsart bei fast allen realisierten FBW-Flugsteuerungen (z.B. A-320, X31-A, Space-Schuttle).
- Bahnführungs-Vorgaberegler ermöglichen den Piloten die direkte Vorgabe bahnbestimmender Größen über seine primären Bedinelemente bei gleichzeitiger Erledigung aller untergeordneten Reglungsaufgaben durch einen Flugrechner. Typische Regelgrößen sind Translationsgeschwindigkeiten und Lagewinkel. Die Regelungsaufgabe umfasst die niederfrequenten Bewegungen und ist mit einem Bahnregelkreis (Autopiloten) vergleichbar.

Bei FBW tritt der Regler zwischen Pilot und Flugzeug (Abbildung 7.7). Der Pilot fliegt das Flugzeug durch den Regler hindurch, wobei die Zuordnung zwischen Steuerknüppelkraft und Regelgröße den Pilotenwünschen angepasst werden kann. Der Regler modifiziert und vereinheitlicht die Flugzeugreaktion in weiten Grenzen.

7.3 Anforderungen an ein Electronic Flight Control System

Die Auslegung des EFCS eines FBW-Flugzeug unterliegt drei Faktoren [6]:

- Anzahl und Komplexität der zu bearbeitenden Funktionen.
- Der Dynamik des zu regelnden Systems.
- Flugsicherheit

7.3.1 Funktionen

FBW Control Systeme haben folgende Funktionen zu erfüllen [6] [2]:

- Automatischer Flug entlang einer vorgebenden Trajektorie.
- Manueller Flug: Hierbei folgt das Flugzeug den Kommandos über den Sidestick und den Pedalen. Der Pilot muss dabei nicht auf die aerodynamischen und strukturellen Grenzen des Flugzeugs achten, diese Aufgabe übernimmt das EFCS.
- Eingeschränkter Flug: Im Falle eines Systemausfalls sind die primären Steuereinrichtungen weiterhin in Funktion. Der Pilot kann das Flugzeug mit erhöhtem Mehraufwand weiter fliegen und landen.

Ein EFCS muss eine Ausfallsrate kleiner $10^{-7}/Fh$ (Flight hours) für militärische Flugzeuge und kleiner $10^{-10}/Fh$ für zivile Flugzeuge haben, diese gilt auch für FBW-Flugzeuge.

Andereseits haben die Funktionen und Möglichkeiten, die ein FBW-Flugzeug bietet zugenommen. Es wird daher gefordert, dass sich die einzelnen Systeme selbst kontrollieren und gegebenenfalls selbstständig abschalten und die Aufgaben an ein redundantes System übergeben. Das Flugzeug muss im schlimsten Fall, das heist beim Ausfall mehrere Systeme, weiterhin durch eine minimale Notsteuerung bedienbar bleiben.

Ein FBW-Sytem muss folgende Tasks bearbeiten [6] [2]:

- Erfassung der eingehenden Daten, bestehend aus den Sensordaten und den Eingabebefehlen des Piloten, das sind im einzelnen:
  - Beschleunigungswerte entlang der Roll-, Nick- und Gier-Achse translatorisch wie auch rotatorisch.
  - Airdaten: Statischer und dynamischer Luftdruck. Luftströmung entlang der Längsachse.
  - Pilotenkommandos

  Diese Daten werden idealerweise noch im Sensor Analog-Digital gewandelt und anschließend von einem zentralen Rechner ausgewertet.
- Korrigieren von Sensorfehlern: Dies betrifft vor allem die Airdaten und Lufströmungen. Sie werden durch das Flugzeug und seinen Triebwerken verfälscht. Durch eine Fehlerrechnung können diese Einflüsse herausgerechnet werden.
- Reglerfunktionen zur Stabilisierung und Dämpfung des Flugzeuges entlang der drei Achsen.
- Redundanz Management: Jeder einzelne Datenverarbeitungsschritt ist durch eine Monitoring Einheit zu überprüfen und im Falle einer Unstimmigkeit ist die Weiterverarbeitung fehlerhafter Ergebnisse zu verhindern. Ist ein fehlerhaftes System entdeckt worden, so wird es ausgeschaltet und durch ein Redundanzsystem zu ersetzen.

7.3.2 Verteilte Rechner Struktur

Um die große Anzahl an Aufgaben zu erfüllen, die an ein modernes FBW-Steuerungs System gestellt werden und gleichzeitig ein größtes Maß an Redundanz zu haben, besteht ein FBW-System aus einer Anordnung eigenständiger Steuergeräten. Jedes Steuergerät besteht neben der CPU aus eine eigenen lokalen RAM und ROM Speicher und besitzt sein eigenes Betriebsystem. Der interne Speicher kann nur von der eigenen CPU adressiert werden. Zugriffe von außen sind nicht möglich. Dies verhindert, dass sich Softwarefehler in einem Rechner nicht auf die anderen Systeme auswirken können [6].

7.4 Electrical Flight Control System Airbus A-320

Seit 1988 ist des Airbus A-320 im Einsatz. Die Standardausführung der A-320 ist für 164 Passagiere ausgelegt. Im Cockpit sind die zwei Steuersäulen durch Sidesticks ersetzt, die Instrumente im Cockpit wurden verringert und durch eine Bildschirmanzeige ersetzt. Die wichtigste Neuerung ist jedoch der Ersatz der mechanischen Seilübertragung von den Steuersäulen zu den hydraulischen Kraftverstärkern an den Rudern. Um eine Steuerbarkeit des Flugzeuges auch bei Ausfall der Elektronik zu gewährleisten, wurde die mechanische Seilübertragung von den Pedalen zu den Seitenrudern beibehalten. Im Notfall ist in der Nickachse die mechanische Trimmung der Höhenruderflosse als Steuerungsmöglichkeit für die Längsachse vorhanden. Die FBW Steuerung erlaubt die Rechnerunterstützung zur Verbesserung der Flugeigenschaften und die Begrenzung der Flugmanöver zur Vermeidung von kritischen Flugzuständen auch bei einer Handsteuerung durch die Piloten [5][2]. Wie bei der primären Steuerung, so ist auch die Seilübertragung von den Gashebeln zu den Treibstoffventilen durch eine digitale Signalübertragung ersetzt worden. Das EFCS besteht aus (Abbildung 7.8):

- Höhen- und Querruderrechner ELAC (Elevator Aileron Computer)
- Gierdämpfungsrechner FAC (Flight Augmentation Computer)
- Störklappen-Höhenruder-Rechner SEC (Spoiler Elevator Computer)
- Vorflügel- und Klappenrechner SFCC (Slate Flaps Control Computer)
- Flugwegrechner FMGC (Flight Management and Guidance Computer)

7.4.1 Steuerung über die seitlichen Steuergriffe (Sidestick)

Die Steuerung des Flugzeuges durch die Piloten erfolgt über die seitlich angebrachten Steuergriffe. Diese sind in der Nullstellung federgefesselt und geben bei einer Auslenkung ein Stellungssignal an den Höhen- und Querruderrechner ELAC ab. Vom ELAC-Rechner werden die elektro-hydraulischen Kraftverstärker der beiden Höhenruder und der Querruder gesteuert. Außerdem wird noch der elektrische Trimmmotor für die Höhenruderflossenverstellung vom Rechner angesteuert. Parallel zum ELAC wird auch der Störklappen-Höhenruder-Rechner SEC vom Sidestick aktiviert. Die Höhen- und Querruderausschläge werden vom ELAC-Rechner entsprechend des aktuellem Flugzustandes und der Steuergriffauslenkung veranlasst, sie sind daher nicht proportional der Steuergriffauslenkung [5][2].

In der Nickachse ist die automatisch Steuerung des Flugweges in der Vertikalen mit einem konstanten Wert der Normalbeschleunigung von 1 g über dem Wert der Auslenkung der Sidesticks vorgesehen. Diese Regelung ermöglicht eine sichere Steuerung durch den Piloten und verhindert unerlaubte Flugzustände.

Die Flugsicherheit wird durch weitere Vorgaben von Grenzwerten im Rechner erhöht,

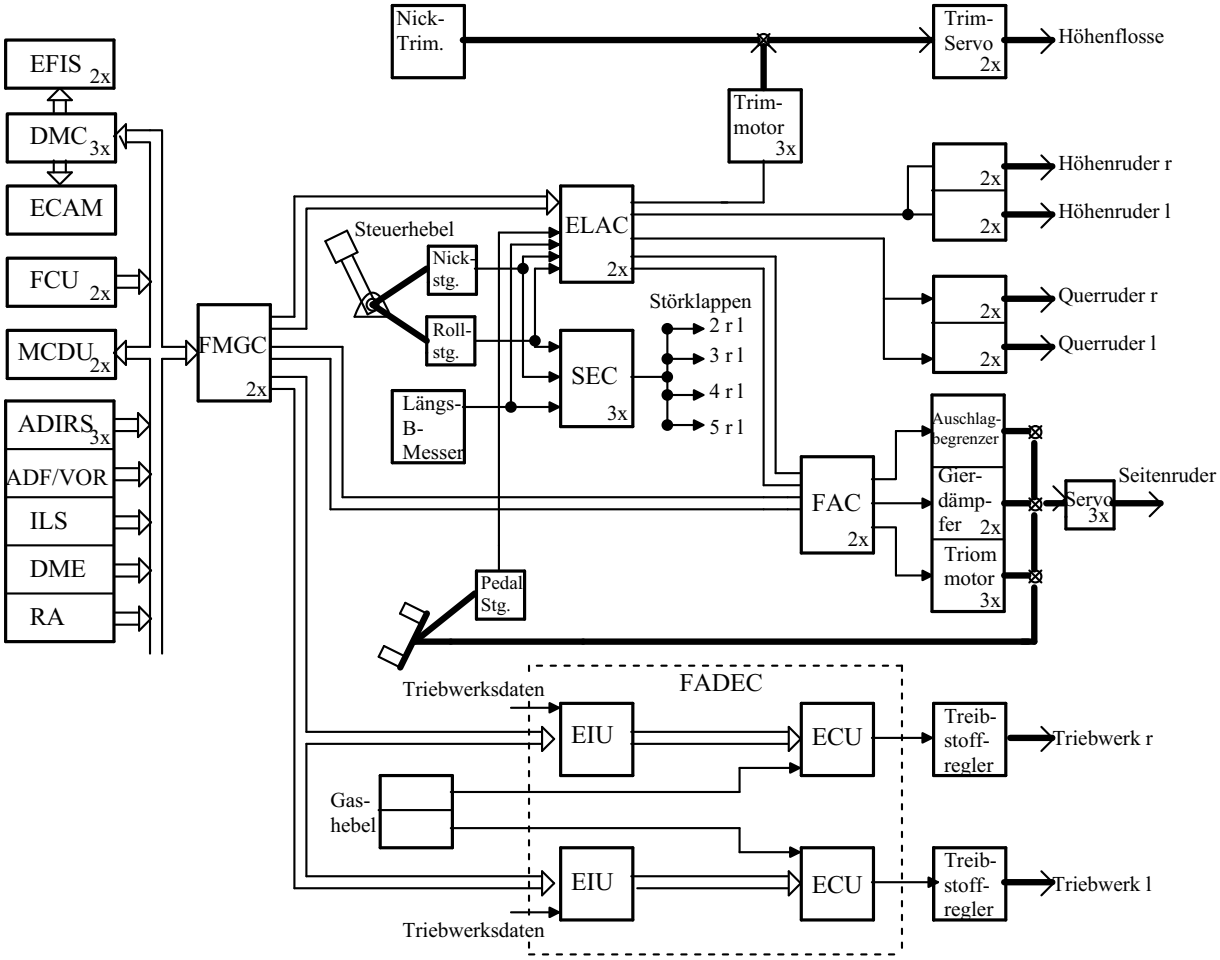


Abbildung 7.8: Airbus A-320, Flugsteuerungs- und Flugführungssystem.

die weder infolge von eingeleiteten Flugmanövern, noch durch atmosphärische Störungen überschritten werden dürfen. Wesentlich ist dabei der laufend eingeschaltete, aber nicht aktive, Vortriebsregler. Zusammen mit der elektrischen Flugsteuerung ermöglicht seine selbsttätige Aktivierung, Gefahrenzustände durch Überschreitung der Betriebsgrenzen zu verhindern. Der Triebwerksschub wird automatisch heraufgesetzt, wenn der Anstellwinkel sich positiven Grenzwerten nähert. Das Trimmsystem der Höhenruderflosse richtet das Flugzeug auf, wenn die maximale zulässige Betriebsgeschwindigkeit erreicht wird. Der Rechner sorgt dafür, dass die zulässigen Lastvielfachen während der Flugmanöver nicht überschritten werden.

7.4.2 Steuerung über die Pedale

Die Steuerung des Seitenruders erfolgt konventionell mit den Pedalen per Seilzug zu den hydraulischen Stellzylindern des Seitenruders [5][2]. Der Gierdämpfungsrechner FAC

erfüllt die Funktion der Gierdämpfung, Ruderausschlagbegrenzung und der elektrischen Trimmung. Eine vom Gierdämpfungsrechner FAC gesteuerte Dämpfungsfunktion wird über einen hydraulischen Dämpfungsstellzylinder in der mechanischen Anlenkung zum Seitenruder eingeführt. Die Information über die Gierdrehgeschwindigkeit erhält der Rechner vom Trägheitsreferenzsystem. Der Gierdämpfer ist während des gesamten Fluges in Betrieb, er führt zusätzlich die Funktion der Kurvenkoordination und der Kompensation des Seitenmomentes bei einem Triebwerksausfall durch. Zur Kurvenkoordination wird dem Rechner die Stellung der Pedale zugeleitet. Der Ruderausschlag wird in Abhängigkeit von der Geschwindigkeit begrenzt. Die elektrische Trimmung des Seitenruders wird über den Gierdämpferrechner vom Bediengerät gesteuert.

7.4.3 Elektrische Steuerung der Triebwerke

Das Steuergerät für die Triebwerke FADEC (Full Authority Digital Engine Control) besteht aus der Engine Contol Unit (ECU) und der Engine Interface Unit (EIU). Wie bei der Höhen- und Querruderbetätigung erfolgt die Übertragung der Gashebelstellung zu den Treibstoffventilen am Triebwerk über eine elektrische Signalübertragung. Die Gashebel dienen als Sollwerteingabe. Die mximal Werte sind mit Rasten versehen [5]:

O Leerlauf

CL Maximaler Steigflug

FLX flexibler Start = MCT maximaler ständiger Schub

TO Maximaler Start = GA Durchstarten R_{idl} Umkehrschub Leerlauf R_{max} Maximaler Umkehrschub

Der gewünschte Betriebszustand wird der ECU mitgeteilt. Vom Luftwerterechner bekommt die ECU über den Datenbus Informationen über den Luftdruck und der Außentemperatur. Die ECU errechnet die aktuellen Grenzwerte der Drehzahl N1 für das Triebwerk. Entsprechend der gewählten Betriebsart und der Grenzwerte für N1 wird das Treibstoffventil gestellt. Den Piloten wird der errechnete Grenzwert und die tatsächliche Drehzahl in Prozent mitgeteilt. Für die automatische Vortriebsregelung wird das Schubkommando vom Flugwegrechner FMG über den Datenbus dem Triebwerks-Anpassungsgerät EIU und von dort zur ECU geführt.

7.4.4 Sicherheit und Redundanz

Die Controller ELAC, FAC und FMGC wurden verdoppelt die Rechner SEC verdreifacht. Je zwei Rechner arbeiten parallel, damit beim Ausfall eines Rechners durch das Umschalten auf das Redundanzsystem kein Zeitverlust entsteht. Bei Ausfahll beider ELAC wird automatisch auf SEC umgeschaltet [2]. Der dritte SEC Rechner ist als Reserve-Rechner vorhanden (Abbildung 7.9).

Darüber hinaus ist der A-320 bei Totalausfall der Elektronik über die mechanische Trimmung des Höhenleitwerkes und der mechanischen Verbindung der Pedale zum Seitenruder eingeschränkt lenkbar [5]. Eine nicht einheitliche Ausführung der Controller soll die Gefahr von systematischen Fehlern verringern . Alle Steuergräte sind mit einer Eigenüber-

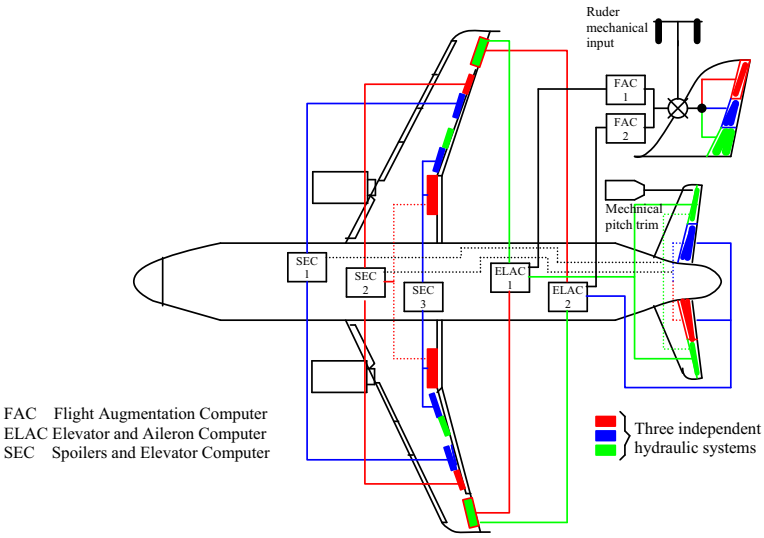


Abbildung 7.9: Redundanzschema der Fly-by-Wire Steuerung für das primäre Steuerungssystem.

wachung versehen. Das heißt sie sind intern durch zwei parallele Rechner realisiert, die sich gegenseitig monitoren (Abbildung 7.10). Sie melden ihren Status dem Centralized Fault Display System.

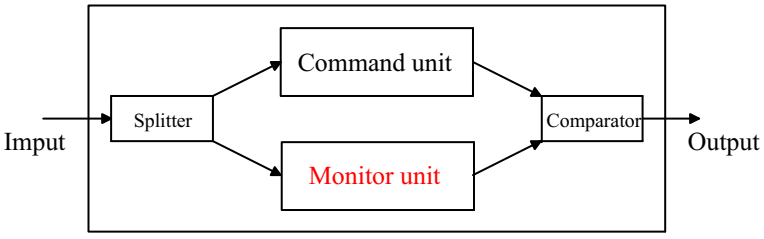


Abbildung 7.10: Eigenüberwachung

- ELAC: Motorola 68000
  - Command unit: Pascal
  - Monitor unit: C
- SEC: Intel 80186
  - Command unit: Assembler

– Monitor unit: Pascal

Die Rechner ECU und EIU für die Triebwerksreglung sind je einem Triebwerk zugeordnet.

7.4.5 Automatischer Flug

Der Flugwegrechner FMGC errechnet entsprechend der in der Flight Control Unit oder in der Multipurpose Control and Display Unit gewählten Betriebsart die Signale für die Betätigung der Ruder [5]. Die Autopilotensignale werden den Rechnern ELAC, SEC, FAC für die Rudersteuerung und den Triebwerksrechnern über den Datenbus mitgeteilt. Desweiteren wird das Electronic Flight Instrument System über den Display Management Computer mit den entsprechenden Informationen versehen.

7.4.6 Luftwerte- und Trägheitsnavigationsystem

Das wichtigste Messsystem des A-320 ist das Luftwerte- und Trägheitsnavigtionsystem ADIRS (Air Data and Inertial Refernce System). Es versorgt den Flugwegrechner und andere Systeme mit Luftdaten und Informationen über die Fluglage des Flugzeuges. Die Messgeräte für den statischen und dynamischen Luftdruck sind mit elektronischen Sensoren ausgestattet, die die Luftdruckwerte in digitale Signale wandeln. Dadurch kommt es zur einer erheblichen Reduzierung der Druckleitungen im Vergleich zu herkömmlichen Membranmessinstrumenten. Die Daten der Anstellwinkelmesser und der Außentemperatursensoren werden im Navigationsrechner des Trägheitssystem verarbeitet. Es werden die Luftwerte im Navigationsrechner umgerechnet und gemeinsam mit den Flugzeugkoordinaten und den Navigationswerten der Trägheitsplattform an alle Verbraucher an Bord zugeführt. Wegen der Bedeutung für die Flugsicherheit ist das ADIRS dreifach vorhanden.

7.5 Zwischenfälle mit dem Airbus A-320

7.5.1 Unfall von Warschau

Am 14. September 1993 gerät ein A-320 in Warschau über das Ende der Ladenbahn hinaus und kommt erst an einem Erdwall zum Stehen. Austretendes Kerosin entzündet sich und das Flugzeug steht in Flammen. Bei diesem Unfall kommen der Co-Pilot und ein Passagier ums Leben [1].

Standart Landeprozedur eines A-320

Sinkt das Flugzeug beim Landeanflug unter 50 Fuß, wechselt das EFCS in den Lande-Modus. Währen des Anfluges gibt es eine automatische Höhenansage für 100, 50, 40, 30, 20 und 10 Fuß. Beim Unterschreiten einer Flughöhe von unter 30 Fuß, verringert das EFCS innerhalb von 8 Sekunden den Neigungswinkel um zwei Grad. Der Pilot hat dabei seine Steuerknüppel zurückzuziehen. Die beiden Hauptfahrwerke bekommen zuerst Bodenkontakt. Befindet sich auf beiden Fahrwerken ein Gewicht von 12 Tonnen und mehr bekommt das EFCS das Signal Air Ground Transition (A/G). Sobald entweder die Drehzahl mindestens eines Rades einer Bodengeschwindigkeit des Flugzeugs von 72 Knoten entspricht oder das A/G-Signal anliegt und die Höhe unter 10 Fuß ist, werden die Ground Spoilers ausgefahren und somit der Auftrieb der Tragflächen zerstört. Gleichzeitig aktiviert das A/G-Signal, das mit einem Anti-Blockiersystem ausgestattete Radbremssystem und schaltet die Schubumkehr frei.

Beschreibung des Unfallverlaufs in Warschau

Die Fluglotsen informierten die Piloten über Seitenwindverhältnisse, jedoch nicht darüber, dass sich der Wind in der letzten Phase des Landeanfluges drehte. Eine Geschwindigkeit von 20 Knoten ist für Landeanflüge mit Seitenwind vorgeben. In Bodennähe herrschte jedoch ein 20 Knoten schneller Rückenwind. Somit setzte der Pilot den A-320 zu schnell und mit rechten Fahrwerk zuerst auf, da er den Airbus unter Annahme von Seitenwind nach rechts gekippt hatte.

Für das EFCS befand sich das Flugzeug bis zum Aufsetzen des linken Hauptfahrwerk noch in der Luft und verhinderte somit ein frühzeitiges Aktivieren der Schubumkehr sowie das Ausfahren der Bremsklappen. Zusätzlich ereichten die Räder durch das Aquaplaning erst sehr spät die erforderliche Drehzahl für die Aktivierung des Bremssystems. Wegen des Aquaplanings war das Anti-Blockiersystem aktiv und verlängerte zusätzlich den Bremsweg. Die Piloten schalteten daraufhin das ABS aus. Durch die Reibungshitze verband sich der Kohlenstoff des Reifenabriebs mit dem Sauerstoff des Regenwassers zu einem Schmierfilm, auf dem der Airbus dem Ende der Piste entgegenrutschte. Um am Ende der Rollbahn nicht mit der Nase voraus auf den Erdwall zu prallen, stellten die Piloten das Flugzeug mit dem Seitenruder quer.

Fazit

Dieser Unfall entstad aus einer Verkettung unglücklicher Zustände. Er zeigt jedoch deutlich, dass die Entscheidungsgewalt in einem eingebetten System wie dem Airbus A-320 nicht mehr eindeutig beim Menschen liegt sondern zwischen Mensch und Maschine aufgeteilt ist. Vielmehr noch werden Endscheidungen des Menschen verhindert, was zum einen notwendig ist, um menschlichem Versagen vorzubeugen. Auf der anderen Seite ist das System nicht immer in der Lage, die Umwelt bzw. die aktuelle Situation so zu begreifen,

wie sie wirklich ist. Ein Regler kann nur auf die Eingangsgrößen reagieren, die er durch seine Sensoren erfassen kann. Entscheidungen können nur im Rahmen vorab festgelegten Grenzen getroffen werden.

7.5.2 Übernahme der manuellen Steuerung bei Fly-by-Wire Flugzeugen mit Sidestick

Anfang März 2002 befand sich ein Airbus A-320 auf dem Flug von den Kanarischen Inseln nach Hamburg, als er im Raum Luxemburg plötzlich durch äußere Einflüsse aus seiner normalen Fluglage ausgelenkt wurde [3].

Das Flugzeug rollte erst nach rechts, bei ca. 10° Schräglage wurde der Autopilot abgeschaltet. Zunächst reagierte der Co-Pilot mit einem Gegenausschlag an seinem Sidestick. Nachdem das Flugzeug zur anderen Seite gerollt war, setzte die Gegenreaktion des Piloten ein. Beide Piloten lenkten ihren Sidestick mehrfach synchron im Rhythmus der Rollbewegungen aus. Beide Ausschläge addierten sich bis zum maximalen Vollausschlag eines Sidesticks.

Der Airbus rollte mehrere Male nach rechts und links und erreichte Rollraten von über $40°/sec$ und Querlagen bis 33°. Bei den beiden Flugzeugführern entstand der Eindruck, dass Flugzeug sei außer Kontrolle. Nach ca. 15 Sekunden hatte der Pilot das Gefühl, das Flugzeug wieder unter Kontrolle zu haben.

Die Untersuchung des Vorfalls ergab, dass der A-320 zur jeder Zeit steuerbar war. Da von keinem der beiden Piloten der Sidestick Priority Pushbutton betätigt wurde, addierten sich die Steuerbefehle der beiden Sidesticks, so dass das Flugzeug Manöver ausführte, deren Ursprung für beide Piloten nicht nachvollziehbar waren. Dadurch entstand der Eindruck, der Airbus sei außer Kontrolle.

7.6 Zusammenfassung

Unter der Voraussetzung, dass das Flugzeug wie am Beispiel des A-320 gezeigt, durch Maßnahmen wie mehrfacher Redundanz der Teilsysteme fehlertolerant und dadurch zuverlässig sind, überwiegen eindeutig die Vorteile, die FBW mit sich bringt. An erster Stelle steht dabei die Gewichtsreduzierung. Zum einen durch den Wegefall der mechanischen Verbindungen zu den Rudern, zum anderen können die Flugzeuge in ihrer Struktur viel leichter ausgelegt werden. Der A-320 z.B. verfügt über eine Lastabminderungsfunktion an seine Tragflächen [5][2]. Die Lastabminderungsfunktion dient dazu die Flügellast bei starker Turbulenz abzumindern. Hierzu werden die Querruder und die Außenspoiler in Abhängigkeit der auf die Flügel wirkenden Böenkräfte ausgeschlagen um eine Gegenkraft zu erzeugen, und somit die auf die Flügelstruktur resultierende Gesamtkraft zu minimieren. Dies erlaubt die Flügel schwächer auszulegen was wiederum zu einer erheblichen Gewichtsersparnis führt.

Ein weiterer Vorteil, der sich wiederum am Besten anhand der Airbus-Familie darstellen lässt, ist eine einheitliche Ausführung der Cockpits unterschiedlicher Flugzeugtypen. Das Flugführungssystem des A-320 hat sich derart bewährt, dass man es bei Airbus in ähnlicher Weise in den Typen A-330 und A-340 übernommen hat. Momentan wird es gerade an das Großflugzeug A-380 angepasst. Dieses für den Pilot ähnliche Handling der unterschiedlichen Flugzeuge macht es einem Piloten wesentlich einfacher von einem Flugzeugtypen auf einen anderen zu wechseln.

Neben der einheitlichen Gestaltung der Flugführungssysteme können dank FBW neue Entwicklung von Flugzeugen in fliegenden Simulatoren wie dem ATTAS (advanced technologies testing aircraft system) des DLR bereits vor der Realisierung getestet werden [5]. Bei der In-flight-Simulation fliegt der Pilot das Versuchsflugzeug durch ein aufgesetztes Rechnermodell des zu simulierenden Flugzeuges hindurch.

Nachdem sich FBW im Flugzeugbau bewährt hat finden die ersten Ansätze statt, diese Technologie auch in Hubschraubern zu übernehmen [4].

Literaturverzeichnis

[1] F. Bender. *Der Airbusabsturz LH2904 in Warschau*, 2003.

[2] R. Brockhaus. *Flugregelung*. Springer Verlag, 2 edition, 2001.

[3] Bundesstelle fuer Flugunfalluntersuchungen. *Übernahme der manuellen Steuerung bei Fly by Wire Flugzeugen mit Sidestick*. Hermann–Blenk–Str. 16 38108 Braunschweig, 1 edition, 2002.

[4] DLR. *Fly – by – Light Forschungshubschrauber EC135*. Braunschweig, 2003.

[5] K. Kracheel. *Flugführungssystemme – Blindfluginstrumente, Autopiloten, Flugsteuerungen*. Bernard Graefe Verlag, 1 edition, 1993.

[6] J. Rauch and M. Roessler. *FLY – BY — WIRE SYSTEMS FOR MILITARY HIGH PERFORMANCE AIRCRAFT aus Real – time system engineering and application*.

Kapitel 8

Software Entwicklung und Verifikation nach DO-178B

Autor: Yanling Song

Betreuer: Dipl.-Ing. Thomas Maier-Komor

8.1 Einleitung

8.1.1 Die Definition von DO-178B

In der Avioniksindustrie spielt Software eine große Rolle. Avioniks-Software werden entwickelt und verwendet, um die Fähigkeiten der mechanischen und analogen Systeme zu verlängern. Das ist in gewissem Sinne viel einfacher als die neue Hardware-einheiten zu entwerfen und zu ändern. Um die Sicherheit der Avioniks-Software zu garantieren und zu schätzen, braucht man die Richtlinien. Und DO-178B ist eine weltweite Richtlinie zur Entwicklung der Avioniks-Software[3], die in den späten achziger Jahren durch die Zusammenarbeit von den Avioniksindustrie und Qualität-und Zertifizierungsorganisation errichtet wurde. Dabei sind RTCA (Radio Technical Commission for Aeronautics) in USA und EUROCAE(European Organisation for Civil Aviation Equipment) in Europa die zwei wichtigsten Organisationen. Die gegenwärtige Version der Richtlinie ist weltweit seit 1992 verwendet worden. Sie ist eine der erfolgreichsten Software-Entwicklung Methoden zur Produktion der sicherheitskritischen eingebetteten Software.

Die Software in Luft- und Raumfahrt Systemen werden nach DO-178B entwickelt und zertifiziert. Damit ist die Sicherheit und Qualität der Software sichergestellt. Und DO-178B hat fünf Levels. Diese Levels beschreiben die Konsequenzen der potenzialen Fehler bei Software.[3]

Level A. Catastrophic; Die Fehler bei der Software verhindern den weiteren sicheren Flug und Landung.
Level B. Hazardous; Sie können mögliche tödliche Verletzungen einer kleinen Anzahl von Inhabern verursachen.
Level C. Major; Sie haben Einfluss auf die Leistungsfähigkeit der Crew, führen zur Unannehmlichkeiten oder möglichen Verletzungen
Level D. Minor; Sie können Sicherheitsleistung veringern.
Level E. No effect; Sie beeinflussen die Sicherheit des Flugzeuges gar nicht.

Level A ist der höchste und kritischste Level bei DO-178B, weil man die Konsequenz der Fehler, die zur Level A gehören, unbedingt vermeiden muss. Wenn die Software nach DO-178B Level A entwickelt und zertifiziert, hat diese Software höhere Sicherheit und gute Qualität.

8.1.2 Die Wichtigkeit der Verifikation und Validierung

Die Software in Avioniksystemen werden als Hilfsmittel für billige Änderung und Funktionsverlängerung eines ursprünglichen Systems gesehen. Aber die Software in Luft- und Raumfahrt System hat höhste Anforderung an Sicherkeit, Zuverlässigkeit, und Nachweisbarkeit. Z.B. Eine Avioniks-Software hat einen Entwicklungsfeher und es macht nicht das, was es soll, sondern etwas anderes, was vielleicht die tödlichen Verletzungen verursachen

können. Deswegen ist bei der Konstruktion von sicherheitskritischen Software eine Korrekte Entwicklung so wichtig, damit das System tut, was es tun soll und nichts anderes. Wie wurde es gewußt, ob ein System zu den Anforderungen errichtet wurde und ob die Anforderungen komplett und Korrekt waren? z.B. Systemanforderung, Softwareanforderung, Prüfanforderung. Wie stellen wir den Beweis zur Verfügung, dass die Fehler beim Software Entwurf und Sogar bei der Software Implementierung nicht zur sicherheitskritischen Situationen führen können? Um solche Probleme zu lösen, gibt es zwei wichtige Methode, nämlich die Verifikation und die Validierung. Die Verifikation stellt es sicher, dass das System nach den Anforderungen errichtet wurde, z.B. es bestätigt, dass der Source C Code die Softwareanforderung richtig implementiert hat. Und die Validierung liefert den Beweis, dass Systemanforderungen, Softwareanforderung, und Prüfanforderungen komplett und korrekt waren[7].

8.2 Software Entwickung nach DO-178B

8.2.1 Ein Überblick über die Software Entwicklung

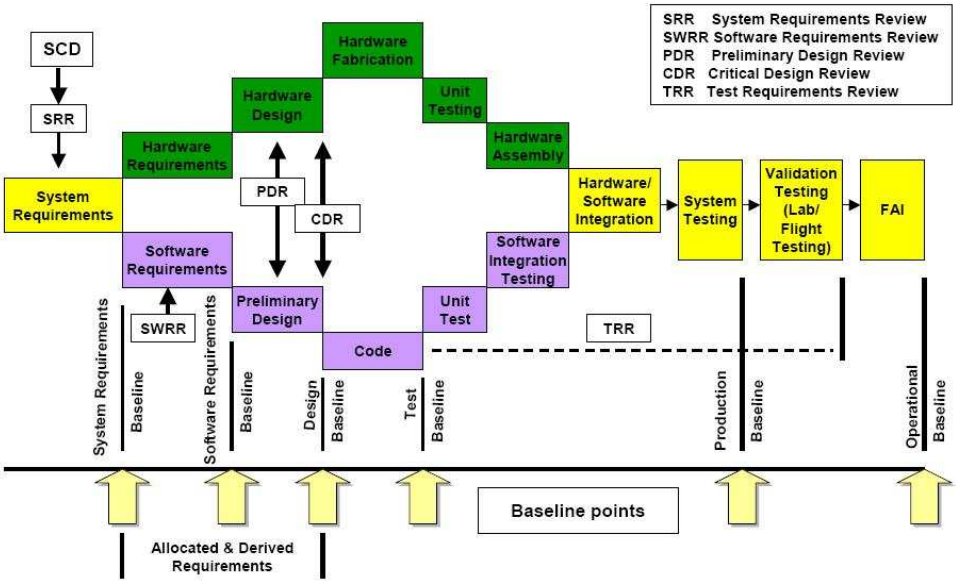


Abbildung 8.1: System Entwicklungsprozess[1]

Die Abbildung 2.1 beschreibt den ganz System Entwickungsprozess. Die Software Anforderungsspezifikationsgruppe analysieren Systemanforderung und bestimmen, welche Funktionen, Sicherheit und Schnittstelle von der Software übernommen werden sollen. Der Software Entwicklungsprozess enthält Softwareanforderung, Software-Entwurf, Quelltex, Modultest, Software-Software Intergration, und Hardware-Software Integration.

Die Abbildung 2.2 ist ein klassisches Wasserfallmodell, aber die Richtlinien bestätigen und erwarten, dass es die Interaktion und die Schleifen zwischen den Stufen gibt. Weil die

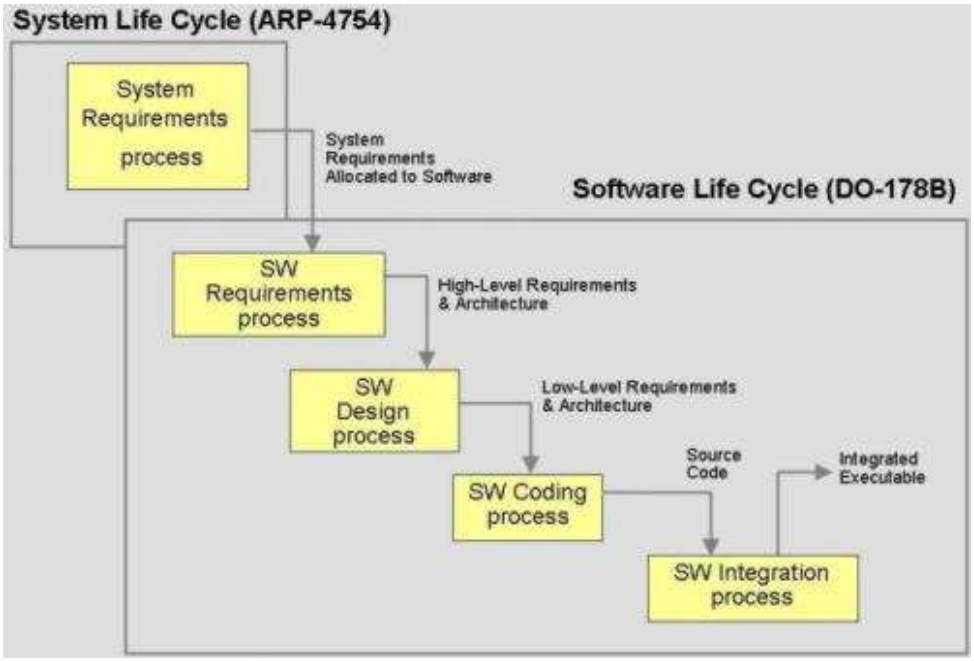


Abbildung 8.2: System Life Cycle[6]

Softwareanforderungen den Systemanforderungen entsprechen sollen, wird ein „Vertrag“ zwischen den Systemplaner und Software Entwickler gefordert. Software Anforderungsspezifikation enthält High Level Anforderung und Architektur. Beim Software Entwurf werden Low Level Anforderung und Architektur behandeln.

8.2.2 Software Anforderungsspezifikation

Software High Level Anforderung enthält Funktions-, Sicherheits-, Schnittstellen-, Fähigkeikts-, Datenelements-, Veränderungs-, Programierungsanforderung, Entwurfsbegrenzung, Anforderung an den Entwickler und algoritische Anforderung. Die Funktionsanforderungen beschreiben, was die Software machen soll, wofür die Software gebraucht wird oder wofür es gebaut wird. Die Sicherheisanforderungen beschreiben alle Gefahren und alles was niemals geschehen darf. Zuerst werden die Funktionsanforderungen im normalen Betrieb und die dabei entstehenden ersten Sicherheitsanforderungen betrachtet und getrennt festgehalten. Danach werden die Anforderungen im realitätsnäheren Fall, unter Berücksichtigung von Störungen, Ausfällen, Fehlern, Fehlbedingungen usw. untersucht und widerum getrennt Funktion/Sicherheit erfasst[4].

Durch das Aufteilen von Anforderungen wird eine bessere Verständlichkeit und damit eine höhere Sicherheit erreicht. Alle diese Anforderungen, die zur High Level Anforderung gehören, sollen getrennt nachweisbar und nachvollziehbar sein, damit kann man System Sicherheit gut bestimmen. Durch die Software Anforderungsspezifikation werden die folgende Ziele erreicht werden[2].

- Durch die Anforderungsspezifikation werden Mehrdeutigkeit, Widersprüche, und undifinierte Bedingungen beseitigt.
- Software High Level Anforderung sind nachweisbar, prüfbar. Und die Prüfungsmethode z.B. Review, Analyse, und Test sind erkannt.

8.2.3 Software Entwurf

Wenn „alle“ Anforderungen formuliert sind, wird die Software in Module aufgeteilt(Entwurf). Dabei werden die Anforderungen auf die Module verteilt. Die Schnittstellen und das Zusammenspiel der Module werden difiniert. Die Funktionsanforderungen können relativ einfach auf die Module verteilt werden, die Sicherheitsanforderungen aber nicht. Sie gelten nur für das gesamte System. In diesem Prozess müssen neue Sicherheisanforderungen für jedes Modul formuliert werden. Außer der üblichen Modularisierung für eine saubere Softwareentwicklung sollte auch eine sicherheitsorientierte Modularisierung stattfinden. Die Sicherheitsrelevante Funktionalität wird von der restlichen Funktionalität getrennt und in wenigen sicheren Modulen eingekapselt. Wo wenige Module vorhanden sind, ist es möglich, einen höheren Aufwand darin zu investieren, sie sicherer zu gestalten.[4]

Beim Software Entwurf gibt es zwei Hauptaufgabe, nämlich Software Architektur und Low Level Anforderung. Während des Software Entwurfs müssen folgende Punkte versichert werden[2].

- Input für Entwurfsprozess ist richtig. unrichtige und unausreichende Inputs werden vom Anforderungsprozess abgeklärt.
- Zwischen High Level Anforderung und Low Level Anforderung steht kein Kompromiss.
- Software Architektur und Low Level Anforderung sind nachweisbar. Und Nachprüfungsmethode, wie Review, Analyse, Test müssen auch nachvollzielbar sein.

Software Entwurf Dokument entspricht Software Entwurf Standard von DO-178B. Im Software Entwurf Dokument stehen folgende detaillierte Informationen zur Verfügung.

- Überblick über Computer Software Configuration Item und entsprechende Beschreibung der Architekture
- Modularisierung von Computer Software Configuration Item
- Funktionsbeschreibung jeder Computer Software Komponent
- Definition der Schnittstelle zwischen den Software Komponenten
- Ausführliche Entwurfsbeschreibung jeder Computer Software Komponente zum Computer Software Unit Level, z.B. die Liste der Parameter und der Variablen, die das CSU(Computer Software Unit) eintragen und herausnehmen

8.2.4 Die Implementierung der Softwaremodulen

Bei der Implementierung von Softwaremodulen kann man folgende Maßnahmen einsetzen, um Entwicklungsfehler zu vermeiden[4]:

- Modularisierung und Trennung von sicherheitsrelevanten Anteilen
- Vorsicht mit überlauf bei Variablen, die ständig inkrementiert werden
- Keine komplexen Optimierungen, einfach und direkt programmieren
- Aussagekräftige Kommentare und Referenzen zu den Anforderungen schreiben
- Anwendung von fachgebietsspezifischen CASE-Werkzeugen, nicht nur allgemeine CASE
- Ausnahmebehandlung von internen Fehlern
- Mögliche Fehlermeldungen oder Rückgabewerte niemals ignorieren
- Inputs müssen immer überprüft werden
- Anwendung einer sicheren Programmiersprache; z.B. Ada ist eine vorgezogene Programmiersprache
- Restriktionen bei der Speicherverwaltung beachten. Es ist empfehlenswert, alle Ressourcen und Speicherbereiche von Anfang an bei der Systeminitialisierung zu alloziiernund reservieren. Dies erspart das Risiko, dass die Ressourcen nicht vorhanden sind, wenn man sie braucht. Leider ist das nicht immer möglich; bei großen Systemen oder bei dynamischen Strukturen kann man nicht im voraus alles reservieren.

Durch Programierung werden Quelltext erzeugt. Dieser Quelltext ist nachvollziehbar und nachweisbar, und implementiert Software Low Level Anforderungen nach software Coding Standard.

8.3 Software Verifikation nach DO-178B

Die Abbildung 3.1 beschreibt den Software Entwicklungsprozess und Verifikaitonsprozess. Zwischen dem Entwicklungsprozess und Verifikationsprozess gibt es immer Interaktionen und Rückgespräche. Software Verifikation ist eine wichtige Methode um Software Qualität zu sichern. Review, Analyse und Test werden benutzt um eine Software zu verifizieren und validieren. Die Unabhängigkeit der Review und Analyse wird von DO-178B gefordert und dokumentiert. Review führt zur qualitativen Einschätzungen der Korrektheit. Analyse steht wiederholbare Nachweise der Korrektheit zur Verfügung.[1] Ausführliche Informationen über individuelle die methode und Abläufe von Review und Analyse werden in Software Verifikation Plan geschrieben. Duch test kann man zeigen, dass Software die High Level und Low Level Anforderungen erfüllt. die Fehler ,die zur unannehmbaren Ausfallzuständen führen könnten, rückgangiggemacht wurden. Die Test Betrachtungsweise und

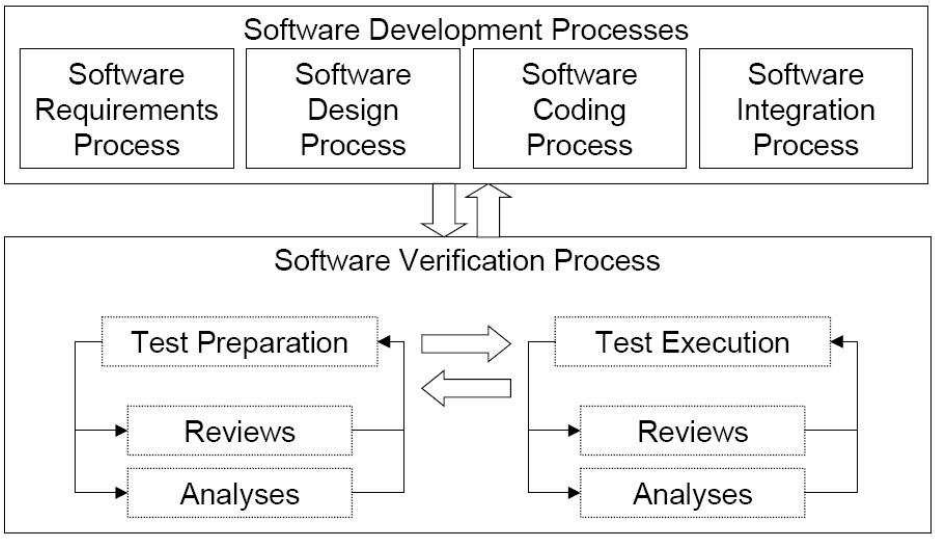


Abbildung 8.3: Software Verifikatonsprozess[1]

methode werden auch in Software Verifikation plan geschrieben. Die Test Entwurf,test Case und Test Ablauf werden in Software Validierungs Dokument geschrieben.

8.3.1 Review und Analyse

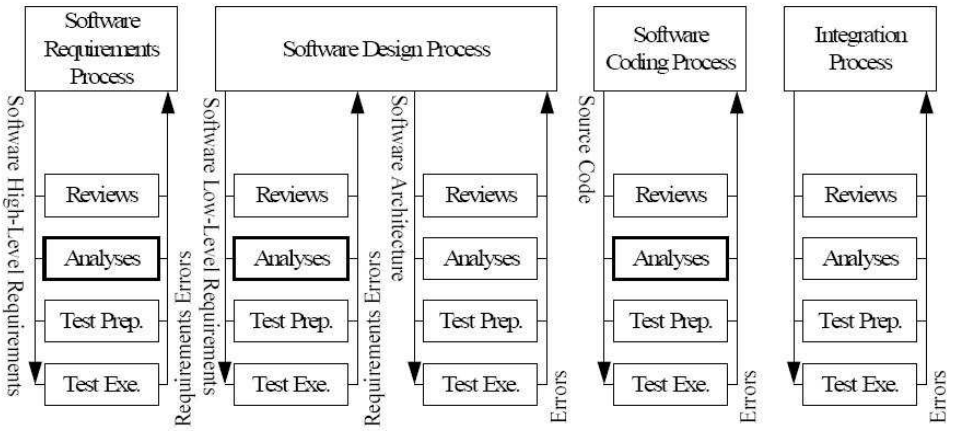


Abbildung 8.4: Review und Analyse beim Verifikationsprozess[1]

- Review und Analyse der High Level Anforderungen. Review und Analyse überprüfen die Befolgung der entsprechenden System Anforderungen, Genauigkeit und Übereinstimmung von Anforderungen, Kompatibilität mit Zielhardware, Nachweisbarkeit und Gültigkeit der gekennzeichneten Überprüfung Methode, Übereinstimmung zu den Standards und Traceability. Traceability bedeuted, man durch manche Methode

z.B. die Simulationen oder Trace Datei wissen kann, wie die Software abläuft, Was beim ablauf passiert, ob die Ergebnisse die Anforderungen erfüllen kann.[2]

- Review und Analyse des Quelltextes. Der Quetext wird auch nachgeprüft und analysiert. Sie überprüfen und bestätigen die Befolgung der Low Level Anforderung und Architektur, die Genauigkeit und die Übereinstimmung von den Anforderungen, die Übereinstimmung zum Standard und Traceability.[2]
- Review und Analyse der Integration Output. Die Entwickler überprüfen die Abwesenheit der unrichtige Hardware Addresse, Speicherüberlappen und Fehlenden software Komponent, runtergelandenen Daten und Speicherkarte.[2]

8.3.2 Test

Trotz all dieser Versuche, Fehler zu vermeiden, werden immer welche übrig bleiben. Hier können Tests nur das vorhandsein von Fehler beweisen. Es gibt zwei komplementäre Katagorien von Test: Blackbox und Whitebox und zwei Zusätze: Test-Suite und Random-tests.

Die Blackboxtest wird nur anhand der Spezifikation generiert. Er berücksichtigt nicht die Implementierungsdetails. Dabei wird die Funktion der gesamten Software getestet, ohne Implementierungskritische Aspekte genauer zu untersuchen. Ein Teil des Blackboxtests ist die Sensibilitätsanalyse der Output-Ergebnisse. Dabei wird getestet, wie empfindlich ein Output auf die verschiedenen Input-wert und internen Zustände des Programms reagiert[4].

Der Whiteboxtest berücksichtigt nur die Implementierung, ohne die gesamte Funktion zu untersuchen. Besonders wichtig sind die Grenzwerte[4].

Man kann das passende Testprogramm Schreibein und aufbewahren. Alle diese programme zusammen bilden eine Test- oder Validierungs-Suite. Diese Suite wird ständig erweitert. Bei jedem Durchlauf sollten die Ergebnisse aufbewahrt werden, um sie automatisch mit den Ergebnissen von nachfolgenden Tests vergleichen zu können. Eine manuelle Bewertung der neuen Testprogramme ist dann nur für die Ergebnisse der neuen Testprogramme erforderlich. Es ist wichtig, auch alte Dinge, die bereits getestet wurden, wieder zu testen, weil Änderungen und Erweitungen die Nebenwirkungen bei den bereits laufenden Teilen haben können[4].

Alle Computer Software Komponent werden geprüft. Die Test Methode wird ausführlich in Software Test Plan erklärt. Test case werden nach Low level Anforderungen geschrieben. Low Level Test überprüfen folgende Fehler:[2]

- Algorithmus Fehler zur Erfüllung der Software Anforderung.
- Falsche Schleifen Handlung
- Falsche Logik Handlung
- Arithmetische Fehler und Abweichung der Matrixhandlung.

- Falsche Berechnung-Reihenfolge und unzulängliche Algorithmuspräzision, Genauigkeit oder Leistung

Die Software-Software Integration Test werden normalerweise durch die Testcase in einem geeigneten Simulator durchgeführt, um folgendes zu überprüfen[2]:

- Falsche Initialisierung von Variablen und Konstanten
- Parameter passing, data corruption
- Unkorrekte Reihfolge der Events und Operationen.

Test Ergebiniss werden in Software Test Reports dokumentiert.

Hardware-Software Integration Test.[2]

- Falsche Unterbrechungshandlung.
- Nicht Erfüllung der Exekution-Zeit-Anforderung.
- Falsche Reaktion auf den Hardware Übergang, Fehler in den Hardware /software Schnittstelle.
- Fasche Control über Memory management
- Stack Overflow

Nach der Lieferung des Systems fängt die „echte Testphase“ unter realen Bedingungen an. Alles davor sind nur simulierte Bedigungen , die nur auf Annahmen basieren und deshalb falsch sein können. Viele Entwicklungsfehler kommen aus unbekannten oder ungenannten Situationen der realen Welt. Die Korrektur der in dieser Phase gefundenen Fehler erfordert einen enormen Aufwand. Nach jeder Korrektur(Änderung) sollte erneut der gesamte Test durchgeführt werden. Außerdem können die Korrekturen ein gefährliches Spiel werden. Z.B. bei den Avionikssystemen, Wo eine Änderung sehr umfangreich werden kann, kann man nicht ahnen, was für weitere Folgen die korrektur bringt.

Nicht nur bei jeder Änderung soll das System unter den aktuellen realen Bedingungen erneut getestet werden, sondern auch jedesmal, wenn man neue oder geänderte Bedingungen hat. Man darf nicht denken, dass ein getestetes System überall gleich arbeiten wird. Ein Beispiel:

Ein 100-fach erprobtes und bewährtes amerikanisches Luftraumkontroll System wurde in England installiert und verhielt sich extrem merkwürdig. Auf dem Monitor verschwanden Flugzeuge und tauchten woanders wieder auf. Das Problem war der Nullmeridian, der über England geht. In den USA brauchte man diesen Sonderfall nicht zu berücksichtigen.

8.3.3 Coverage-Analyse

Die Testergebnisse müssen natürlich ausgewertet werden. Man kann die Reaktion des System semiautomatisch gegen die Spezifikation vergleichen. Weiterhin sollte man eine

Test-Coverage-Analyse durchführen. Diese Analyse kann vollautomatisch mit Werkzeugen geschehen[4]. Beim Programmcode werd untersucht, ob alle Programmfäden durchgelaufen sind (sequentielle Blocks, Schleifen und if-then-else). Dies ist hilfreich, um die Qualität unserer Tests zwar nur grob, aber besser als gar nicht zu bewerten.

Durch Coverage-Analyse werde folgende Fehler entdeckt.[1]

- Tote(„nicht testbar“) Code
- Deaktivierter Code
- Unspezifizierte Funktionen
- Nicht verifizierte Funktionen

8.4 Die Zertifizierung der Avioniks-Software

Ziel der Zertifizerung ist, nachzuweisen, dass der Software Entwicklungsprozess und Verifikationsprozess die Anforderung von DO-178B erfüllen kann.

Es ist notwendig, die Regierng/Region Initiativen zu erwähnen, die innerhalb des Landes wichtig sind.

- FAA: Zertifizerungsorganisation für Avioniks für USA
- Transport Canada: Zertifizerungsorganisation für Avionics für Canada
- JAA (Joint Aviation Authority). Zur Zeit eine Koalition der unterschiedlichen europäischen lokalen Behörden - eine Autorität für die ganze Westeuropa in einigen Jahren vorgesehen.
- JahreCEAT (eine Niederlassung der französischen Armee) führt Software-Zertifizerung im Namen DGAC(die Administration für zivil Avioniks) durch

Eine Avioniks-Software wird innerhalb jedes Landes zertifiziert. Aber die Zertifizierungsorganisationen in unterschiedlichen Länder haben wechselseitige Vereinbarungen. z.B. der Zertifizierungsplan von JAA ist von FAA auf der Grundlage von dem Audit akzeptiert worden. CAST (Certification Authorities Software Team): Ist die Software spezifische Organisation ,wo alle diese Autoritäten auf dem Welt (FAA, Transport Canada and JAA) sich 3-4 mal pro Jahr treffen und über die Standards von unterschiedlichen Länder diskutieren.

Alle Software Entwicklungen- und Vefikationsdaten werden für Zertifizerung gefodert, Alle Software Komponenten müssen DO-178B erfüllen und nach System Anforderung geprüft werden. Es gibt keine allgemeine Akzeptanz. Neue Versionen müssen auch geprüft und zertifiziert. Die System Entwickler sollen der Zertifizierungsorganisation die folgende für Review und Akzeptanz abgeben.

- Software Development Plan: Definition der softwareleben und Entwicklungsenvironment.
- Software Verification Plan: Die Methode, durch die die Ziele des Software Verifikationsprozess erreicht werden können.
- Software Configuration Management Plan: Die Methode, durch die die Ziele der Software Configuration Management erreicht werden können.
- Software Quality Assurance Plan: Die Methode, durch die die Ziele der Software Qualitätsanforderung erreicht werden können.
- Software Verifikations Results: Results der Tests, Review und Analyse.

Literaturverzeichnis

[1] John Joseph Chilenski: *Software Development under DO-178B*
Boeing Commercial Airplanes, 01.2002.
http://www.opengroup.org/rtforum/jan2002/slides/safety-critical/chilenski.pdf

[2] *Software design and development as per DO-178B Requirements*
HCL Technologies Limited, Global IT Services Company.
http://www.hcltechnologies.com/pdf/DO-178B.pdf

[3] *Was ist DO-178 and Level*
http://www.aonix.com/objectada_DO-178B.html

[4] Sergio Montenegro: *Methoden und Techniken für die Entwicklung sicherheitsrelewanter Systeme*
Workschop „Ada Deutschland and Tagung 2001“, 27/28.März.2001, München
http://www.first.gmd.de/ sergio/public/methoden.html

[5] Siegfried Hörfarter: *Objektorientierte Konzepte in sicherheitskritischen Echtzeitanwendungen*
Berner&Mattner Systemtechnik GmbH, 18.07.2001, Ottobrunn.
http://www.bms.de/service_support/Download/OO_Konzepte_in_SK_Echtzeitanwendungen.pdf

[6] *SCADE Suite and KCG Code Generator. De Facto Standard in European Avionics Qualified for Safety-Critical Software Development*
http://www.esterel-technologies.com

[7] Andreas Mitschele-Thiel: *Systems Engineering with SDL*
Wiley, 2001.

Kapitel 9

Bussysteme in der Automobilindustrie

Autor: Michael Albrecht

Betreuer: Dipl.-Ing. Philipp Harms

9.1 Bussysteme in der Automobilindustrie

9.1.1 Geschichte der Elektronik in der Automobilindustrie

Die Geschichte der Elektronik im Automobil begann 1915 durch Fords Einführung des elektrischen Lichtes. Bis Ende der 80er Jahre bestand die Elektronik weitestgehend aus eigenständigen, unabhängig operierenden Steuereinheiten. Anfang der 90er Jahre wurde erstmalig der von Bosch entwickelte CAN-Bus im Antriebsbereich eingesetzt. Von da an nahm der Elektronikanteil im Automobil rasant zu. Im Jahr 2000 lag der Anteil bereits bei rund 20%, bis 2010 wird ein Anstieg auf 35% (Merker Management Consulting) prognostiziert. Das Elektroniksystem des aktuellen 7er BMS (E65) besteht beispielsweise aus fünf Bussystemen, die 45 bis 75 Steuergeräte miteinander vernetzen und 2500 Signale transportieren[1]. Damit ist der Grad an Komplexität erreicht, der neue Konzepte für die Systemintegration erfordert. Es wird erwartet, dass die Anzahl an Steuergeräten in den neuen Fahrzeuggenerationen künftig zugunsten einer zunahme an Funktionalität abnehmen wird.
Nach der Einführung des CAN-Busses im Antriebsstrang folgte in den 90er Jahren der Einstz von Bus-Technologien in den Bereichen Karosserieelektronik, Infotainment und nun auch in den passiven Sicherheitssystemen. Fortsetzen wird sich diese Entwicklung in den nächsten Jahren mit der Einführung der X-by-Wire-Technologie. Die unterschiedlichen Anwendungsgebiete im Automobil stellen ganz individuelle Anforderungen an das Kommunikationssystem, daher werden auch zukünftig immer mehrere Systeme parallel Verwendung finden.

9.1.2 Gründe für ein Bussystem

Statt einzelne Geräte über dedizierte Signalleitungen zu verbinden, gewinnen Bussysteme immer mehr an Bedeutung. Bussysteme haben hierbei folgende Vorteile:

- Jedes Gerät muss nur einmal an den Bus angeschlossen werden. Dadurch sinken Gewicht, Raumbedarf und Preis der Verkabelung. Es muss nicht jedes Gerät mit dem Sensor verbunden sein von dem es seine Informationen benötigt. In den Autos der Oberklasse werden ca. 3,5 km Kabel verlegt. Durch die Einführung eines Bussystems konnte das Gewicht beim neuen Audi A8 um 120 kg verringert werden.
- Ein an den Bus angeschlossenes Gerät kann mit allen anderen Geräten am Bus kommunizieren. Somit ist es möglich, dass ein Steuergerät einem anderen Steuergerät seine Meßergebnisse und Auswertung zu Verfügung stellt, bzw. auf Ergbenisse der anderen Steuergeräte zugreifen kann.
- Durch die Kommunikation zwischen den Steuergeräten ist eine Realisierung von sicherheitsrelevanten Eigenschaften mögich. Dadurch hat z.B. das Lenkungssteuermodul Möglichkeiten, sich Informationen über die Gewschwindigkeit zu holen, um somit den maximalen Lenkwinkel zu bestimmen.

- Durch Broad/Multicasts sind Diagnose-Verfahren, die den Bus überwachen, möglich. Anhand diese Methode kann man überprüfen, ob das Bussystem noch intakt ist oder ob Störungen in der Verbindung der Geräte vorliegen und somit auf ein Notfallsystem (z.B Ersatzbus) umgeschaltet werden muss.
- Über redundante Leitungen kann die Ausfallsicherheit eines Systems erhöht werden. Fällt ein Bussystem aus, kann auf eine zweite Verkabelung zurückgegriffen werden. Dies ist aber nur möglich, wenn man nicht 3,5 km Kabel doppelt absichern muss, sondern eben nur ein Bussystem, welches einen viel geringeren Kaberlverbrauch hat.
- Standards für den Bus, seine Schnittstellen und die Kommunikation fördern die Modularität der Geräte, so dass einerseits Geräte mit gleichartigen Funktionalitäten wiederverwendet werden können und andererseits der Variantenvielfalt vieler Automobile Rechnung getragen wird. Durch festgelegte Standards sind die Automobilhersteller nicht mehr auf einen Zulieferer angewiesen, sondern haben die Auswahl mehrerer Geräte, was wegen der fehlenden Monopolstellung zu einer Kostensenkung der Steuergeräte führt.

9.1.3 Eingesetzte Bussyteme in der Automobilindustrie

Bei der Verwendung von Bussystemen im Automobil stehen andere Anforderungen im Vordergrund als bei Computer-Netzwerken. Neben dem Preis sind vor allem sicherheitsrelevante Eigenschaften wichtig. In den folgenden Kapiteln wird daher nicht auf ATM, Ethernet oder Token-Ring eingegangen, sondern auf die drei wichtigsten Bussysteme die heute in Fahrzeugen der oberen Klassen stark verbreitetet sind.

- **CAN-Bus**(Controller-Area-Network)
- **LIN-Bus**(Local-Interconnect-Network)
- **MOST-Bus**(Media-Oriented-System-Transport)

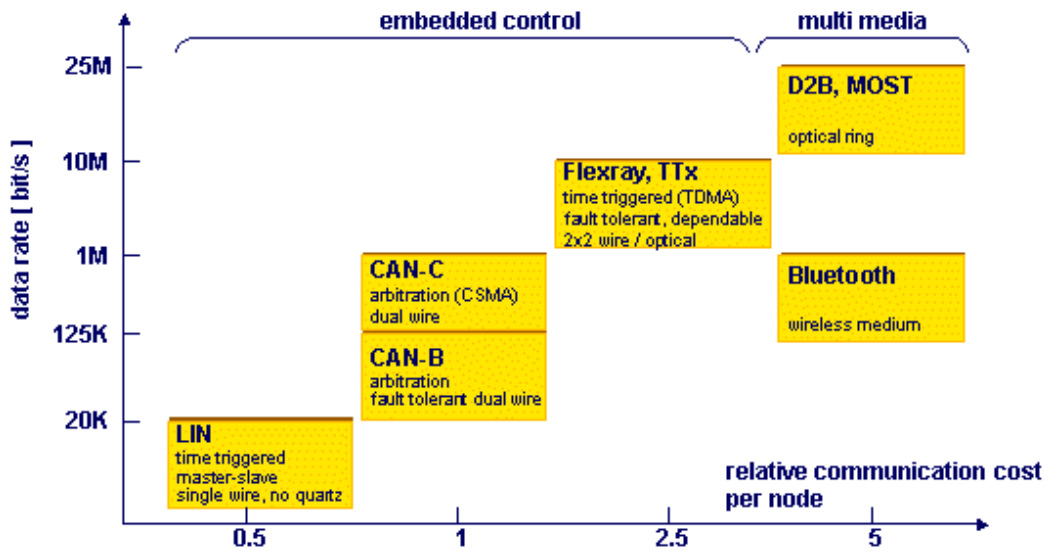


Abbildung 9.1: Bus-Übersicht
[2]

9.1.4 CAN-Bus

Das Controller Area Network - kurz CAN - wurde 1981 von Bosch entwickelt und findet seit Anfang der 90er Jahre verstärkten Einsatz in Kraftfahrzeugen. Durch die hohe Verbreitung von CAN sind die Komponenten sehr preisgünstig und in hohen Stückzahlen erhältlich.
CAN ist ein Standard der International Organization for Standardization (ISO 11898), der die untersten beiden Layer des OSI-Modelles, den Physical Layer (Bitübertragungsschicht) und den Data Link Layer (Datensicherungsschicht), spezifiziert.
Merkmale:

- CAN verfügt über eine Datengeschwindigkeit von 10 KBit/s bis 2 MBit/s. In der Automobilindustrie werden zwei verschiedene Geschwindikeiten eingesetzt. Für den Antriebs-CAN liegt die Datenrate bei 1 Mbit/s. Für die Ansteuerung des sogenannten Komfort-CAN wird ein CAN-Bus mit einer Geschwindigkeit von 500 KBit/s eingesetzt. Die Wahl der unterschiedlichen Geschwindkeiten wurde aus Kostengründen und aus Sicherheitsgründen angewandt. Je höher die Datenübertragungsgeschwindigkeit, desto teurer sind die einzelnen Sende-/Empfangseinheiten. In vielen Bereichen werden aber diese hohen Übertragungsraten gar nicht gebraucht und man kann einen langsameren und somit billigeren CAN-Bus verwenden.
- Als physikalisches Medium kommen Twisted-Pair Leitungen zum Einsatz, im Störfall kann auch auf eine Eindrahtleitung zurückgegriffen werden kann, wenn die entsprechenden Schaltvorrichtungen vorgesehen sind. Kupferkabel haben aber den Nachteil,

dass sie elektromagnetische Strahlung aussenden bzw. anfällig für die Einstrahlung solcher Störstrahlung ist. Der Vorteil solcher Leitungen sind die geringen Kosten, die einfache Verarbeitung und der unkomplizierte Einbau in ein Automobil.

- Das Zugriffsverfahren ist CSMA (Carrier-Sens-Multiple-Access). Dies bedeutet, dass alle Geräte auf einen gemeinsammen Bus zugreifen und untereinander aushandeln müssen, wer senden darf. Die Kommunikation erfolgt ereignisgesteuert. Will ein Rechner eine Nachricht versenden, so gibt er seinem zugehörigen CAN-Kontroller den Auftrag dazu. Dieser versendet die Nachricht, sobald der Bus frei ist. Tritt bei der Übertragung eine Kollision auf, sendet der CAN-Controller mit der höchsten Priorität (niedrigster Identifier) weiter und alle anderen CAN-Controller stoppen ihre Übertragung und warten bis der Bus wieder frei ist.
- Der Nachweis über die Realzeitbedingung ist sehr aufwendig. Da ein Knoten, wenn er einmal sendet, nicht beliebig lange senden darf, ist es möglich einen Realzeitnachweis für einen CAN-Bus durchzuführen. Dieser Nachweis muss aber immer wieder neu durchgeführt werden, sobald neue Geräte hinzukommen, auch ist die Vergabe der einzelnen Prioritäten für die Steuergeräte sehr aufwendig. Damit ist dieser Bus-Typ für harte Realzeitanwendungen in der Automobilindustrie ungeeignet.
- Anwendung in der Automobilindustrie:
  - Antrieb-CAN (1 MBit/s): Motorsteuerung, Getriebesteuerung
  - Komfort-CAN (500 KBit/s): Klimaanlage, Lampensteuerung

9.1.5 LIN-Bus

LIN - das Local Interconnect Network - ermöglicht eine kostengünstige Kommunikation für Sensoren und Aktoren dort, wo die Bandbreite und Vielseitigkeit von CAN nicht erforderlich ist.
Die Idee eines einheitlichen Kommunikationsstandards für mechatronische Systeme führt am Rande der VDI-Tagung „Elektronik im Kraftfahrzeug“ 1998 zur Gründung einer Arbeitsgruppe durch die Firmen DaimlerChrysler, BMW, Audi, Volkswagen, Volcano Communications Technologies und Motorola. Aus diesen Firmen setzt sich heute das Steering Kommitee zusammen. Im März 2000 wurde das LIN-System auf der SAE-Konferenz in Detroit einem breiten Fachpuplikum vorgestellt. Bereits 2001 erfolgte bei DaimlerChrysler der erste Serieneinsatz. Die Version 1.1 erschien im April 2000 und bestand aud den Teilen Protokoll, Bitübertragungsschicht, Configuration Language Description und dem Application Program Interface. Die Version 1.2, die im November 2000 erschien, enthielt relativ wenige Änderungen. In der aktuellen Version 1.3 wurde die Bitübertragungsschicht überarbeitet. Die Bitübertragungsschicht spezifiziert das Verhalten der Transceiver genauer und begrenzt die Drift der Versorgungsspannung und der Masse.
Merkmale:

- Der LIN-Bus verfügt über eine Übertragunsgeschwindigkeit von 19,6 KBit/s

- Der Physical Layer besteht aus einem 12V Eindrahtbus. Da hier die Kabel jedoch nicht verdrillt sind, ist dieser Bus noch anfälliger gegenüber elektromagnetischer Störstrahlung. Wegen dieser starken Anfälligkeit wird dieser Bus nur unterhalb von einem CAN-Steuergerät eingesetzt und auch nur in einer geringen örtlichen Ausdehnung (ca. 2 m) angewendet. Die genaue Definition der Bitübertragungsschicht mit - unter anderem - definierter Flankensteilheit sichert ein verbessertes EMV-Verhalten der Subsysteme.
- Es gibt einen einzigen Master und mehrere Slaves im Netzwerk. Die Kommunikation beim LIN-Bus läuft auch über Identifier ab. Im Gegensatz zum CAN-Bus können hier die einzelnen Steuergeräte nicht einfach senden wann sie wollen, sondern müssen auf eine Berechtigung vom Master warten. Im Master ist eine Scheduling-Tabelle hinterlegt, in der definiert ist, in welcher Reihenfolge und in welchem Zeitraster die einzelnen Nachrichten gesendet werden. Jede Kommunikation wird vom Master initiiert. Daher besteht eine Nachricht immer aus einem Header, der vom Master erzeugt wird, und einem Response des Slaves.
- LIN soll als „Billigbus“ unterhalb von CAN zur Vernetzung von Temperatur und Feuchtigkeitssensoren, Aktuatoren und Beleuchtungselementen eingesetzt werden, also überall dort, wo ein CAN-Kontroller aus Platz- und Kostengründen nicht sinnvoll ist. Er wird z.B in einer Tür für die Verriegelung, Fensterheber und Spielgelsteuerung eingesetzt.

9.1.6 MOST-Bus

Der Standart MOST (Media Oriented System Transport) wurde 1999 von der Audi AG und Harman/Becker festgelegt, als man sich Gedanken über das neue Multimediasystem für den A8-Nachfolger machte. Im Jahr 2000 kam es dann zu Bildung der MOST Cooperation, in der unter anderem zusätzlich BMW und DaimlerChrysler mitarbeiten. Das Anwendungsgebiet umfasst alle Multimedia-Anwendungen, also Audio, Video, Navigation und Telekommunikation (Internet). MOST ist ISO/OSI standardisiert. Der MOST-Bus zeichnet sich durch eine hohe Bandbreite bei gleichzeitig niedrigen Kosten aus. Die heutige Verwendung von Multimedia-Anwendungen zeigt Abbildung 9.2.

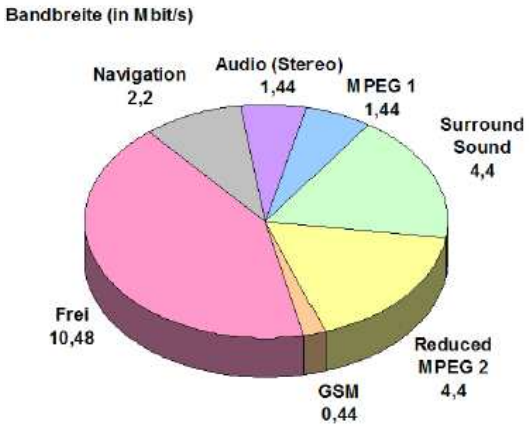


Abbildung 9.2: Multimedia-Anwendungen im MOST-Bus
[3]

Merkmale:

- Synchroner Datenstrom, d.h. alle am Bus angeschlossenen Geräte benötigen den gleichen Takt
- Hohe Datenraten bis zu 24,8 MBit/s in der ersten Spezifikation. In der zweiten Spezifikation sind es 150 MBit/s.
- Es wird die synchrone (bis zu 24 Mbit/s) und asynchrone (bis zu 14,4 Mbit/s) Übertragung von Nachrichten unterstützt, zusätzlich existiert ein asynchroner Steuerkanal mit bis zu 700 kBit/s.
- MOST ordnet jedem Knoten eine Adresse zu, die zur Kommunikation als Absender- oder Empfänger-Adresse verwendet wird.
- Als Physical Layer kommt ein optisches Medium (Plastic Optic Fibre - POF) zum Einsatz. Durch diese Art der Übertragung ist der MOST-Bus für elektomagnetische Störstahlung nicht anfällig.
- Es werden Ring-, Stern- und Ketten-Topologien unterstützt.
- MOST erlaubt Plug & Play mit bis zu 64 Knoten, die über ein Powermanagement verfügen.
- Echzeit-Audio-Fähig und Echtzeit-Video-Fähig.
- Datenübertragung im MOST-Bus:
  Ein spezieller Teilnehmer im Ring erzeugt den Takt, auf den sich alle anderen Teilnehmer synchonisieren. Dieser so genannte Timing Master erzeugt kontinierlich Daten, die in Frames eingeteilt sind. Frames kann man sich vorstellen wie Wagons eines Datenzuges (Abbildung 9.3), die der Timing Master erzeugt und zyklisch auf die Reise um den Ring schickt.

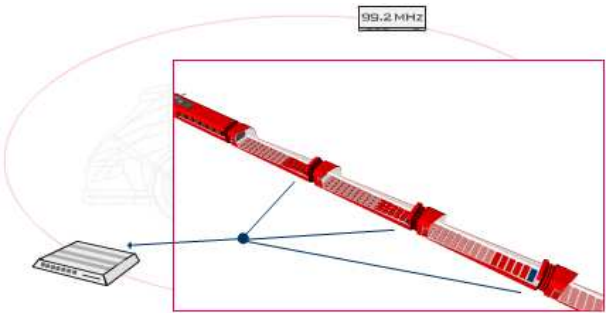


Abbildung 9.3: Datentransport über Most erfolgt in einem seriellen, synchronen Datenstom, der in Frames eingeteilt ist

[4]

Jeder MOST-Frame stellt eine Anzahl von Sitzen zur Verfügung, von denen jeder 1 Byte aufnehmen kann. Insgesamt enthält ein Frame 512 Bit bestehend aus:

- 0...480 Bit synchrone Daten. Die Zuteilung der Kanäle erfolgt über Time Division Multiplexing, die Bandbreite eines Kanales kann flexibel angefordert werden.
- 0...480 Bit asynchrone Daten. Zur Kollisionsvermeidung wird wie bei CAN die bitweise Arbitrierung verwendet, im Unterschied zu CAN werden aber Absender- und Empfänger-Adressen spezifiziert.
- 32 Bit Kontroll-Daten. Dazu zählen Preamble (für die Synchronisation), Boundary Descriptor (bestimmt die Länge des nachfolgenden Datenbereichs, Control Frame (für die Diagnose und Frameverwaltung) und zum Schluß noch Frame Control und Parity für die Fehlererkennung.

Ein Teil dieser „Sitze“ ist für synchrone Daten reserviert (maximal 480 Bit), ein weiterer für asynchrone (maximal 480 Bit) und der Rest für Kontrolldaten und administative Kommunikation (Abbildung 9.4).

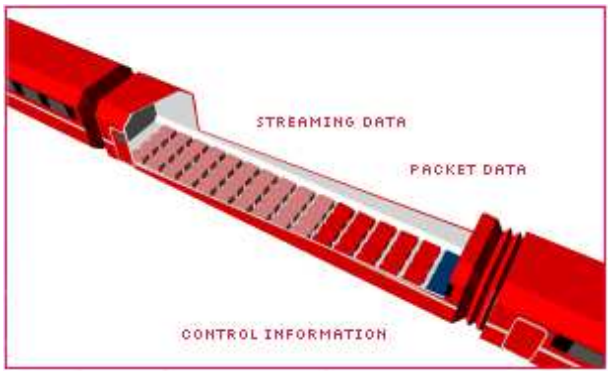


Abbildung 9.4: Übersicht über die Aufteilung eines MOST-Frame

[4]

In die synchronen „Sitze“ werden vor allem Audio- und Videosignale vom Sender (z.B. TV) zur Senke (z.B. Verstärker) transportiert. Da jeder Empfänger die Daten nicht aus den Sitzen entfernt, sonder lediglich herauskopiert, können mehrere Senken dasselbe Signal empfangen. Dadurch können z.B. in einem Fahrzeug mit Multimedia-Ausstattung Fahrer und Passagier dasselbe Bildsignal sehen. Im synchronen Bereich werden Daten ohne Adressen transportiert, woraus die extrem hohe Bandbreiten-Ökonomie von MOST resultiert. Sender und Empfänger der Daten einigen sich , in welchen Sitzen eines jeden Frames ihre Daten transportiert werden und reservieren sich diese. Sie dürfen von keiner anderen Datenquelle im System benutzt werden, um deren Daten zu transportieren.
Im Gegensatz dazu erfolgt im asynchonen Bereich und im Bereich der Kontolldaten ein Transport von adressierten Datenpaketen. Jedes Datenpaket, das in den dafür vorgesehenen Sitzen transportiert wird, ist mit einer Empfängeradresse versehen. Anhand dieser Adresse erkennt der Empfägner dass das Paket für ihn bestimmt ist. Da alle Geräte Zugriff auf alle Daten haben, entfallen teure Baugruppen wie Mischer oder Schalt-Matritzen, die in herkömmlichen analogen Systemen erforderlich sind. Komplett digitale Systeme sind dadurch kostengünstiger als analoge Lösungen und bieten dabei eine höhere Funktionalität und Flexibilität.

- Anwendung findet der MOST-Bus im der Multimediatechnik und im Infotainment Bereich.

9.1.7 Gateways

Die einzelnen Bussysteme LIN, CAN und MOST arbeiten mit unterschiedlichen Protokollen und Übertragungsmethoden. Für einen netzübergreifenden Datenaustausch sind daher Verfahren zur Protokollumsetzung notwendig. Dabei lassen sich zwei verschiedene Ansätze verfolgen:

Supergateway

Als Supergateway wird ein einzelnes, zentrales Gateway bezeichnet, das alle im Fahrzeug verwendeten Bussysteme miteinander verbindet. Dazu muss das Gateway neben den Hardwareanforderungen, wie z.B. unterschiedliche Steckerverbindungen, auch die steigende Komplexität der Software zur Protokollumsetzung und damit eine höhere Rechenleistung bewältigen. Probleme ergeben sich zudem bei der Fehlerausbreitung und wie diese in einem solchen System kontrolliert werden kann. Letztlich ist auch die Variantenvielfalt der Fahrzeuge - vom gering ausgestatteten Kleinwagen bis zum vollausgestatten Luxusmodell - nachteilig, da ein universelles Gateway im Hinblick auf Preis/Leistung, Funktionalität und Testbarkeit ungünstig ist.

Backbone-Architektur

Bei einer Backbone-Architektur übernehmen innerhalb der einzelnen Netzwerke lokale Gateways, die mit einem Backbone-Bus verbunden sind, die Protokollumsetzung. Dieser Backbone muss entsprechend den Anforderungen dimensioniert sein. Durch diesen dezentralen Ansatz wird eine modulare Vernetzung der Bussysteme ermöglicht, da ein Gateway hard- und softwareseitig nur mit dem Backbone verbunden sein muss. Die Funktionalität, die Kosten und die Skalierbarkeit und damit eine bessere Unterstützung der Variantenvielfalt sind die Vorteile einer solchen Architektur.

Einsatz in der Automobilindustrie

Trotz des Vorteils von einer Backbone-Architektur wird in der Automobilindustrie zur Zeit nur der Supergateway eingesetzt. Dies hat folgende Gründe

- Kosteneinsparung, da nur ein Gerät eingebaut und gewartet werden muss.
- Gewichtsersparnis, da nur ein Gerät und nicht mehrere Gateways eingesetzt werden.
- Die lokale Ausdehnung ist im Auto nicht so gross, und dadurch fällt der Vorteil der modularen Verkabelung weg.

9.1.8 Bussystem im Audi A8

Die Abbildung 9.5 zeigt das Bussystem vom neuen Audi A8. Da auch in diesem Modell ca. 75 Steuergeräte zum Einsatz kommen, würde die Grafik zu unübersichtlich werden und deshalb ist dies nur eine vereinfachte Darstellung der eingesetzten Steuergeräte.

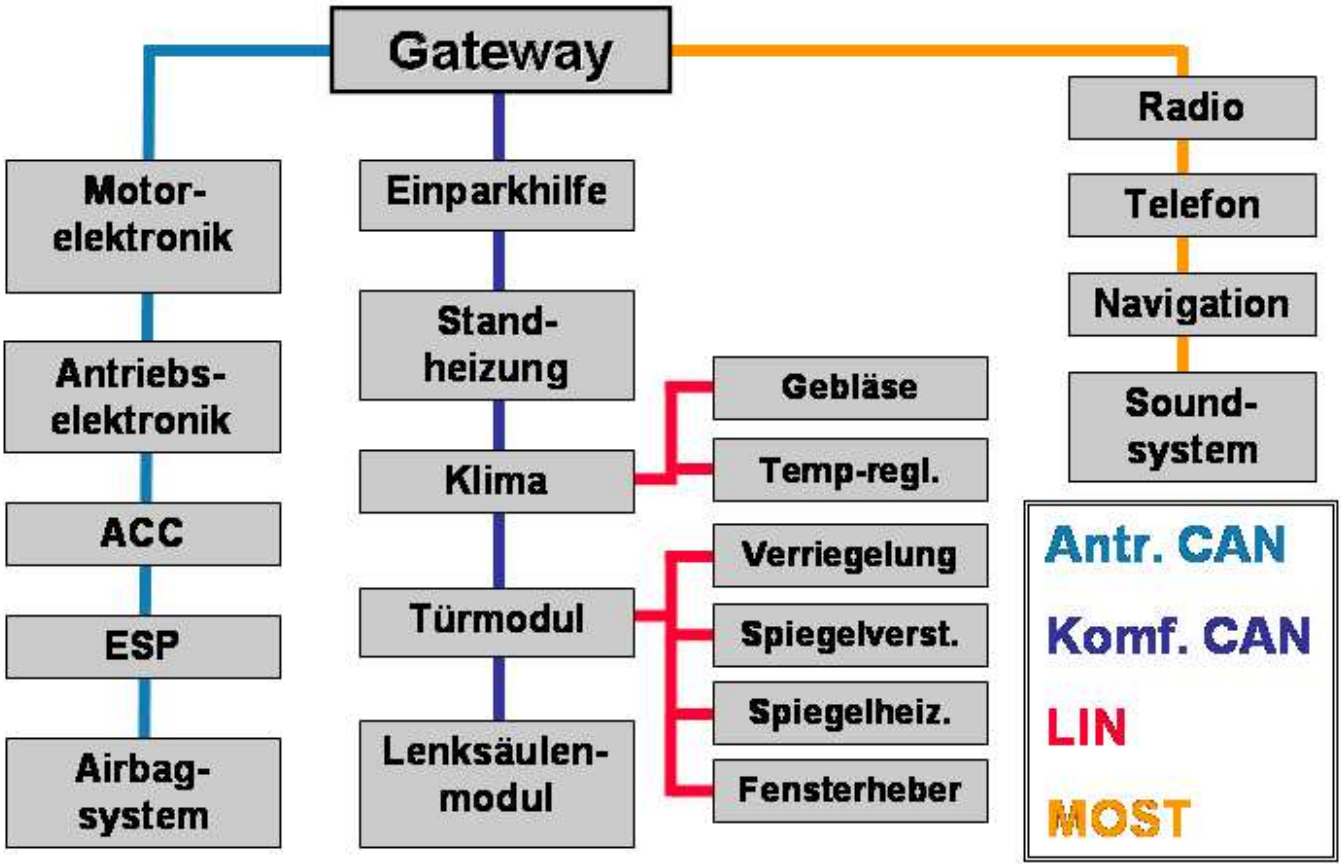


Abbildung 9.5: Übersicht über das Bussystem und die Steuergeräte im Audi A8 [5]

9.1.9 Zusammenfassung

Wie man sieht, befindet sich mehr als nur ein Bussystem im Einsatz in der Automobilindustrie. Jeder Bus hat seine Vor- und Nachteile und kann zur Zeit nicht durch einen anderen Bustyp ersetzt werden. Wie man in Abbildung 9.1 erkennen kann, sind die Kosten für einen einzelnen Netzwerk Bustyp abhängig. Die Grafik ist auf die Kosten für einen CAN-Knoten normiert. Ein LIN-Knoten, der zwar nur eine Übertragunsgeschwindigkeit von 19.6 KBit/s hat, kostet nur die Hälfte und ist somit für kurze Stecken mit niedrigen Datenraten der ideale Bus. Der MOST-Bus, der das fünffache von einem CAN-Bus Kostet hat eine sehr hohe Übertragungsgeschwindigkeit und auch eine hohe Störunempfindlichkeit gegenüber elektromagnetischen Störungen. Ein weiterer Nachteil des MOST-Busses ist, dass er zur Datenübertragung Glasfaserverbindungen einsetzt und diese einen relativ großen Biegeradius benötigen. Damit ist es nicht so leicht möglich, jede Stelle im Auto ohne Probleme zu erreichen. Deswegen wird der MOST-Bus den CAN-Bus nicht ablösen. Was aber keines von diesen drei Bussystemen ermöglicht ist der Einsatz in x-by-wire. Unter x-by-wire fällt zum Beispiel das steer-by-wire, dass die mechanische Verbindung zwischen Lenkrad und Räder durch einen Bus ersetzten soll. Für diese Art von Anwendung gibt es zur Zeit zwei Bustypen - TTP (Time-Trigger-Protokoll) und Flexray. Wobei sich Flexray in der Automobilindustrie durchsetzen wird.

Literaturverzeichnis

[1] Reichart, G.: Systemvernetzung oder Systemautonomie in der Kfz-Elektronik. Vortrag an der FH München am 25.03.03

[2] Lin-Subbus, http://www.lin-subbus.org/main.asp?cls=online&method=view&id=968

[3] Dohmke, Thomas: Bussysteme im Automobil CAN, FlexRay und MOST. Technische Universität Berlin Fakultät Elektrotechnik und Informatik Fachgebiet Softwaretechnik

[4] MOST-Cooperation, http://www.mostcooperation.com

[5] Elektronik Automotiv: Sonderausgabe Audi A8, www.elektroniknet.de

[6] Elektronik Automotiv Dezember 2002, www.elektroniknet.de

[7] Automobiltechnik 02.09.03

Kapitel 10

FlexRay

Autor: Lukas Brostek

Betreuer: Dipl.-Ing. Philipp Harms

10.1 Weshalb ein neues Bussystem?

Das Automobil der Zukunft wird durch ein Schlagwort bestimmt: Drive-by-Wire. Unter dieser Bezeichnung werden Systeme zusammengefasst, die herkömmliche mechanische Systeme in Fahrzeugen ersetzen sollen. Schon ansatzweise im Einsatz ist beispielsweise Brake-By-Wire als Ersatz des hydraulischen Brems-Kreislaufes durch eine elektronische Bremse, bei dem Bremsimpulse über Draht (Wire) ausgelöst werden. Noch Zukunftsmusik ist hingegen die elektronische Lenkung Steer-By-Wire, welche eines Tages die Lenkstange und das Lenkgetriebe ersetzen soll.

Drive-by-Wire bietet eine Menge von Vorteilen gegenüber mechanischer und hydraulischer Verbindungen. Beispielsweise wird es so möglich, ein Fahrzeug mit Joysticks anstatt Lenkrad und Pedalen zu lenken. Sicherheitssysteme wie ESP können relativ einfach Regelungseingriffe auch über die Lenkeinstellung der Rader vornehmen, was bei herkömmlicher Bauweise sehr Kompliziert wäre. Das Fahrzeuggewicht könnte beträchtlich gesenkt werden, Wartungsarbeiten wie das Wechseln von Hydraulikflüssigkeiten würden entfallen. Schließlich könnten durch Drive-by-Wire ganz neue Funktionen verfügbar gemacht werden: das virtuelle Verbinden von LKWs, das automatische Einparken oder gar das ganz automatische Fahren.

Aufgrund der Vielzahl an Komponenten in einem Automobil wird man die für solche Anwendungen notwendigen Steuergeräte wohl kaum über dedizierte Signalleitungen verbinden. Um eine unüberschaubare Vernetzung zu vermeiden, wird man zu Bussystemen greifen müssen. Allerdings eignen sich herkömmliche Fahrzeug-Bussysteme wie CAN, LIN oder MOST hierfür nicht besonders.

Zunächst einmal benötigt Drive-by-Wire eine relativ hohe Bitrate, man geht heute von

10 Mbit/s aus. Außerdem muß das Bussystem realzeitfähig sein, das heißt, die Sendezeit einzelner Nachrichten darf unter keinen Umständen einen bekannten Wert übersteigen. Beispielsweise müssen im Fall eines Unfalls die Zündsignale für die Airbags in einer bestimmten Zeit bei diesen ankommen, auch wenn gleichzeitig alle möglichen Crash-Sensoren ihre Daten versenden wollen. Aber der wohl wichtigste Aspekt bei Drive-by-Wire Anwendungen ist eine höchstmögliche Fehlertoleranz. Man stelle sich nur einmal vor, gerade mit 200 km/h auf der Autobahn zu fahren und plötzlich funktioniert die Lenkung aufgrund irgendeines Fehlers im Bussystem nicht mehr. Schließlich wird von der Automobilindustrie hohe Flexibilität verlangt. Es soll von einem Bussystem eine möglichst billige Variante für Kleinwagen existieren, ebenso wie eine für die Luxusklasse mit erheblich mehr Steuergeräten.

Aus diesen Gründen haben Industrie und Forschung in den letzten Jahren nach einem System gesucht, welches all diesen Anforderungen entspricht. Wie so oft kam man auch hierbei auf mehrere Lösungen:

Das eine Lager sieht im TTP-Bus der österreichischen Firma TTTech die Basis für zukünftige Drive-by-Wire Anwendungen. Seit nun etwa schon 20 Jahren wird an diesem geforscht. Vor einigen Jahren erstellten auch einige Automobilkonzerne ihre ersten Drive-by-Wire Versuchsfahrzeuge auf Basis dieses Busses. Heute wird er aber fast ausschließlich nur noch von der Luftfahrtindustrie unterstützt. Dadurch, daß hinter TTP eine Firma steht, müssen für dessen Benutzung nämlich Lizenzkosten gezahlt werden.

Dies ist wahrscheinlich der Hauptgrund, warum 1999 DaimlerChrysler und BMW das FlexRay-Konsortium gründeten. FlexRay ist ein Bussystem, welches technisch dem TTP-Bus sehr ähnlich ist. Firmen, die dem FlexRay-Konsortium beitreten, dürfen diesen aber lizenzkostenfrei einsetzen. Heute sind neben beiden Gründungskonzernen praktisch alle großen europäischen und amerikanischen Automobilunternehmen, d.h. Volkswagen, General Motors und Ford Mitglieder dieses Konsortiums. Bei der Entwicklung spielen des weiteren Bosch, Philips und Motorola eine wichtige Rolle. Im folgenden werde ich die technischen Eigenschaften von FlexRay beschreiben.

10.2 FlexRay-Controller

Der FlexRay-Controller übernimmt die Kommunikationsaufgaben für den Host-Controller, welcher jedes beliebige Steuergerät im Fahrzeug sein kann, und besteht aus 3 Hauptkomponenten [1]:

Der Kommunikations-Controller ist der Hauptbestandteil eines FlexRay-Controllers und regelt die gesamte Kommunikation.

Der Transceiver ist an den Bus angeschlossen und generiert und empfängt das übertragungssignal. Dieses kann sowohl elektrisch als auch optisch sein. Der Transceiver beinhaltet einige Diagnosefunktionen. Beispielsweise kann er Unterspannung, übertemperatur und Busleitungsfehler erkennen und dies an den Host-Controller melden.

Der Bus Guardian hat eine sicherheitstechnische Aufgabe: Sollte ein Bus-Knoten versuchen, außerhalb der ihm im Kommunikationszyklus zugewiesenen Zeit zu senden, entzieht der Bus Guardian dem Transceiver die Sendefreigabe. Das heißt, der Knoten kann nicht zu einem so genannten "Bubbling Idiot" werden und die gesamte Kommunikation stören. Hierfür benötigt der Bus Guardian eine eigene Zeitbasis und somit eine eigene Uhr.

Um Strom zu sparen besitzt der FlexRay-Controller ein Power Management mit den drei Betriebsmodi "Normal", "Stand-by" und "Sleep". Während des normalen Betriebs wird der Physical Layer mit 5V versorgt. Wenn die Kommunikation über den Physical Layer nicht benötigt wird, der Host im Knoten aber selbst aktiv bleiben soll, kann der FlexRay-Controller in einen stromsparenden Stand-by-Modus geschickt werden. Wird die Funktion des gesamten Knotens nicht mehr benötigt, kann er in den Sleep-Modus gehen, die Stromversorgung wird dann gänzlich abgeschaltet.

Zur Einführung von Redundanz kann ein zweiter physikalischer Kanal an den FlexRay-Controller angeschlossen werden. Sollte eine Leitung beispielsweise durch einen Kurzschluß ausfallen, kann die gesamte Kommunikation über den zweiten Kanal fortgesetzt werden. Hierfür müssen im FlexRay-Controller natürlich ein zweiter Transceiver und Bus Guardian vorhanden sein. Alternativ kann der zweite Kanal aber auch zur Erhöhung der Datenrate eingesetzt werden.

10.3 Physical Layer (Schicht 1)

Die Brutto-übertragungsrate bei FlexRay beträgt 10 Mbit/s für jeden physikalischen Kanal. Sollte ein optionaler zweiter Kanal nicht zur Gewinnung von Redundanz verwendet werden, verdoppelt sich die übertragungsrate. Die Kommunikation kann wie gesagt sowohl elektrisch, als auch optisch erfolgen. Die erste Variante hat den Vorteil, daß durch das Auswerten von Spannungspegeln im Transceiver Aussagen über den Leitungszustand gemacht werden können. Optische übertragung hat hingegen den großen Vorteil, daß die gesamte EMV-Problematik entfällt. Es können also beispielsweise keine Störspannungen in die Leitungen induziert werden. Ebenso senden die Leitungen keine Störungen aus.

Da FlexRay unter dem Gesichtspunkt großer Flexibilität entwickelt wurde, unterstützt es unterschiedlichste Bustopologien: Zum einen kann die Verschaltung in Form eines einfachen passiven Busses geschehen. Man kann aber auch in Sternform verbinden, wobei im Zentrum ein sogenannter Aktiver Stern-Knoten stehen muß. Schließlich können auch

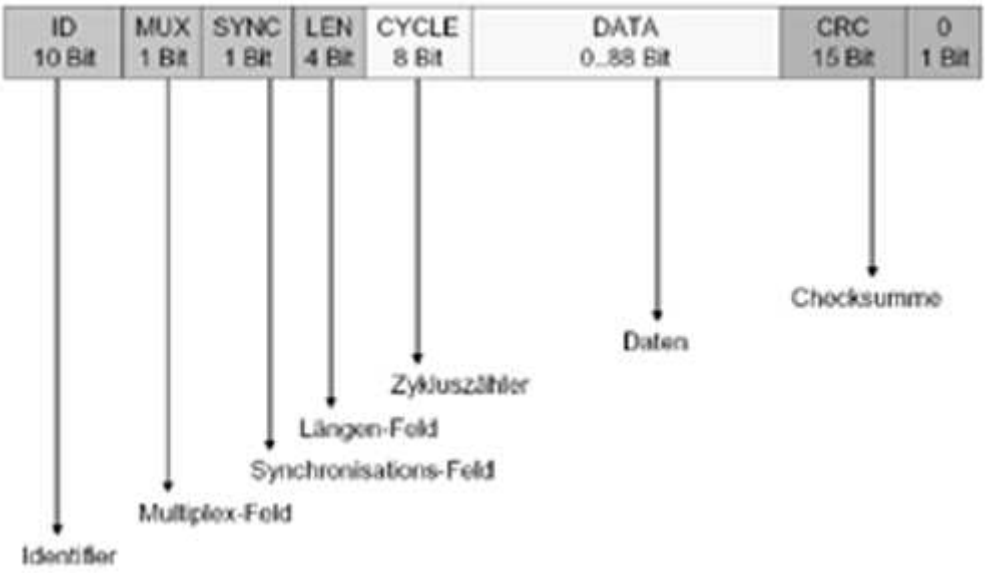


Abbildung 10.1: Das bei FlexRay eingesetzte Nachrichtenformat

hybride Bus-Sterntopologien gewählt werden.

Soll die übertragungsrate mehr als 1 Mbit/s betragen, müssen Aktive Sterne eingesetzt werden, da nur diese Topologie Punkt-zu-Punkt Verbindungen liefert. Hierdurch werden ideale Bus-Abschlüsse erzielt, was erst eine hohe Datenrate zulässt. Ausserdem werden die einzelnen Zweige in diesem Fall elektrisch voneinander getrennt, da Daten durch den aktiven Stern an alle angeschlossenen Zweige übertragen werden. Dies hat den großen Vorteil, daß Zweige im Fehlerfall, beispielsweise einem Kurzschluß, durch den aktiven Stern abgeschaltet werden können und nicht die übrige Kommunikation beeinflußen.

10.4 Kommunikationsprotokoll

Abbildung 10.1 zeigt das bei FlexRay verwendete Nachrichtenformat [2]. Das ID-Feld beinhaltet die Identity einer Nachricht, welche durch den sendenden Knoten bestimmt wird. Hierdurch wird auch die Position im statischen Teil und die Priorität im dynamischen Teil des Kommunikationszyklus, auf den später noch eingegangen wird, bestimmt. Durch setzten des MUX-Bits können Nachrichten mit der selben ID unterschieden werden. Das SYNC-Bit wird bei den Nachrichten für die Uhrensynchronisation gesetzt. Das LEN-Feld gibt die Länge des Datenfelds (DATA) an. Im CYCLE-Feld befindet sich ein Zykluszähler, der in jedem Kommunikationszyklus von Neuem mit jeder Nachricht hochgezählt wird. Schließlich wird jede Nachricht mit einem Cyclic Redundancy Check gesichert. Hierbei wird ein Code mit einer Hamming-Distanz von sechs Bit verwendet, das heißt zwei Bit-Fehler können korrigiert und fünf noch erkannt werden.

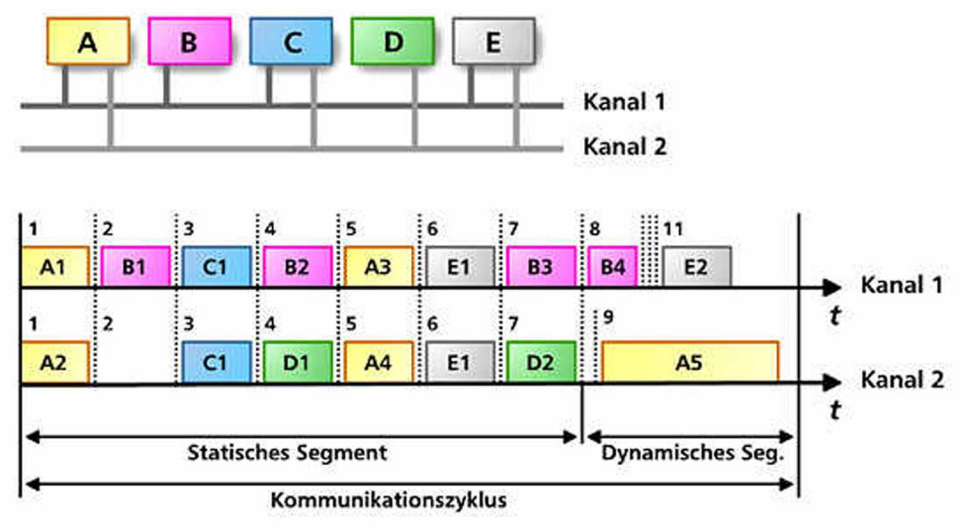


Abbildung 10.2: Kommunikationszyklus, bestehend aus einem statischen und einem dynamischen Teil

Wann ein Knoten nun eine solche Nachricht versenden darf, legt der Kommunikationszyklus [2] fest. Wie in Abbildung 10.2 gesehen werden kann, besteht jeder Kommunikationszyklus bei FlexRay aus zwei Teilen: einem zeitgesteuerten und einem ereignisgesteuerten.

Der zeitgesteuerte (engl.: time triggered) Teil ist statisch und funktioniert nach dem Prinzip des Time Division Multiple Access (TDMA). Dies heißt, daß jedem Bus-Knoten ein fester Zeitschlitz (static slot) zugewiesen ist, in welchem er seine Nachrichten senden muß. Hierdurch wird eine vordefinierte Auslastung dieses Teils des Zyklus erreicht und somit eine überlastung des Kommunikationssystems prinzipiell ausgeschlossen. In diesem Teil werden harte Realzeitbedingungen erfüllt, da die Sendezeitpunkte aller Nachrichten von vornherein feststehen und Kollisionen unter keinen Umständen auftreten können. Es kann relativ einfach die maximale Zeitdauer berechnet werden, die vom Zeitpunkt des Sendewunsches eines Knoten bis zum Eintreffen der entsprechenden Nachricht bei einem Ziel-Knoten vergehen kann.

Der zweite Teil des Kommunikationszyklus, also der ereignisgesteuerte (engl.: event triggered) oder auch dynamische, funktioniert nach dem Prinzip des sogenannten Flexible Time Division Multiple Access (FTDMA). Hierbei gibt es keine festen Zeitschlitze für die einzelnen Knoten, das heißt jeder Knoten darf beliebig lange (innerhalb bestimmter Grenzen) Nachrichten senden. Selbstverständlich darf ein Knoten hierbei nur anfangen zu senden, falls nicht gerade ein anderer Knoten sendet. Dies entspricht so weit dem bekannten Carrier Sense Multiple Access (CSMA), wie man es beispielsweise vom CAN-Bus kennt. Beim FTDMA wird aber zusätzlich das Auftreten von Kollisionen verhindert. Hierzu wird der dynamische Teil des Kommunikationszyklus in sogenannte Minislots ein-

geteilt, wobei jedem Minislot eine Nachrichten-ID zugewiesen wird. Nun darf jeder Knoten erst zum Zeitpunkt des Minislots mit der entsprechenden ID beginnen eine Nachricht zu senden. Somit können nicht mehrere Knoten quasi gleichzeitig mit dem Senden einer Nachricht beginnen. Da die Minislots mit niedriger ID vor denen mit höherer kommen, kann durch dieses Verfahren auch noch eine Priorisierung der Nachrichten in diesem Teil vorgenommen werden. Zum Beispiel wird eine Nachricht mit der niedrigsten ID in jedem Kommunikationszyklus gesendet werden dürfen, da der Bus in diesem Fall durch keine andere Nachricht belegt sein kann.

Somit vereint FlexRay die Vorteile synchroner Zugriffsverfahren, wie zum Beispiel das Erfüllen harter Realzeitbedingungen, mit den Vorteilen, die ein asynchrones Zugriffsverfahren bietet. Hier sei beispielsweise die Flexibilität in Hinsicht auf die Datenmenge, die jeder Knoten in einer Kommunikationsrunde schicken darf, zu nennen. Die Grenze zwischen statischem und dynamischen Teil des Kommunikationszyklus kann zudem in der Entwicklungsphase variabel gesetzt werden, je nachdem, ob eher viele zeitgesteuerte oder ereignisgesteuerte Nachrichten erwartet bzw. benötigt werden. Im laufenden Betrieb allerdings muß diese Grenze fest bleiben.

Ein solcher Zyklus fällt und steht mit der Uhrensynchronisation. Daher möchte ich mich als nächstes hiermit beschäftigen.

10.5 Uhrensynchronisation

Da bei FlexRay in der Entwicklungsphase ein fester Kommunikationszyklus festgelegt wird an den sich jeder Knoten zu halten hat, muß jeder FlexRay-Controller über seine eigene Uhr verfügen. Hierbei reicht es aber nicht aus, jedem Controller einen Quarz einzubauen und darauf zu vertrauen, daß dieser über eine ausreichend hohe Genauigkeit verfügt. Zum einen sind die Fertigungstoleranzen zu hoch, zum andern arbeiten die Controller insbesondere in einem Automobil unter äußerst unterschiedlichen Umgebungstemperaturen. Daher müssen die Uhren aller FlexRay-Controller ständig synchronisiert werden. Weshalb dabei außerdem eine möglichst exakte Synchronisation erreicht werden sollte, verdeutlicht Tabelle 10.1.

Da praktisch niemals eine hundertprozentige Synchronisation erreicht werden kann, wird zwischen den einzelnen Nachrichten eine Lücke eingefügt, der sogenannte Interframe Gap.

Interframe Gap (in μs)	30	30	30	4	4	4
Nutzdaten pro Nachricht	16	24	246	16	24	246
Nutzbare Bandbreite (in %)	29,6	38,7	86,6	57,1	66,7	95,3

Tabelle 10.1: Die nutzbare Bandbreite steigt mit kleiner werdendem Interframe Gap

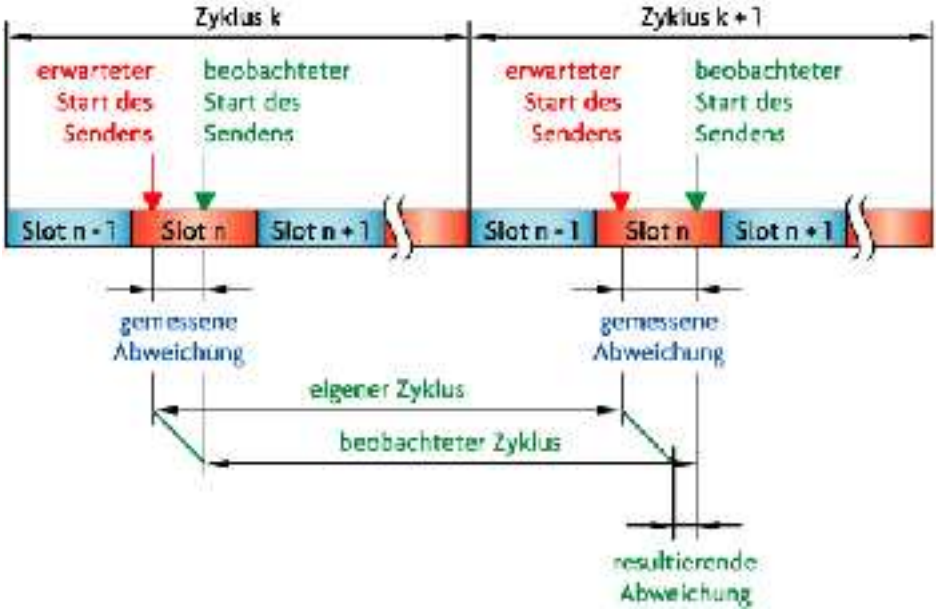


Abbildung 10.3: Zur Bestimmung des Offsets wird eine Zeitmessung benötigt, zur Bestimmung der Steigungsdifferenz zwei

Diese Lücke soll verhindern, daß es im Fall nicht ganz exakt synchroner Knoten nicht sofort zu einer Kollision der Nachrichten kommt. Durch den Interframe Gap wird jedoch die nutzbare Bandbreite verringert, daher sollte dieser möglichst klein ausfallen, was wiederum eine möglichst exakte Synchronisation voraussetzt.

Bei der Synchronisation von Uhren gilt es zwei Korrekturen vorzunehmen [3]. Das eine Ziel liegt darin, daß alle Uhren gleich schnell gehen, das heißt also, daß die Kommunikationszyklen aller Knoten gleich lange dauern. Dies ist die sogenannte Steigungs-Korrektur. Nach dieser werden die Uhren in der Regel trotzdem noch nicht die gleichen Werte zur gleichen Zeit anzeigen. Es muß nämlich noch durch die sogenannte Offset-Korrektur dafür gesorgt werden, daß auch alle Kommunikationsrunden zeitgleich beginnen. Erst dann wird Synchronisation der Uhren vorliegen.

Um die Steigungsdifferenz bzw. den Offset zu ermitteln, werden in einem FlexRay Netzwerk bestimmte Referenzknoten vereinbart, welche fortwährend Synchronisationsnachrichten aussenden. Durch die eigene Uhr erwartet nun jeder Knoten zu einem bestimmten Zeitpunkt das Eintreffen einer solchen Synchronisationsnachricht. Der tatsächlich beobachtete Zeitpunkt des Eintreffens kann nun aber vom erwarteten abweichen. Diese Abweichung entspricht dem Offset. Wird im darauffolgenden Zyklus wieder diese Abweichung bestimmt, ergibt die Abweichung beider Offset-Werte die Steigungsdifferenz zwischen Sende- und Empfangsknoten. Abbildung 10.3 verdeutlicht die Messung der Zeitabweichungen.

Das Festlegen mehrerer Referenzknoten hat den Nachteil, daß keine ganz exakte Synchronisation erreicht wird, da die Referenzknoten auch nicht völlig gleich laufen können.

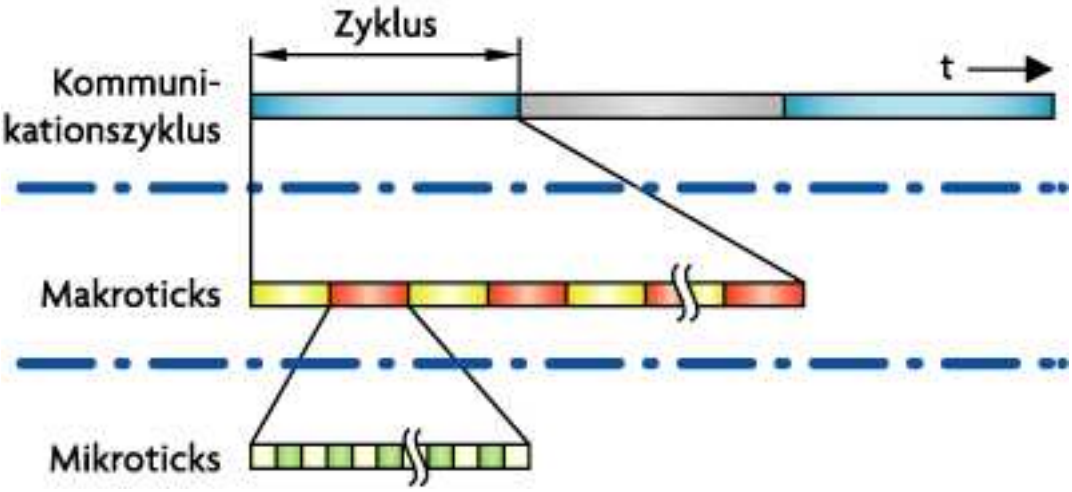


Abbildung 10.4: Die Zeithierarchie bei FlexRay

Es bringt aber den sicherheitstechnischen Vorteil, daß man sich nicht auf eine Referenz verlassen muß, welche ja auch ausfallen könnte.

Hat ein Knoten die Abweichungen seiner Uhr zu der der Referenzknoten ermittelt, muß er noch seine Uhr entsprechend korrigieren. Hierfür wird der Mittelwert der Abweichungen zu den einzelnen Referenzen verwendet. Ein Kommunikationszyklus bei FlexRay besteht aus einer bestimmten Anzahl Makroticks (Abbildung 10.4). Diese bestehen wiederum aus Mikroticks, welche direkt aus den Quarztakten der einzelnen Knoten bestimmt werden. Bei der Offset-Korrektur wird nun der vorletzte Makrotick eines Zyklus entsprechend verkürzt bzw. verlängert, so daß der darauf folgende genau zur selben Zeit beginnt, wie der Referenzzyklus. Während dieser Korrektur dürfen keine Nachrichten versendet werden.

Für die Korrektur der Steigungsdifferenz berechnet ein Knoten, um wieviele Mikroticks sein Zyklus verkürzt bzw. verlängert werden muß, um die selbe Länge wie die des Referenzzyklus zu erreichen. Diese Anzahl an Mikroticks wird dann gleichmäßig über den gesamten Kommunikationszyklus aufgeteilt und die Korrektur vorgenommen.

10.6 Zusammenfassung und Ausblick

FlexRay ist ein Fahrzeug-Bussystem, daß sich durch Realzeitfähigkeit und eine hohe Fehlertoleranz auszeichnet. Durch einen Bus Guardian wird das Auftreten von “Bubbling Idiots” verhindert, Zweige auf denen Fehler auftraten, können durch den Aktiven Stern abgeschaltet werden, und die Uhrensynchronisation verläßt sich auf mehrere Referenzknoten. Außerdem unterstützt der FlexRay-Controller einen zweiten Redundanzkanal. Somit ist es sehr wahrscheinlich, daß zukünftige Drive-by-Wire Automobile auf diesem System basieren werden, nicht zuletzt auch wegen seiner wirtschaftlichen Vorteile gegenüber Alternativen wie dem TTP-Bus.

Das FlexRay-Konsortium wurde erst 1999 gegründet und die Entwicklung ist noch keineswegs abgeschlossen. Die auf der Homepage des Konsortiums [4] veröffentlichte Roadmap sieht vor, daß im ersten Quartal 2003 erste FPGA Versionen des Kommunikations-Controllers, des Transceivers und des Bus Guardians zur Verfügung stehen. Im vierten Quartal 2004 sollen dann sogenannte "Automotive Qualified Silicon" Versionen der drei Komponenten eines FlexRay-Controllers bereit stehen. Dies sind Bauteile, die für den Einsatz in Serienfahrzeugen geeignet sind.

Literaturverzeichnis

[1] Elend, B.: *FlexRays Electrical Physical Layer* Elektronik Automotive, Jan. 2003

[2] Heinecke, H.; Schedl, A; Berwanger, J.: *FlexRay - Ein Kommunikationssystem für das Automobil der Zukunft* Elektronik Automotive, Sept. 2002

[3] Rausch, M.: *Optimierte Mechanismen und Algorithmen in FlexRay* Elektronik Automotive, Dez. 2002

[4] FlexRay - Homepage: *www.flexray.com*

Kapitel 11

Sensor-/Aktorfusion

Autor: Gunther Nagy

Betreuer: Dipl.-Ing. Benito Liccardi

11.1 Einführung

Ein Fahrzeug kann sich nur dann autonom im Strassenverkehr bewegen, wenn ihm hinreichende Informationen über sich selbst, die Strasse und vor allem über Objekte und Hindernisse in der Fahrzeugumgebung zur Verfügung stehen. Diese Informationen gewinnt das Fahrzeug durch Verwendung unterschiedlicher Sensoren.

Sensoren müssen folgende Elemente der Fahrumgebung erfassen:

- Fahrbahn (Verlauf und Reibwerte)
- Verkehrszeichen
- Randbebauung
- stehende Objekte
- andere Fahrzeuge
- ungeschützte Verkehrsteilnehmer (z. B. Fussgänger)
- Witterung (z. B. Temperatur, Niederschlag)

Für Objekte (z. B. andere Fahrzeuge, Fussgänger, Hindernisse) müssen u. a. Position, Abstand zum eigenen Fahrzeug, Geschwindigkeit, Größe und Bewegungsrichtung erfasst werden.

Abbildung 11.1 auf Seite 160 gibt einen Überblick über die Sensoren die zur Fahrumgebungserfassung eingesetzt werden können. Teilweise werden diese Sensoren bereits heute zur Realisierung von Fahrerassistenzsystemen verwendet. Beispielsweise werden in Parkassistenzsystemen Ultraschallsensoren verbaut.

In den weiteren Abschnitten dieser Arbeit werden die einzelnen Sensoren näher betrachtet. Die Funktionsweisen der Sensoren, die möglichen Einsatzgebiete, ihre Vorteile aber auch ihre Schwachstellen werden erläutert.

Da sich Fahrerassistenzsysteme und erst recht „autonome Fahrzeuge“ noch in der Entwicklung befinden und keineswegs vollständig ausgereift sind, gibt es nur wenige detailierte Informationen über den Einsatz von Sensoren zur Fahrumgebungserfassung. Die vorliegende Arbeit erhebt deshalb keinen Anspruch auf eine erschöpfende Betrachtung des Themas. Ausserdem werden nicht alle denkbaren Sensoren, sondern nur die nach Meinung des Autors wichtigsten Sensoren betrachtet.

11.2 Navigation

Damit sich ein Fahrzeug autonom von seinem gegenwertigen Standort zu einem anderen Standort bewegen kann, muss es zunächst eine geeignete Route, die diese beiden Orte verbindet, auswählen. Dazu wird bereits heute das vom amerikanischen Militär entwickelte,

weltweit verfügbare Satellitennavigationssystem GPS (Global Positioning System) verwendet. Mit Hilfe des GPS wird die gegenwertige Position auf einer digitalen Landkarte bestimmt und davon ausgehend eine Route zum Ziel ausgewählt. Während der Fahrt kann auf äussere Umstände (z. B. Staus) reagiert werden und eine alternative Route ausgewählt werden.

Die Genauigkeit der Navigation hängt im wesentlichen von zwei Faktoren ab: Der Qualität des digitalen Kartenmaterials und der Ortungsgenauigkeit des GPS. Mit gewöhnlichem GPS erreicht man heute Ortungsgenauigkeiten von etwa 10 m. Mit sog. differentiellem GPS (DGPS) wird eine Ortungsgenauigkeit von ca. 1 cm erreicht.

DGPS verwendet, zusätzlich zum normalen GPS, die Daten ortsfester GPS-Empfänger an genau bekannten Positionen als Referenz. Durch vergleich der aktuell empfangenen GPS-Daten mit den bekannten, sehr genauen Daten, ermittelt die ortsfeste Station den aktuellen Fehler in der Positionsbestimmung. Diese Daten werden per Funk an die mobilen GPS-Empfänger übertragen.

Die wesentlichen Vorteile von GPS sind die Einsetzbarkeit in schmutzigen Umgebungen und die Unabhängigkeit von den Witterungsverhältnissen. Allerdings wird die Funktionsfähigkeit von GPS durch Reflexionen und Abschattungen an Gebäuden oder sogar Bäumen teilweise stark eingeschränkt. [2]

Denkbar ist aber auch der Einsatz der Navigation zur Spurführung. Bei einer Genauigkeit der Positionsbestimmung von ca. 1 cm könnte ein Fahrzeug die Fahrbahn- bzw. die Fahrspur-Grenzen allein aufgrund von Navigationsdaten einhalten.

Die Navigation ist aber auch aus einem anderen Grund von Bedeutung: Wenn die statische Umgebung (alle nicht beweglichen Objekte) des Fahrzeugs bekannt ist, kann die Auswertung der Sensordaten einfacher und zuverlässiger erfolgen. „Die Interpretation der aktuell gemessenen [Sensor-]Signale könnte in einem wirklichkeitsnaheren Kontext erfolgen und nicht statische (bewegliche) Objekte besser von statischen separiert werden“[12]. Die Daten der statischen Objekte müssten digitalen Karten entnommen werden.[12]

11.3 Ultraschall-Sensoren

Wie Abbildung 11.1 auf Seite 160 entnommen werden kann, eignen sich Ultraschall-Sensoren zur Bestimmung des Abstands von Objekten vor und hinter dem Fahrzeug. Mit Ultraschall-Sensoren kann der „Ultranah“-Bereich bis etwa 1,5 m überwacht werden.

Zur Entfernungsmessung mittels Ultraschall wird die Laufzeit von Schallimpulsen bestimmt. Diese Schallimpulse haben Frequenzen von ca. 20 bis 200 kHz und werden mit Hilfe piezoelektrischer Elemente gesandt und empfangen. Schall breitet sich nicht entlang einer Linie und auch nicht in einer Ebene, sondern in Form einer sog. dreidimensionalen Schallkeule aus. Dies hat den Vorteil, dass ein dreidimensionaler Raum und nicht nur eine Ebene abgetastet wird. Andererseits ergibt sich dadurch eine schlechte Winkelauflösung,

da sich ein Objekt irgendwo auf einem Kreisbogen um den Mittelpunkt des Sensors befinden kann. Der Öffnungswinkel einer Ultraschallkeule liegt meist zwischen 5° und 30°. Je grösser der zu messende Abstand ist, desto Grösser wird die Unsicherheit über den genauen Abstand.

Der Erfassungsbereich von Ultraschall-Sensoren liegt typischerweise zwischen 30cm und 10m, bei einer Genauigkeit der Entfernungsmessung von ca. 1%. Im automotive Bereich werden Ultraschall-Sensoren aber, wie bereits erwähnt, nur für Entfernungen bis ca. 1,5 m eingesetzt.

Ultraschall-Sensoren können auch bei widrigen Umweltbedingungen eingesetzt werden. Schmutzpartikel in der Luft oder dünne Ablagerungen auf der Sensoroberfläche beeinflussen die Genauigkeit des Sensors nicht. Ausserdem erkennen Ultraschallsensoren Objekte unabhängig von deren Material und Farbe sehr zuverlässig. Probleme bereiten diesen Sensoren lediglich weiche Materialien, wie z. B. Stoff, da diese Materialien den Schall stark streuen und dämpfen.

Die größten Vorteile von Ultraschall-Sensoren sind ihr geringer Preis und ihre handliche Baugröße, wodurch sie nahezu überall montiert werden können.

Die Schallgeschwindigkeit in Luft ist Abhängig von der Luftfeuchte, dem Luftdruck und der Lufttemperatur, wobei die Temperatur den grössten Einfluss hat. Deshalb muss beim Einsatz von Ultraschall-Sensoren zumindest die Lufttemperatur berücksichtigt werden. Da die Ausbreitungsgeschwindigkeit von Schallwellen, verglichen mit der Lichtgeschwindigkeit, relativ gering ist, können Ultraschall-Sensoren nicht für zeitkritische Messungen eingesetzt werden.

Probleme mit Ultraschall-Sensoren ergeben sich, wenn mehrere dieser Sensoren zeitlich und örtlich benachbart eingesetzt werden: Es kann zu einem Übersprechen kommen, d. h. ein Sensor nimmt den von einem anderen Sensor emittierten Schall wahr. Dieses Problem kann durch Frequenzmodulation der ausgesandten Schallwellen beseitigt werden.

Ebefalls problematisch sind Mehrfachechos zwischen Sensor und Messobjekt, sog. Spiegelreflexionen. Glatte Oberflächen leiten den Schall um, so dass der Schall nicht vom eigentlich zu detektierenden Objekt, sondern von einem anderen Objekt zum Sender zurückreflektiert wird. Dadurch werden falsche Entfernungen gemessen.

Desweiteren kann ein Ultraschall-Sensor nicht im unmittelbaren Bereich vor dem Sensor detektieren. Dies liegt daran, dass zum Senden und Empfangen des Schalls das selbe Piezoelement eingesetzt wird. Nach dem Senden des Schalls muss also auf Empfang umgeschaltet werden. In dieser Zeit kann nicht gemessen werden.

Erfolgreich eingesetzt werden Ultraschall-Sensoren heutzutage beispielsweise in sog. Parkassistenzsystemen. Diese Parkassistenzsysteme warnen den Fahrer wenn er Gefahr läuft beim Parken auf ein Hinderniss auf zu fahren. [3][5][4]

11.4 Radar-Sensoren

Radar[1]-Sensoren verwenden elektromagnetische Wellen um Objekte zu orten: Eine Antenne sendet elektromagnetische Wellen aus und in der Strahlungsrichtung befindliche Objekte reflektieren diese. Dadurch gelangt ein Teil der ausgesandten Leistung zum Sender zurück. Durch Auswertung von Frequenz, Laufzeit und Leistung des Echosignals kann der Sender Daten des Objekts, wie z. B. die Geschwindigkeit, die Bewegungsrichtung oder auch die Größe, bestimmen. [10]

Ähnlich wie Schall, breitet sich die für Radar-Sensoren eingesetzte elektromagnetische Strahlung nicht entlang einer Linie und auch nicht in einer Ebene, sondern in Form einer sog. dreidimensionalen Radarkeule aus. Einerseits hat dies, wie bei Ultraschall-Sensoren, den Vorteil, dass nicht nur eine Ebene, sondern ein dreidimensionaler Raum abgetastet wird. Andererseits ergibt sich dadurch eine schlechte Winkelauflösung, da sich ein Objekt irgendwo auf einem Kreisbogen um den Mittelpunkt des Sensors befinden kann. Der Öffnungswinkel einer Radarkeule kann jedoch in einem großen Bereich eingestellt werden. Dazu später mehr.

Der Große Vorteil von Radar-Sensoren ist ihre weitgehende Unempfindlichkeit gegenüber Dunkelheit, Verschmutzung oder sonstigen Witterungsverhältnissen [7]. Radar-Sensoren können in Fahrzeugen zur Erfassung von Objekten in zwei Entfernungsbereichen eingesetzt werden (vgl. [3]):

- Fernbereich bis ca. 120m
- Nahbereich bis ca. 14m

11.4.1 Long-Range-Radar-Sensoren

Für den Fernbereich werden dabei sog. Long-Range-Radar-Sensoren (LRR-Sensoren) mit Arbeitsfrequenzen von 77 Gigahertz verwendet. Die Antennen dieser LRR-Sensoren sind für eine hohe Verstärkung und ein gutes SNR (signal-to-noise ratio) ausgelegt. Dies führt aber dazu, dass die Radarkeulen, die diese Antennen erzeugen, einen sehr kleinen Öffnungswinkel haben (typisch ca. 10° bis 15°). Da die LRR-Antennen fest in eine bestimmte Richtung strahlen, führt dies dazu, dass nur ein sehr schmaler Bereich vor dem Fahrzeug abgetastet werden kann. Ausserdem sind LRR-Sensoren auch heute noch sehr teuer.[13]

Die Geschwindigkeit von Objekten kann mit LRR-Sensoren auf etwa 1 km/h genau bestimmt werden, der Abstand von Objekten kann mit einer Toleranz von ca. 5% des Abstandes ermittelt werden. [14]

Um Objekte zuverlässig detektieren zu können werden heutzutage meist drei fest ausgerichtete Radarkeulen eingesetzt die unabhänig voneinander Objekte erfassen. Die LRR-Technologie wird bereits heute sehr sicher beherrscht. Mit LRR-Sensoren werden heute

[1] Radar: Radio Detection and Ranging

Abstand und Geschwindigkeit vorausfahrender Fahrzeuge überwacht. Dies ist beispielsweise in ACC[2]-Systemen der Fall.

11.4.2 Short-Range-Radar-Sensoren

Für den Nahbereich werden sog. Short-Range-Radar-Sensoren (SRR-Sensoren) verwendet, die im Frequenzbereich von 24 Gigahertz arbeiten. Wie Abbildung 11.1 auf Seite 160 zeigt, kann mit SRR-Sensoren das ganze nähere Umfeld des Fahrzeugs überwacht werden.

SRR-Sensoren können so konstruiert werden, dass sie einen sehr großen Winkelbereich abtasten. Werden mehrere dieser Sensoren um das Fahrzeug herum angebracht, lässt sich das ganze nähere Umfeld des Fahrzeugs überwachen. Allerdings lässt sich, wie bereits erwähnt, zunächst nichts über die genaue Lage von Objekten aussagen. Wenn die Sensoren allerdings so angebracht sind, dass sich ihre Erfassungsbereiche überlappen, kann die genaue Lage von Objekten mittels Triangulation bestimmt werden.

Die Geschwindigkeit von Objekten kann mit SRR-Sensoren auf etwa 0,5 km/h genau bestimmt werden, der Abstand von Objekten kann mit einer Toleranz von ca. 5% des Abstandes ermittelt werden. [14]

SRR-Sensoren werden in Europa in naher Zukunft wahrscheinlich nicht in Fahrzeuge verbaut, da es aufgrund der verwendeten Frequenz und der benötigten Bandbreite (ca. 8 GHz) Probleme bei der Zulassung gibt. In Amerika sind SRR-Sensoren allerdings schon genehmigt.

SRR-Sensoren haben niedrigere Anschaffungskosten als LRR-Sensoren [13]. Allerdings erforden die Antennen von SRR-Sensoren eine aufwendige Feinjustierung nach der Montage am Fahrzeug. Deshalb sind die Gesamtkosten für SRR-Sensoren derzeit noch höher als für LRR-Sensoren.

11.5 Lidar-Sensoren

Methoden, um Entfernungen und Geschwindigkeiten mit Hilfe von Laser zu bestimmen, werden oft auch Lidar (Light Detection and Ranging, analog zu Radar) genannt.

Lidar-Sensoren können in autonomen Fahrzeugen für zwei Dinge eingesetzt werden: Zur Spurführung und für die Objekterkennung und Objektverfolgung. Prinzipbedingt haben Lidar-Sensoren den Nachteil, dass sie sehr empfindlich auf Schmutz, Staub, Nebel oder auch Regen reagieren, d. h. ihre Leistungsfähigkeit wird stark eingeschränkt. Wie später gezeigt wird, gibt es für einige dieser Einschränkungen aber bereits Lösungen.

[2]ACC: Adaptive Cruise Control

11.5.1 Spurführung mit Lidar-Sensoren

Ein am Fahrzeug angebracheter Lidar-Sensor rotiert um seine senkrechte Achse, sendet dabei einen Laserstrahl aus und empfängt das an Reflexionsmarken reflektierte Licht wieder. Dabei wird der Winkel der Reflexionsmarke in Bezug auf die Fahrzeuglängsachse und die Entfernung zur Marke bestimmt. Damit lässt sich die Position des Fahrzeugs auf der Strecke bestimmen und die Spur kann gehalten werden.

Als Reflexionsmarken kommen künstliche und natürliche Reflektoren in Frage. Künstliche Reflektoren sind z. B. extra angebrachte Marken oder Leitplanken, natürliche Reflektoren sind z. B. Wände und Bordsteine. Mit dem beschriebenen Verfahren lässt sich die Position eines Fahrzeugs mit einer Genauigkeit von ±10 mm bestimmen. [2]

11.5.2 Obejekterkennung und Objektverfolgung mit Lidar-Sensoren

Laserscanner

Um zu zeigen was sog. Laserscanner bereits heute leisten können, wird hier beispielhaft einer vorgestellt: Der ALASCA (Automotiver Laserscanner) der Fa. IBEO Automotive Sensor GmbH. Abbildung 11.3 auf Seite 161 zeigt diesen Laserscanner.

„Die Grundlage [des ALASCA] bildet ein rotierender infraroter (IR) Lichtstrahl. Mit dessen Hilfe werden über eine Pulslaufzeitmessung die Objektkonturen in der Fahrzeugumgebung erfasst. Die Messdaten-Vorverarbeitung extrahiert und klassifiziert mögliche Objekte. Für jedes Objekt werden der Objekttyp, die Entfernung, Richtung, Geschwindigkeit und Beschleunigung zur weiteren Signalverarbeitung angegeben“ [7]. Die erfassten Objekt-Daten werden beispielsweise auf dem CAN[3]-Bus ausgegeben und können damit von nachgeschalteten Applikationen ausgewertet werden.

Abbildung 11.4 auf Seite 162 zeigt den Aufbau des ALASCA.

In [7] wird die Funktionsweise des ALASCA wie folgt beschrieben:

- Rotierender Spiegel überträgt IR Strahl.
- IR Strahl wird vom Zielobjekt reflektiert.
- Fotodiode empfängt den reflektierten Strahl.
- Laufzeitmessung liefert die Objektdistanz.
- Winkelencoder des Spiegelmotors liefert Winkelauflösung.
- Weitere Berechnung der Objektgeschwindigkeit und -beschleunigung.
- Objekterfassung und -verfolgung in Abhänigkeit der gewünschten Applikation.

[3]CAN: Controller Area Network

Der ALASCA kann Objekte im Abstand von 0,3 bis 80 m verfolgen und Objekte im Abstand von 0,3 bis 256 m detektieren. Die Entfernung wird dabei auf ±1 cm genau gemessen. Er deckt einen horizontalen Bereich von 240° ab und liefert dabei, abhänig von der Rotationsfrequenz, eine Winkelauflösung von 0,25° bis 1°. Der ALASCA tastet vertikal einen Bereich von 3,2° ab, wobei dieser Winkel nicht kontinuierlich, sondern in vier Ebenen abgetastet wird (vgl. 11.5 auf Seite 162). Diese vier Ebenen werden unabhänig voneinander abgetastet und die Objekte in diesen vier Ebenen werden ebenfalls unabhängig voneinander klassifiziert. Auf diese Weise kann die Nick-Bewegung des Fahrzeugs, beispielsweise beim Durchfahren eines Schlaglochs, ausgeglichen werden. Ausserdem liefert der Scanner dadurch auch bei der Fahrt von einer Ebene in eine Steigung noch verwertbare Daten.

Der ALASCA kann zwei reflektierte Strahlen pro Messung verarbeiten und daraus zwei Distanzwerte errechnen. Dadurch kann er auch Objekte hinter Regentropfen oder verschmutzten Abdeckungen erkennen. Da es spezielle Kunststoffe gibt, die für Infrarot-Licht durchlässig sind, kann der ALASCA auch hinter Abdeckungen montiert und damit einfach in das Design eines Fahrzeug integriert werden.

Der ALASCA kann der Laser Klasse 1 zugeordnet werden und ist damit für das menschliche Auge ungefährlich. [9][8][7]

Mehrstrahlige Lidar-Sensoren

Mit mehrstrahligen Lidar-Sensoren können analog zu LRR-Sensoren Geschwindigkeit und Entfernung vorausfahrender Fahrzeuge ermittelt werden.

Abbildung 11.6 auf Seite 162 veranschaulicht das Prinzip mehrstrahliger Lidar-Sensoren. Zur Messung werden typischerweise 16 Infrarotstrahlen ausgesandt. Im Gegensatz zu Laserscannern sind diese 16 Strahlen fest in eine bestimmte Richtung ausgerichtet. Durch Auswertung der von vorausfahrenden Objekten reflektierten Strahlen können Geschwindigkeit und Abstand dieser Objekte bestimmt werden. Da je nach Größe des Objekts evtl. nicht alle Strahlen von diesem Objekt reflektiert werden, kann auch die Position des Objekts relativ zum eigenen Fahrzeug bestimmt werden.

Mehrstrahlige Lidar-Sensoren können ähnlich wie LRR-Sensoren eingesetzt werden und haben auch eine vergleichbare Performance. Allerdings hat die Lidar-Technologie gegenüber der Radar-Technologie eindeutige Kostenvorteile. [7]

11.6 Kamera-Sensoren

Für Kameras sind vielfältige Einsatzgebiete denkbar. Kameras in Fahrzeugen werden jedoch mit ständig wechselnden und sehr unterschiedlichen Lichtverhältnissen konfrontiert (z. B. Dunkelheit im Tunnel, Gegenlicht bei tief stehender Sonne). „Gegenwärtig verfügbare Videokameras kommen mit diesem Spektrum wechselnder Lichtverhältnisse noch nicht

zufriedenstellend zurecht“ [2]. Ausserdem haben Kameras Probleme mit Verschmutzungen.

11.6.1 Spurführung mit Kamera-Sensoren

Für die Spurführung mit Kameras gibt es grundsätzlich zwei Ansätze. Beim ersten Ansatz erfasst eine unter dem Fahrzeug angebrachte Kamera eine extra auf der Fahrbahnmitte angebrachte Linie. Durch Auswertung der Bildinformation kann die Abweichung von dieser Linie bestimmt werden. Dieser Ansatz hat aber zwei Nachteile: Zum Einen kann eine starke Verschmutzung der Fahrbahn dazu führen, dass die Linie nicht mehr erkannt werden kann. Zum Anderen muss die Linie, wie bereits erwähnt, extra angebracht werden. Deshalb ist dieser Ansatz für Strassenfahrzeuge ungeeignet. Allerdings lässt er sich gut für Fahrerlose Transportfahrzeuge in sauberer Umgebung, z. B. in Lagerhallen, einsetzten. [2]

Beim zweiten Ansatz kommt als Sensor „eine im Bereich der Windschutzscheibe angebrachte Kamera zum Einsatz, die den Fahrbahnbereich bis ca. 50 m vor dem Fahrzeug erfasst„ [7]. Mittels digitaler Bildverarbeitung wird die Fahrbahn von der sonstigen Umgebung unterschieden. Auch dieser Ansatz erfodert zunächst gut strukturierte Umgebungen, wie Autobahnen oder Bundesstrassen, in denen die Fahrspur klar von der restlichen Umgebung abgegrenzt ist. Im Nutzfahrzeugbereich wird mit der beschriebenen Methode bereits heute ein sog. „Spurverlassenswarner“ implementiert, der den Fahrer auf der Autobahn warnt, wenn er Gefahr läuft die Spur unbeabsichtigt zu verlassen.

11.6.2 Objekterkennung mit Kamera-Sensoren

Zur Objekterkennung mittels Kameras gibt es im wesentlichen zwei Ansätze:

- Monobasierte Verfahren
- Stereobasierte Verfahren

„Stereobasierte Verfahren werten mehrere, gleichzeitig aus verschiedenen Positionen aufgenommene Bilder aus, um aus den unterschiedlichen Projektionen durch Triangulierung die Information über die Außenwelt zu gewinnen“ [15]. Monobasierte Verfahren „werten zu demselben Zweck mehrere, nacheinander von einer Kamera aufgenommene Bilder aus“ [15].

11.6.3 Infrarotkameras

Die Qualität der Information die mit Kameras über die Umgebung gewonnen werden kann, hängt, wie bereits erwähnt, stark von den gerade vorherschenden Lichtverhältnissen ab. Ein denkbarer Weg dies zu umgehen ist der Einsatz von Infrarotkameras: Der zu erfassende Bereich der Umgebung wird mit ungebündeltem Infrarotlicht fernlichtartig ausgeleuchtet

und das von Objekten zurückgestreute Licht wird mit Infrarotkameras wieder eingefangen. Auf diese Weise kann ein Graustufenbild der Umgebung gewonnen werden.

Derzeit ist der Einsatz von Infrarotkameras allerdings nur zur Verbesserung der Nachsicht geplant. Das aufgenommene Graustufenbild soll dem Fahrer angezeigt werden, damit er sich Nachts einen besseren Überblick über mögliche Hindernisse oder den Strassenverlauf verschaffen kann.

Wird für das ausgesandte Infrarotlicht eine Frequenz-Modulation verwendet, kann dieses vom Infrarotanteil des Sonnenlichts unterschieden werden. Damit wäre ein Einsatz von Infrarotkameras auch tagsüber denkbar. Ob dies sinnvoll ist, wird sich erst noch zeigen. [7] [4]

11.7 Sensorfusion

Heutige Fahrerassistenzsysteme bestehen meist aus einem Sensor und einer nachgeschalteten Applikation, die die Sensordaten auswertet und darauf basierend eine Fahrerassistenzfunktion realisiert. Abbildung 11.7 auf Seite 163 veranschaulicht diesen Sachverhalt.

Wie die vorangegangenen Abschnitte gezeigt haben, haben alle Sensoren die zur Fahrumgebungserfassung in Frage kommen Vor- und Nachteile. Ein so komplexes System wie ein autonomes Fahrzeug, lässt sich mit dem Prinzip heutiger Fahrerassitenzssysteme nicht realisieren. Um den Traum vom autonomen Fahren in der Zukunft realisieren zu können, muss ein anderer Weg gegangen werden: Die sog. Sensorfusion.

Bei der Sensorfusion werden die einzelnen Sensoren, bzw. die Daten die sie liefern, nicht mehr unabhängig voneinander betrachtet. Vielmehr werden die Einzelsensoren mit ihren verschiedenen Messprinzipien, Erfassungsbereichen, Genauigkeiten und Einsatzgebieten in einer geeigneten Netzwerkarchitektur miteinander verknüpft.

„Die Vielzahl der Messwerte aller Einzelsensoren wird mit Hilfe eines Fusionsprozesses zu einem konsistenten synthetischen Abbild der Fahrumgebung verschmolzen. Dieses synthetische Abbild steht dann als sogenanntes Umfeldmodell ... zur Verfügung“ [1]. Zu diesem Umfeldmodell „[...] gehört auch eine symbolische, klassifizierende Beschreibung der Szenendynamik, wie zum Beispiel: Fahrzeug in der linken Fahrspur überholt vorausfahrendes Fahrzeug “ [1]. Nachgeschaltete Applikationen können auf das Umfeldmodell zurückgreifen und damit die Funktionen, die für autonomes Fahren notwendig sind, realisieren.

Durch die Verknüpfung der Einzelsensoren kann die Fahrumgebung viel umfassender und zuverlässiger erfasst werden, als dies mit isolierten Komponenten möglich ist.

In Abbildung 11.8 auf Seite 164 ist die Vernetzung der Einzelkomponenten dargestellt. Die dort erwähnten Applikationen werden teilweise bereits heute in Fahrzeugen eingesetzt. Allerdings greifen diese heute noch nicht auf ein Umfeldmodell zurück.

Das Thema „Sensorfusion“ befindet sich derzeit noch stark in der Forschung und ist noch nicht soweit ausgereift, dass es tatsächlich eingesetzt werden kann. Ohne Sensorfusion ist

autonomes Fahren aber nicht denkbar. [1]

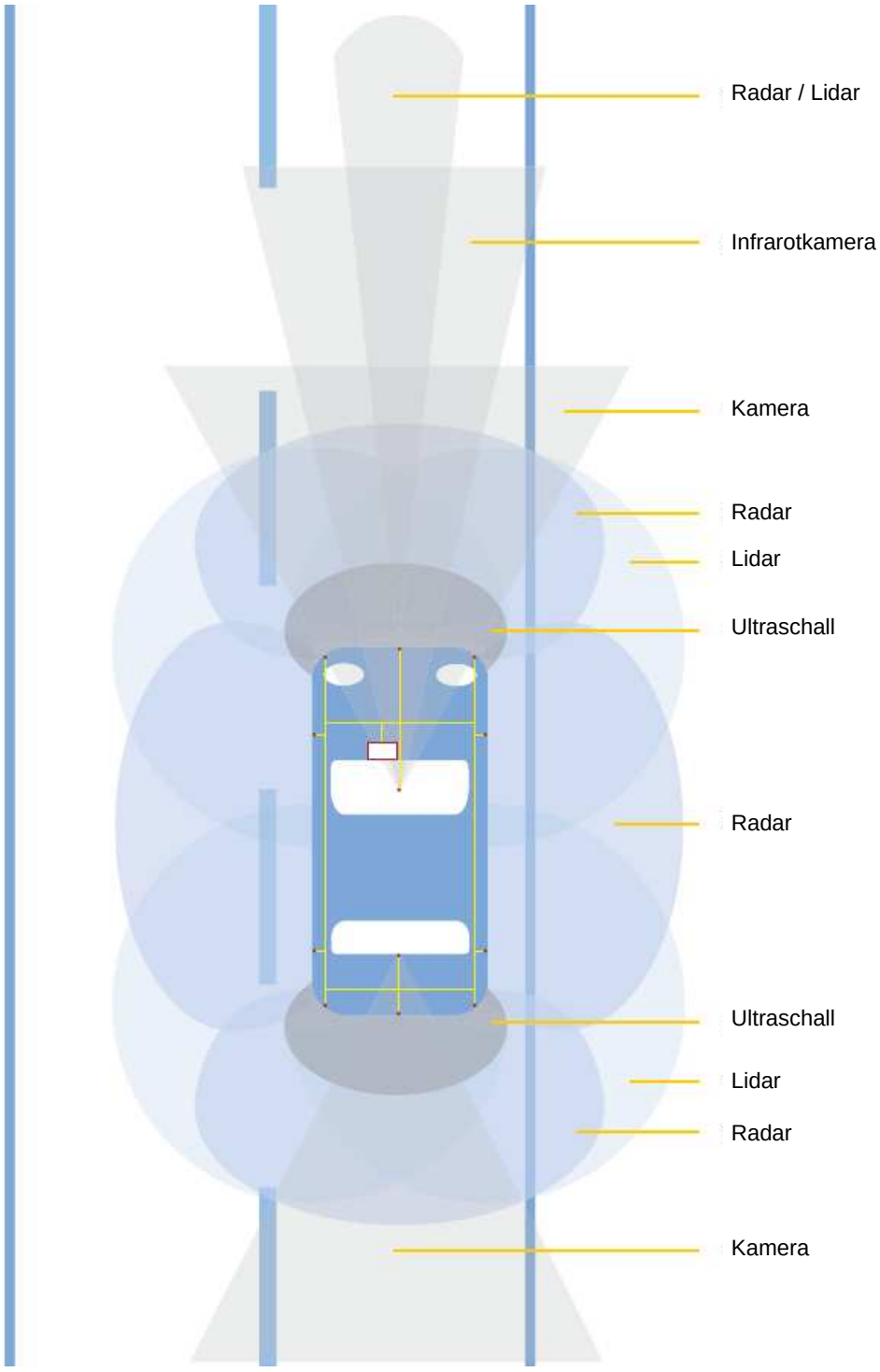
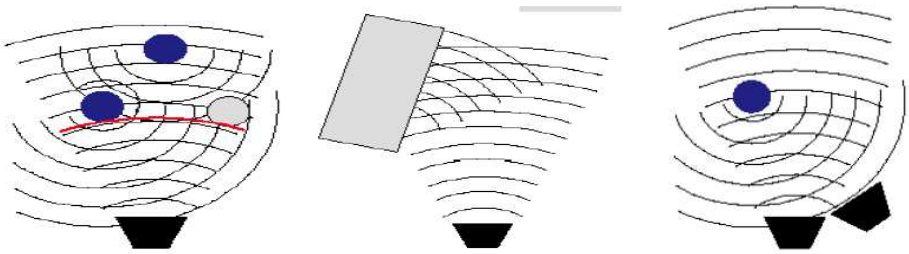


Abbildung 11.1: Sensoren zur Umgebungserfassung [1]



(a) Winkelgenauigkeit (b) Spiegelreflexion (c) Übersprechen

Abbildung 11.2: Probleme bei Ultraschall-Sensoren [4]

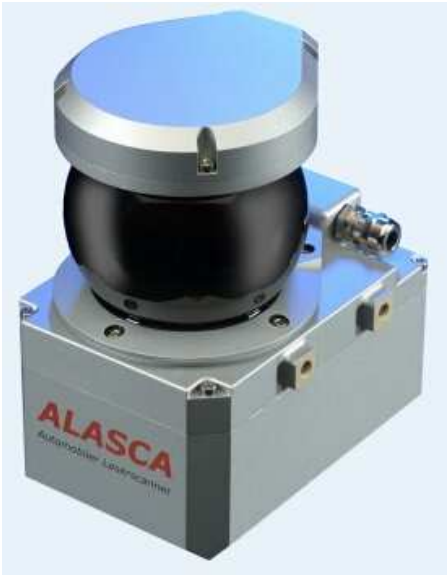


Abbildung 11.3: Der ALASCA Laserscanner [8]

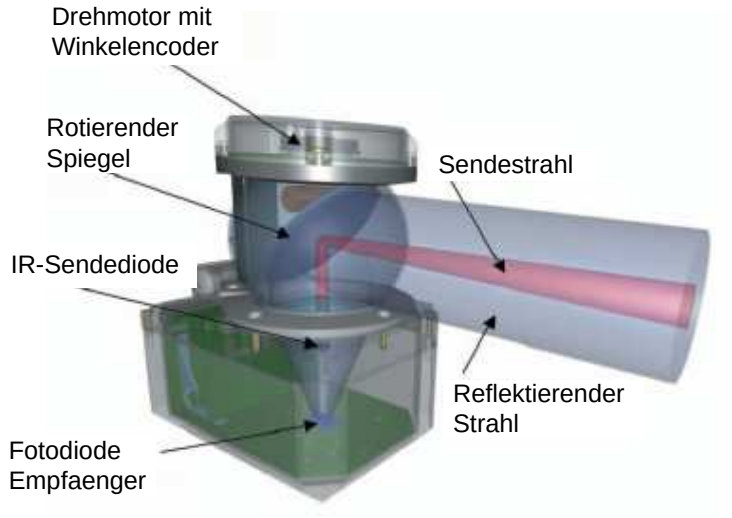


Abbildung 11.4: Aufbau des ALASCA [7]

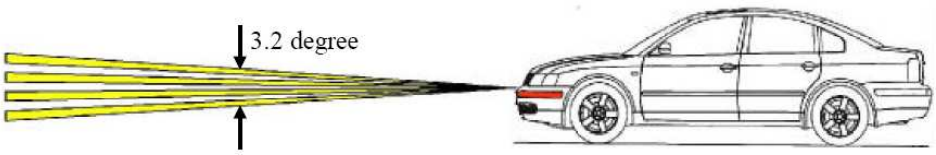


Abbildung 11.5: ALACA: Vertikale Abtast-Ebenen [11]

Abbildung 11.6: Funktionsprinzip mehrstrahliger Lidar-Sensoren [7]

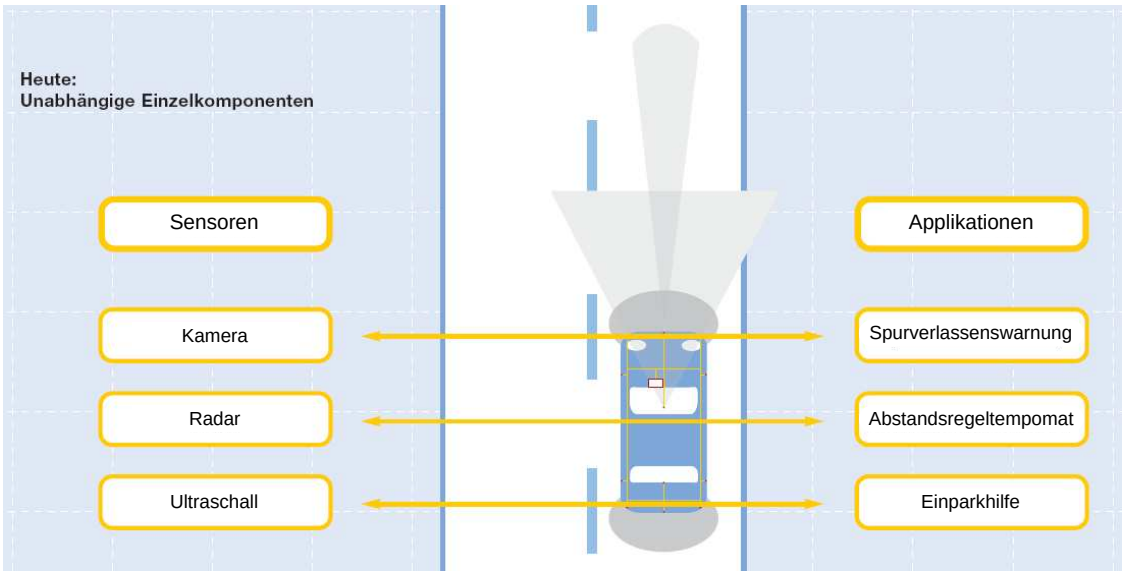


Abbildung 11.7: Heutige Fahrerassistenzsysteme: Unabhängige Einzelkomponenten [1]

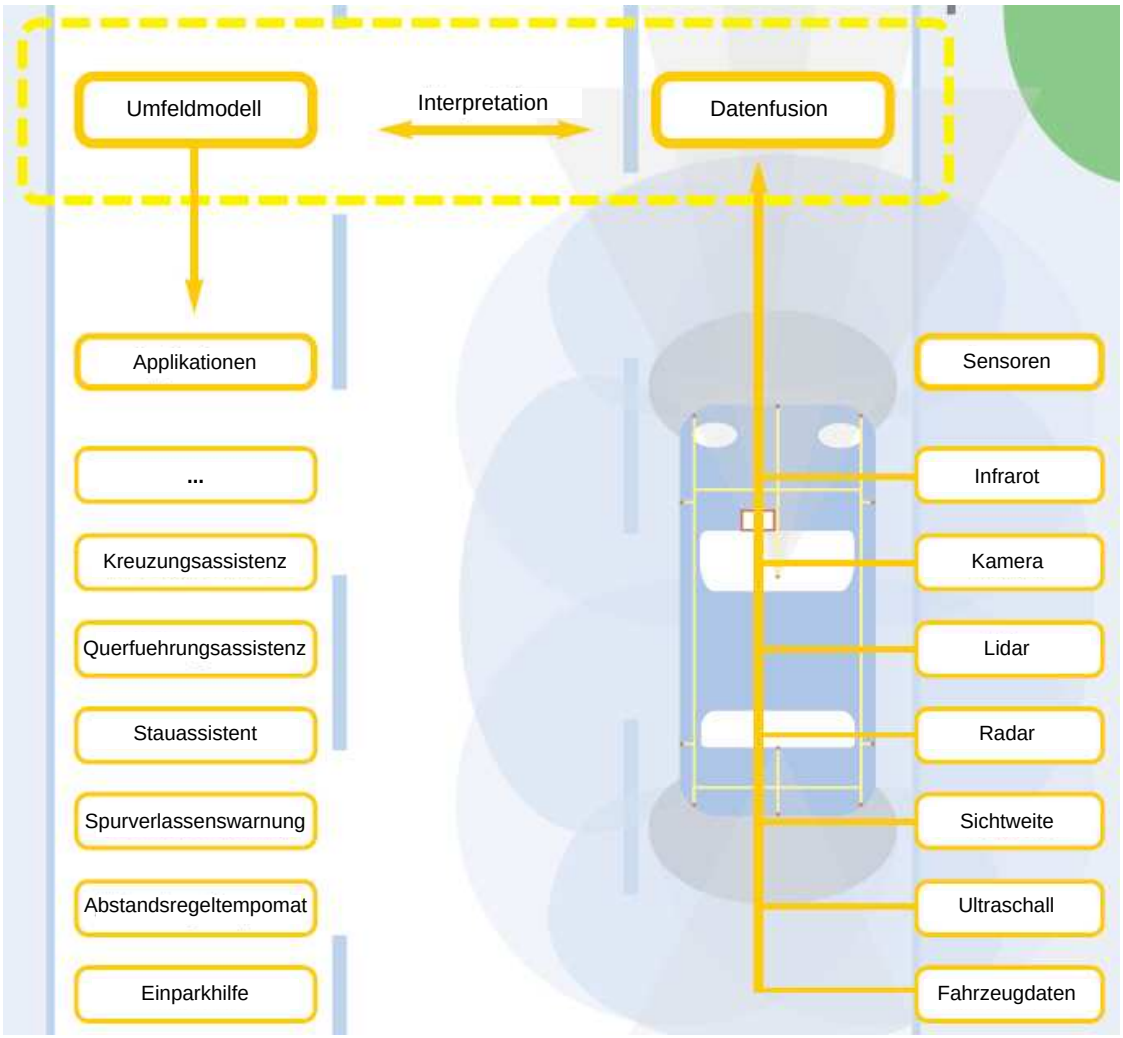


Abbildung 11.8: Autonomes Fahren in der Zukunft: Vernetzte Komponenten [1]

Literaturverzeichnis

[1] Invent: Fahrumgebungserfassung und Interpretation.
http://www.invent-online.de/downloads/FUE_handout-D.pdf : 18.10.2003.

[2] Bernd, T.: Autonomes Fahren.
http://www.uni-koblenz.de/~agrt/lehre/ss2003/seminar/thomas_bernd.pdf :
13.10.2003.

[3] Knoll, P. M.: Das „sensitive“ Auto. In: Auto&Elektronik (2003), Nr. 2/3, S. 10-14.

[4] Graas, B.: Präzision und Robustheit von Sensoren und Sensorfusionsmethoden.
http://resy.informatik.uni-kl.de/teaching/seminar.trends_in_der_robotik/
ausarbeitungen/13.Bernard_Graas_Sensorfusionsmethoden.pdf : 27.10.2003.

[5] N. N.: ohne Titel.
http://www.ghibsy.de/deutsch/02_dokumente/fertigungstechnik/Robotic.pdf :
17.10.2003.

[6] Lages, U. ; Fürstenberg, K. ; Willhoeft, V.: Nahfeldüberwachung rund um das Fahrzeug: Der Laserscanner uns seine Möglichkeiten zur Entlastung des Fahrers.
http://www.ibeo-as.de/html/public/files/
IBEO_AS_VDI_SAE_JSAE_2001_BERLIN.pdf : 25.10.2003.

[7] Hella KG Hueck & Co.: Technische Information : Elektronik - Fahrerassistenz-Systeme.
http://www.hella.com/produktion/HellaPortal/WebSite/InternetImages/
automotiv_08_2003/fahrer_assistenz.pdf : 22.10.2003.

[8] ALASCA: http://www.alasca.info : Oktober 2003.

[9] IBEO Automobile Sensor GmbH: http://www.ibeo-as.de : Oktober 2003.

[10] Tasikas, P.: Grundprinzip des Radars.
http://www.msw.ch/pdf-files/radar.pdf : 30.10.2003.

[11] Fuerstenberg, K. C. ; Dietmayer, K. C. J.: Laserscanner Innovations for Detection of Obstacles and Road.
http://www.mrm.e-technik.uni-ulm.de/homepage/fuerstenberg/
030307_AMAA_Final_A4.pdf : 30.10.2003.

[12] Schraut, M.: Umgebungserfassung auf Basis lernender digitaler Karten zur vorausschauenden Konditionierung von Fahrassistenzsystemen.
http://tumb1.biblio.tu-muenchen.de/publ/diss/ei/2000/schraut.pdf : 05.11.2003.

[13] Mende, R. ; Zander, A.: A Multifunctional Automotive Short Range Radar System.
http://www.smartmicro.de/
GRS_2000_Multifunctional_Short_Range_Radar_System.pdf :
12.11.2003.

[14] N., N.: Safety at your fingertips.
http://www.mts-web.de/download/77GHzAutomotiveRadarbrochure.pdf :
16.11.2003.

[15] Rieder, A.: Fahrzeuge sehen : Multisensorielle Fahrzeugerkennung in einem verteilten Rechnersystem für autonome Fahrzeuge. Dissertation : 2000.

Kapitel 12

Fahrerassistenzsysteme - Pre-Crash Systeme

Autor: Alexander Buchner

Betreuer: Dipl.-Ing. Benito Liccardi

12.1 Was sind Pre-Crash Systeme?

Pre-Crash Systeme bilden eine Untergruppe der Fahrerassistenzsysteme und zielen vor allem darauf ab, den Fahrzeuginsassen größt möglichen Schutz zu bieten. Es werden verschieden Fahrzeugbereiche und Fahrzeugeinrichtungen optimiert, so dass bei einer Kollision ein möglichst geringes Verletzungsrisiko auftritt.

> **Definition von Pre-Crash:**
> Vorgang der unmittelbar nach Starten des Autos Informationen über Fahrzeuge, Fußgänger und andere Hindernisse erfasst, mit diesen eine Crash-Wahrscheinlichkeit abschätzt und bei kritischen Situationen selbstständig reagiert.

12.1.1 Informationen zur Auslöseentscheidung

Mit Hilfe von verschiedenen Sensoren, wie Radar, Lidar, Infrarot, Videokameras oder Ultraschall werden Informationen über die Umgebung gesammelt. Durch die anschließend durchzuführende Datenfusion erhält man ein genaues Abbild der Fahrzeugumgebung. Somit können Hindernisse erkannt und als PKW, LKW oder Personen klassifiziert werden. Außerdem können Parameter wie Kollisionsgeschwindigkeit, Kollisionswinkel, Crash-Überdeckung (Offset) und die Zeit bis zur Kollision (time to collision, TTC) bestimmt werden. Aus dieser Fülle von Daten wird die Kollisionswahrscheinlichkeit errechnet. Sollte diese Kollisionswahrscheinlichkeit einen kritischen Wert annehmen, dann wird das System die entsprechenden Aktuatoren aktivieren.
Beispielsweise reduziert das ACC (Adaptive Cruise Control) bei zu geringem Abstand zum voraus fahrenden Fahrzeug automatisch die Geschwindigkeit.

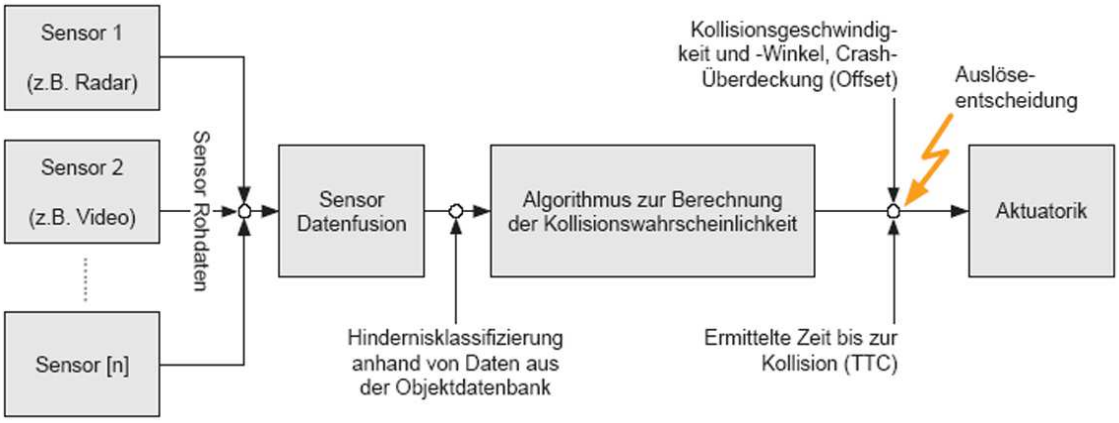


Abbildung 12.1: Strategie zur Ermittlung der Unfallwahrscheinlichkeit

12.1.2 Zeitliche Einordnung von Pre-Crash

In Bild 12.2 ist der zeitliche Verlauf einer Kollision dargestellt.
Mit den einzelnen Phasen: Fahrt, Warnung, Korrektur, Pre-Crash, Crash und Post-Crash Reagiert aus irgendwelchen Gründen der Fahrer nicht, oder lässt sich eine kritische Situation nicht mehr entschärfen und steht eine Kollision unausweichlich bevor, dann spricht man von der ''Pre-Crash" Phase. Diese beginnt mit dem Überschreiten des ''Point of No Return" (PNR), der bei komplexen Verkehrssituationen jedoch äußerst schwierig zu ermitteln ist und im Normalfall nur einige hundert Millisekunden vor dem physikalischen Kontakt liegt. Im folgenden werden exemplarisch einige aktive (Kapitel 12.2) wie auch

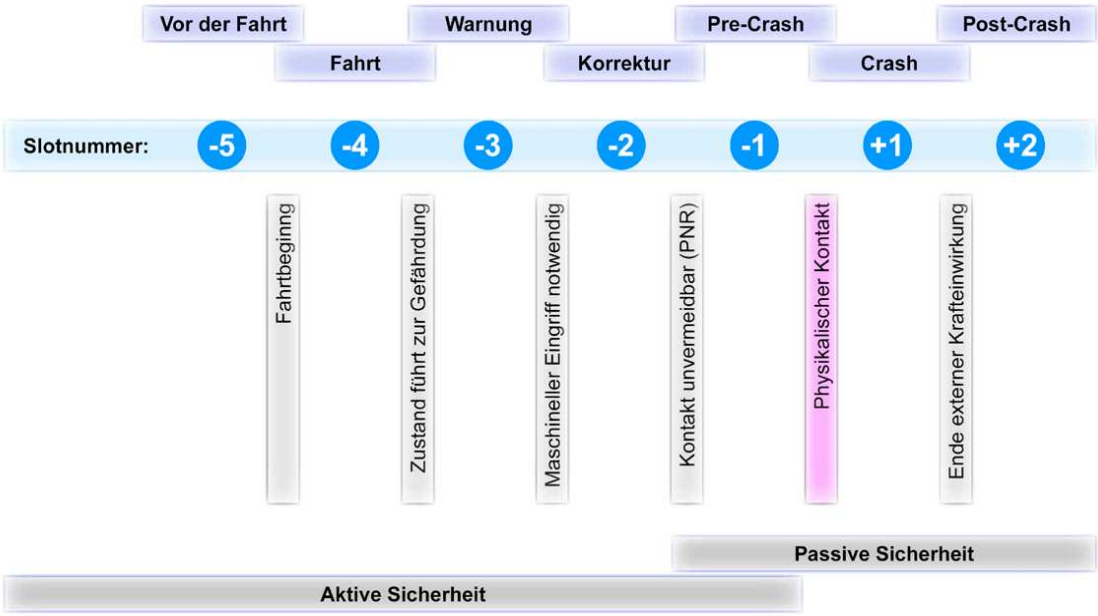


Abbildung 12.2: Zeitliche Einordnung im Fahrverlauf

passive (Kapitel 12.3) Sicherheitssysteme erläutert.
Unter **aktiven Sicherheitssystemen** versteht man Systeme, die bereits vor einer Kollision tätige werden. Sie sind so konzipiert, dass sie Unfälle vermeiden können bzw. in der Pre-Crash Phase die Kollisionsenergie vermindern.
Passive Sicherheitssysteme hingegen reagieren auf eine Kollision. Sie sollen die Insassen bzw. den Unfallgegner vor (schweren) Verletzungen verschonen.

12.2 Aktive Sicherheitssysteme

Die in der Pre-Crash Phase zur Verfügung stehende Zeit erscheint auf den ersten Blick sehr kurz, im Vergleich mit bestehenden Systemen der passiven Sicherheit handelt es sich jedoch um eine vergleichsweise lange Zeitspanne. Auslösezeiten von pyrotechnischen Systemen, wie Airbags und Gurtstrammern, liegen mit 5-50 Millisekunden um ein bis zwei

Größenordnungen darunter. In der durch Pre-Crash zur Verfügung stehenden Zeit vor dem Crash (TTC, Time To Collision) können auch nicht-pyrotechnische und somit meist reversible Systeme eingesetzt werden, deren Auslösezeiten über den oben genannten 5-50 Millisekunden liegen. Reversibel bedeutet in diesem Zusammenhang, dass das System nach einer eventuellen ''Fehlauslösung" ohne äußere Einflüsse selbständig in den Ursprungszustand zurückversetzt werden kann. Solche Systeme sind z.B. elektromagnetische Gurtaufroller (EMA) oder federgespannte Motorhaubenaktuatoren für den Fußgängerschutz. Deren Auslösezeit liegt im Bereich von 100 - 200 ms. Die Fähigkeit der Reversibiliät von Schutzsystemen wird bei der Entwicklung von BMW-Fahrzeugen als Voraussetzung zum Einsatz in der Pre-Crash Phase gesehen, da die von den Algorithmen ermittelten Kollisionswahrscheinlichkeiten anhand der aktuell und in naher Zukunft verfügbaren Sensordaten nur in wenigen Fällen über 95-99% liegen. Die daher über einen längeren Einsatzzeitraum zwangsläufig auftretenden Fehlauslösungen lassen ein Zünden von nicht reversiblen Rückhalte- bzw. Schutzsystemen vor der Kollision mit dem heutigen Entwicklungsstand nicht realisierbar erscheinen.

12.2.1 Autonomer Lenkeingriff

Aktive Font Steering (AFS) bezeichnet eine Vorrichtung zum selbständigem, aktiven Eingriff in das Lenkverhalten des Fahrzeugs. In Bild 12.3 ist die Realisierung über ein Planetengetriebe zwischen Lenkrad und Zahnstange dargestellt, in dass das AFS über einen Stellmotor in die Lenkung mit eingreifen kann. Somit bleibt ein noch gesetzlich vorgeschriebene mechanische Verbindung zwischen Lenkrad und Vorderachse erhalten. Zu den

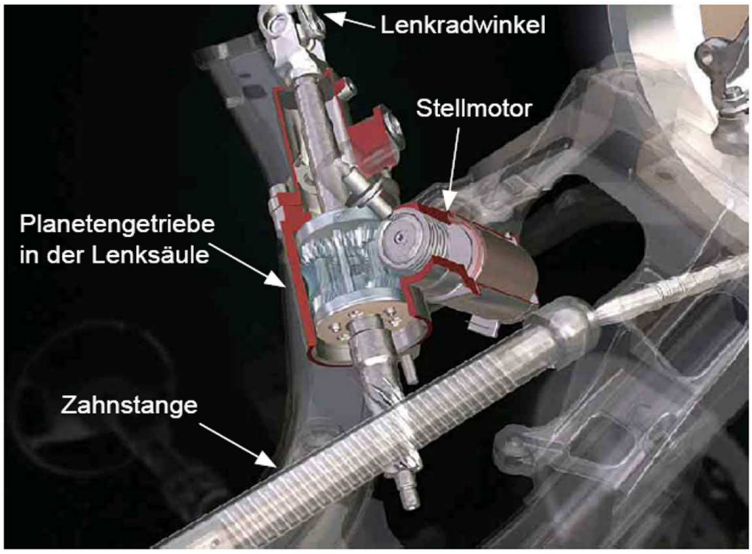


Abbildung 12.3: Zeitliche Einordnung im Fahrverlauf

Aktionen des AFS sind selbstständige Ausweichmanöver bei drohender Kollision denkbar. Jedoch aufgrund der meist sehr komplexen Verkehrssituationen wird die Bestimmung ei-

nes unfallvermeidenden, alternativen Fahrweges durch den Boadrechner anhand der Fahrzeugumfeldinformationen auch in näherer Zukunft technisch nicht zuverlässig realisierbar sein. Zu groß ist das Risiko, bei einem vom Fahrzeug bestimmten Ausweichmanöver einen höheren Schaden als bei der ursprünglichen Route zu verursachen.
Mit weniger Risiko behaftet sind Eingriffe in den Lenkwinkel bei Bremsungen mit unterschiedlichen Reibwerten an den Rädern. Eine solche Situation tritt häufig im Winter auf, bei schnee- oder eisbedeckten Fahrbahnrändern mit niedrigem Reibwert und trockener Streckenmitte mit hohem Reibwert. Dadurch baut des Fahrzeug ein Giermoment zur Fahrbahnmitte hin auf. Das DSC (Dynamic Stability Control) korrigiert diesen Einfluss durch Wegnahme von Bremskraft auf der Fahrerseite, was jedoch den Bremsweg verlängert. Durch autonomes Gegenlenken ließe sich das ursprüngliche Giermoment durch ein Gegenmoment kompensieren und somit wertvoller Bremsweg gewinnen, da die Bremskraft nicht so stark reduziert werden muss. Mit einer Verkürzung des Anhalteweges um bis zu 10 % aus 100 km/h zielt dieses System auf eine Reduzierung der Unfallenergie (Abbildung 12.4).

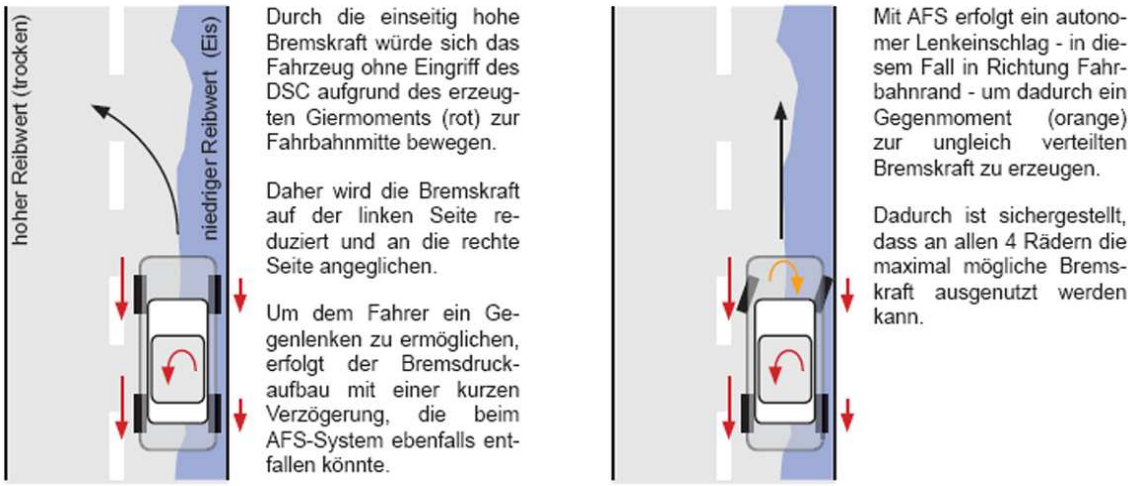


Abbildung 12.4: Zeitliche Einordnung im Fahrverlauf

12.2.2 Bremsassistent mit Vorspannung des Bremssystems

Der bereits in Serie befindliche, herkömmliche Bremsassistent erkennt eine Notbremsung des Fahrers am abrupten zurückgenommenen Gasbefehl und der sehr kurz darauf folgenden schnellen Betätigung des Bremspedals. In dieser Situation wird über eine Hydraulik-Pumpe die maximale Verzögerung aufgebaut, um eine Verlängerung des Anhalteweges durch zu zaghaftes Bremsen vieler Fahrer zu vermindern.
Bei Erkennung einer drohenden Kollision wird durch eine ''Vorspannung" der Bremshydraulik eine weitere Bremswegverkürzung erzielt. Hierbei wird der Druck im System moderat erhöht, um den Luftspalt zwischen Bremsscheiben und Bremsbelägen zu eliminieren.

Bei nassen bzw. winterlichen Straßenverhältnissen wird gleichzeitig die Bremsscheibe getrocknet und eine eventuelle vorhandene Streusalzschicht auf dem Reibflächen entfernt. Durch die damit kürzere Ansprechzeit ist eine Verbesserung des Bremsweges um bis zu 4 Meter aus 100km/h erreichbar. Die frühzeitige Bremswirkung vor dem ersten Kontakt mit dem Bremspedal führt aufgrund ihrer geringen Intensität kaum zur Ablenkung des Fahrers. Die warnende Wirkung dieser Maßnahme hat eher positiven Einfluss auf dessen Aufmerksamkeit.
Eine autonome Verzögerung mit höherem oder gar maximalem Bremsdruck bereits vor dem ''point of no return" (PNR) ist aufgrund einer möglichen Irritation des Fahrers kaum realisierbar. Ist die Kollision nach dem PNR jedoch nicht mehr vermeidbar, so kann diese automatische Vollbremsung dazu beitragen, die Kollisionsenergie nochmals zu minimieren. In den maximal 500 ms der Pre-Crash Phase lassen sich - ausgehenden von einer konstanten Verzögerung mit 9 m^2 - bis zu 16 km/h Geschwindigkeitsreduzierung erreichen.

12.3 Passive Sicherheitssysteme

12.3.1 Optimierung der Insassenposition

Ein großer Einflussfaktor bei der Auslösung von Rückhaltesystemen ist die korrekte Position der Insassen. Zu locker sitzende Gurte, zu geringe Entfernung vom sich entfaltenden Airbag (out of position), oder eine zu flache Sitzlehnen- bzw. Sitzflächenneigung können sich negativ auf die Insassenbelastung auswirken.
Reversible Systeme, wie elektrische Sitz- und Lehnenverstellung und elektromagnetische Gurtaufroller (EMA) können bereits in der Pre-Crash Phase aktiv werden und ein ''Untertauchen" des Sicherheitsgurtes (Submarining-Effekt) verhindern. Somit können die Insassen bereits zu Beginn der Kollision optimal an die Verzögerung des Fahrzeugs gekoppelt werden.
Die mit herkömmlichen 12 Volt betriebenen EMA-Systeme erlauben Gurtkräfte an der Wickellwelle von 200 bis 300 Newton und erreichen eine Ansprechzeit von ca. 150 ms. Diese Krafteinwirkung vor der Kollision genügt, die durch dicke Winterkleidung oder zu locker sitzende Gurte verursachte gurtlose Zeit zu minimieren. Zudem lässt sich damit die bei einer Vollbremsung auftretende Vorverlagerung der Insassen begrenzen, wodurch dies optimal an der Fahrzeugverzögerung teilnehmen und vom Airbag zurückgehalten werden können.
Die nach einer Kollision zündenden pyrotechnischen Gurtstrammer mit über 2000 Newton können mit dieser Technik jedoch nicht ersetzt werden, da die erforderliche Leistung beim 12 Volt Boardnetz zu hohe Ströme und somit zu große Kabelquerschnitte erfordern würde. Erst die leistungsfähige 42 Volt Technik verspricht eine entsprechend belastbare Stromversorgung im Fahrzeug. Damit wären unter anderem auch stärkere Sitzverstellungsmotoren zur Längs-, Höhen-, und Lehnenverstellung in der Pre-Crash Phase denkbar - ähnlich dem in 4-6 Sekunden agierenden DaimlerChrysler PreSafe-System - lediglich mit praxisnäher-

en Ansprechzeiten im Bereich der Pre-Crash Dauer. Momentan eingesetzte, elektrische Sitzverstellungen arbeiten auch bei PreSafe nur mit einer Geschwindigkeit von maximal 20 mm pro Sekunde.

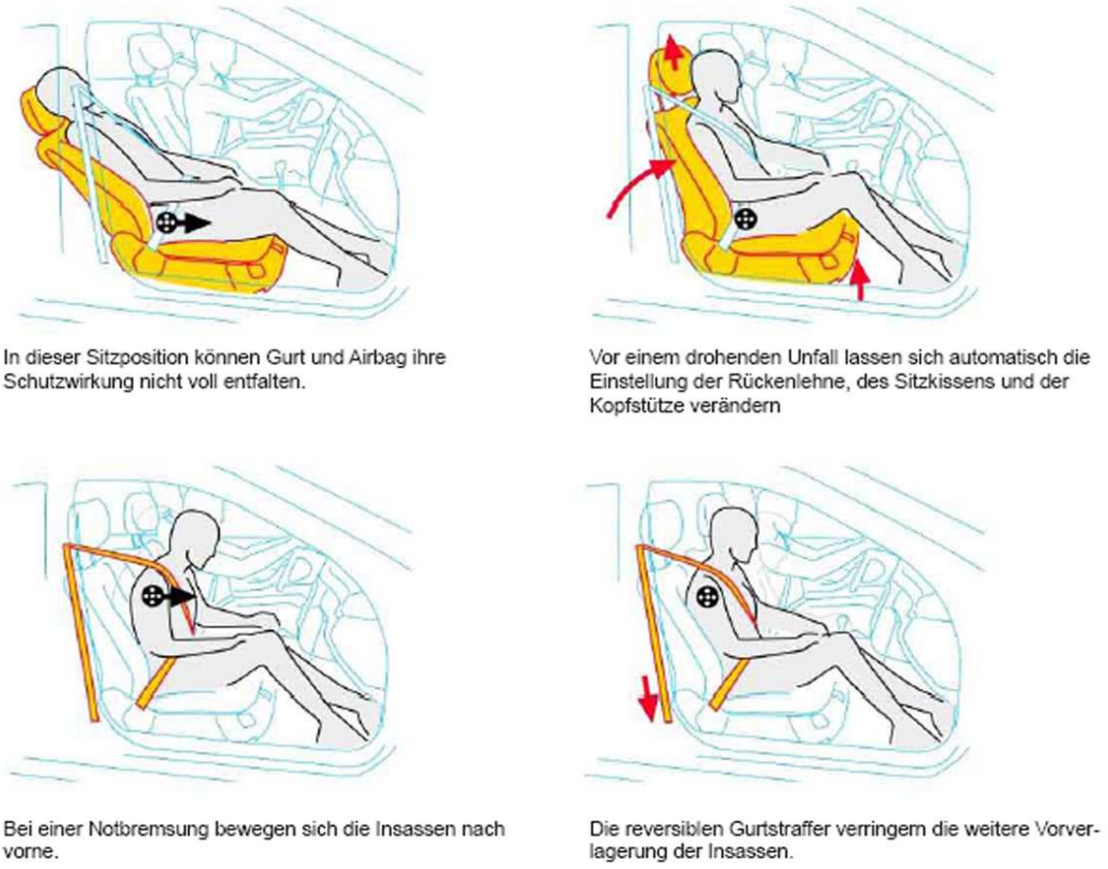


Abbildung 12.5: Reversible Gurtstrammer und Sitzverstellung vor dem Unfall

12.3.2 Schutzmechanismen bei Fußgängerunfällen

Fußgänger sind die schwächsten Verkehrsteilnehmer und können durch den Kontakt mit dem Fahrzeug schwer verletzt werden. Um mögliche Verletzungen zu minimieren und die in wenigen Jahren in Kraft tretenden Vorschriften zum Fußgängerschutz zu erfüllen, sind spezielle Schutzmechanismen erforderlich. Neben dem erweiterten Bremsassistenten, der durch die Reduzierung der Bewegungsenergie bei allen Unfalltypen vorteilhaft zum Einsatz kommen kann und den adaptiven Stoßfängern, existieren folgende weitere Maßnahmen zum Fußgängerschutz.

Motorhaubenaktuatorik

Die Motorhaube wird bei diesen Systemen aktiv, d.h. mit spezieller Aktuatorik um ca. 60-100 Millimeter angehoben, um den Deformationsweg für den Fußgänger zu erhöhen. Durch den gewonnenen Raum zwischen Motorhaube und harten Motorteilen oder Federdomen lässt sich die Verletzungsschwere teils deutlich reduzieren. Der von BMW favorisierte Aktuator beruht auf der potenziellen Energie vorgespannter Federn, die in der Lage sind, die Motorhaube innerhalb von 60 ms anzuheben. Das System ist nach einer eventuellen Fehlauslösung lediglich teilreversibel, da die zur Aktivierung benötigte Zündvorrichtung in der Werkstatt ausgetauscht werden muss. Bis zu dieser Reaktivierung ist der Schutzsmechanismus funktionslos. Bild 12.6 zeigt einen Auslösemechanismus,

Abbildung 12.6: Motorhaubenaktuator zur Anhebung der Motorhaube

der sowohl die Kraftaufnehmer in der Stoßstange als auch Daten der Umfeldsensorik mit einbezieht. Ist ein Hindernis als Fußgänger klassifiziert und tritt unmittelbar darauf eine typische Krafteinleitung in den Stoßfängern auf, so kann mit hoher Wahrscheinlichkeit von einer Fußgänger-Kollision ausgegangen werden. Durch diese Sensordatenfusion kann eine robustere Auslösung der Schutzmechanismen erfolgen.

Externe Airbags

An der Fahrzeugfront und im Bereich der Windschutzscheibe angebrachte, externe Airbags dienen der optimalen Verzögerung des getroffenen Fußgängers und können die Verletzungsschwere durch den Kontakt am Fahrzeug stark reduzieren. Jedoch sind aus Gründen der Realisierbarkeit (Langzeithaltbarkeit, Einbauraum) und der Kosten einer vollflächigen Abdeckung der Anprallflächen dem Einsatz solcher Systeme enge Grenzen gesetzt.

Bild 12.7 zeigt ein solches System, das lediglich den unteren Bereich der Windschutzscheibe und der Scheibenwischer abdeckt. Der besonders für große Personen verletzungsträchtig, weil hochstabile Windschutzscheibenrahmen wird - vor allem im oberen Bereich - nicht geschützt.

Abbildung 12.7: Externer Airbag zur Verminderung der Fußgängerverletzungen

Verbesserung der Auslösegenauigkeit

Bisherige Prototypen der aktiven FGS-Systeme (Fußgängerschutz-Systeme) verwenden lediglich Kraftaufnehmer in den Stoßstangen, um Schutzvorrichtungen anzusteuern. Eine Unterscheidung zwischen einem Begrenzungspfosten oder einem Fußgänger ist jedoch mit dieser Methode kaum möglich. Durch die teils nicht reversiblen Systeme könnten somit bei Bagatell-Unfällen unnötig hohe Reparaturkosten entstehen. Mit Hilfe weiterer verschiedener Sensoren ist es jedoch möglich, in der Pre-Crash Phase eine Fußgänger-Kollision zuverlässiger zu erkennen, und nur in diesem Falle entsprechende Schutzsysteme zu aktivieren.
Durch eine Sensordatenfusion der Umfeldsensoren mit Kraftaufnehmern in der Stoßstange des Fahrzeugs ist eine robustere bzw. zuverlässigere Auslösung von aktiven Motorhauben, adaptiven Stoßfängern und gegebenfalls externen Airbags möglich.
Als Umfeldsensorik eignen sich hierzu aufgrund der IR-Erkennung und der hohen Unfallhäufigkeit bei Fußgängern-Unfällen in Dunkelheit vor allem ein NightVision-System. Aber auch Lidar und Video-Systeme eignen sich als zusätzliche Sensorik, wobei deren Erkennungsleistung im Gegensatz zu NightVision stark von der Umgebungshelligkeit und der Reflektivität der Fußgängerkleidung abhängt.

12.4 Potential der Pre-Crash Technik bei Realunfällen

In diesem Kapitel wird versucht, das Potenzial der Pre-Crash Sensorik bei den häufigsten Realunfällen näher zu beleuchten. Wichtige Leistungsmerkmale eines Sensorsystems sind hierbei sowohl die sichere Detektion der Relativgeschwindigkeit, als auch eine robuste Klassifizierung eines Hindernisses im Fahrzeugumfeld. Allein die Vorhersage, dass eine Kollision mit einem nicht näher beschriebenen Objekt bevorsteht reicht nicht aus, um spezifische Schutzmaßnahmen einzuleiten. Erst die Information, ob es sich dabei um einen Fußgänger, Motorradfahrer, PKW oder einen LKW handelt, ermöglicht eine zuverlässigere Prognose der Crash-Schwere, wie sie zur Auslösung der in den vorigen Kapiteln vorgestellten Schutzmechanismen unbedingt erforderlich ist.
Die meisten verfügbaren Sensoren sind ursprünglich entweder nur für die Detektion oder nur für die Klassifizierung von Objekten - z.B. in der Produktionstechnik - entworfen worden und werden nun im Zuge einer verbesserten automotive-Tauglichkeit für beide Anwendungsfelder optimiert, was nicht immer auf Anhieb für zufrieden stellende Resultate sorgt. Es ist daher naheliegend, auch eine Fusion der Daten zweier unterschiedlicher Sensortypen in Betracht zu ziehen. Besonders interessant sind die Kombinationen aus Radar/Video sowie Lidar/Video, da hierbei die gute Objektdetektion von Radar bzw. Lidar mit der guten Objektklassifizierung der Videosysteme verbunden wird.
Aufgrund der größtenteils erst im Entwicklungsstadium befindlichen Technik, sowohl bei den Sensoren, als auch den damit anzusteuernden Schutzmechanismen, ist eine exakte Angabe einer Erkennungswahrscheinlichkeit oder der Verminderung der Verletzungsschwere noch nicht möglich.
Besondere Aufmerksamkeit gilt dabei dem Fußgängerschutz, da die Gesetzgebung auf diesem Gebiet bereits konkrete Formen annimmt. Anstatt aufwändige, konstruktive Maßnahmen am Fahrzeug selbst vorzunehmen, bietet sich hier die Pre-Crash Technnik an, die Unfallzahlen und das Verletzungsrisiko deutlich zu senken.

Unfallzahlen

Fußgänger sind die im Straßenverkehr am geringsten geschützte Personengruppe, da sie über keinerlei Schutzvorrichtungen zur Energieabsorption verfügt, wie dies bei Fahrzeuginsassen der Fall ist. Zudem gibt es mangels Durchführbarkeit keine gesetzlichen Vorschriften zur verbesserten Sichtbarkeit, wie z.B: helle und reflektierende Kleidung. 1999 wurden in Deutschland bei knapp 38.000 Verkehrsunfällen mit Fußgängerbeteiligung über 800 Personen getötet. Dies entspricht einem Rückgang von 15% der Unfälle und knapp 50% der Getöteten seit 1991, der unter anderem auf die Verbesserung der Infrastruktur, die bessere Ausbildung, die Optimierung des Rettungswesens und auch auf Geschwindigkeitsbegrenzungen zurückzuführen ist. Aber auch die konstruktiven Veränderungen an modernen Fahrzeugen der letzten Jahre, wie beispielsweise runde, aerodynamisch gestaltete Frontstrukturen trugen ihren Teil zu dieser Verbesserung bei.
Angesichts dieses bereits positiven Trends hat sich die Europäische Kommission das Ziel gesetzt, die Zahl der getöteten Fußgänger bis zum Jahr 2010 im Vergleich zu 1999 um

weitere 30% zu reduzieren (Abbildung 12.8). Erreicht werden soll dieses Ergebnis durch

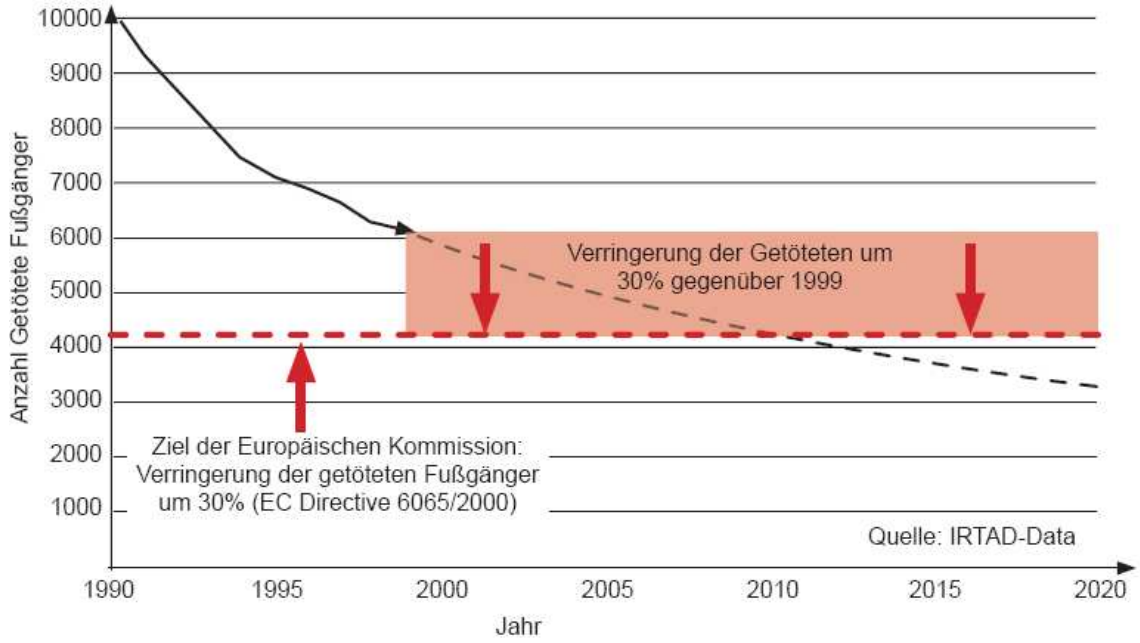


Abbildung 12.8: Ziel der Europäischen Kommisssion, die Fußgängerunfälle um 30% zu vermindern

zusätzliche Maßnahmen im Bereich der Fußgängersicherheit bei PKWs, da im Jahr 2000 knapp 70% der Fußgänger bei Kollisionen mit dieser Fahrzeugkategorie getötet wurden. Der Verband europäischer Automobilhersteller (Association des Constructeurs Européens d'Automobiles, ACEA) schlug in einer freiwilligen Selbstverpflichtung im Juni 2001 ein Maßnahmenpaket für PKW mit einem zulässigen Gesamtgewicht unter 2,5 Tonnen mit folgendem Inhalt vor (Abbildung 12.9). Phase 2 beeinhaltet noch keine endgültigen Testvorschriften. Bis spätestens Juli 2004 kann ACEA daher einen alternativen Maßnahmenkatalog vorschlagen, der den Möglichkeiten der Fahrzeugkonstruktion entgegenkommt, jedoch ein mindestens gleichwertiges Schutzpotenzial für Fußgänger offeriert, wie dies vom European Enhanced Vehicle-safety Committee, EEVC [http://www.eevc.org] gefordert wird. Diese, 1970 gegründete, staatliche Organisation besteht aus Repräsentanten verschiedener europäischer Nationen mit dem Ziel, die Grundlagenforschung auf verschiedenen Gebieten der Fahrzeugsicherheit voranzutreiben. Die Working Group 17 (WG17) des EEVC beschäftigt sich mit dem Fußgängerschutz. Die gewonnenen Erkenntnisse dienen als Basis für zukünftige Richtlinien bzw. Gesetzgebung und besitzen selbst keinen verbindlichen Charakter.

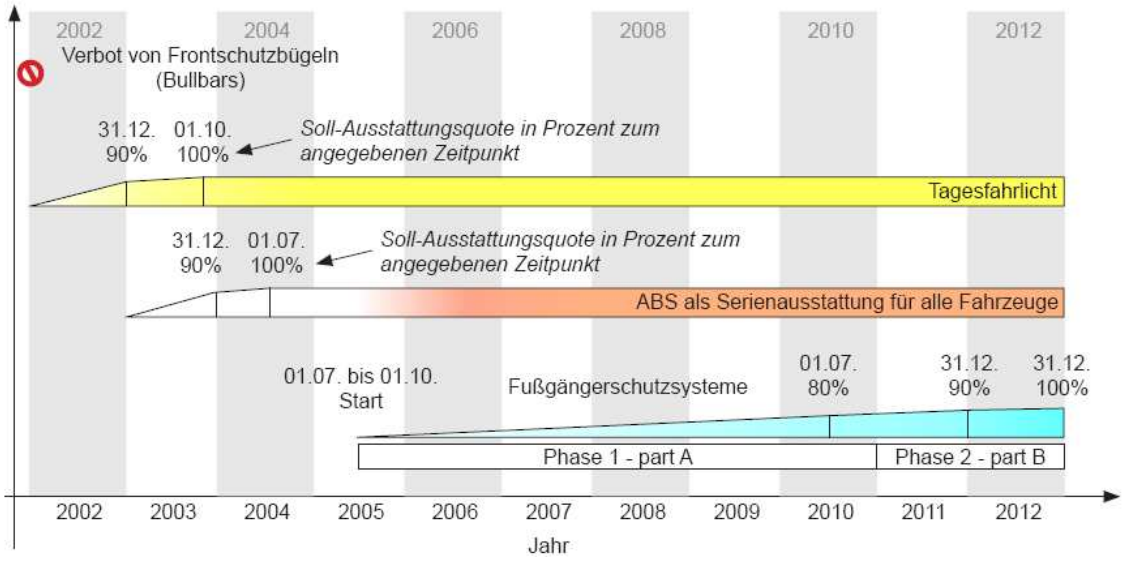


Abbildung 12.9: Zeitplan zur Einführung der Fußgängerschutzmaßnahmen

Literaturverzeichnis

[1] Sven Alvater. *Diplomarbeit über Konzeptentwicklung alternativer Seitenschutzsysteme.* München, 2002.

[2] BMW. *www.bmw.de.*

[3] ContinentalTemic. *www.temic.de.*

[4] DaimlerChrysler. *www.daimlerchrysler.de.*

[5] Werner Dr. Foag. *Potenzial for Pedestrian Protection via Autonomus Brake.* München, 2003.

[6] Thomas Goernig. *Weiterentwicklung Insassenschutz - Pre-Crash Sensorik - Sensorik zum Fußgängerschutz.* München, 2002.

[7] Thomas Prof. Langwieder, Klaus; Kramlich. *Charakteristik von Fußgängerunfällen.* München, 2001.

[8] SiemensVDO. *www.siemensvdo.de.*