1 Einleitung

1.1 Vorwort

In den letzten Jahren sind verschiedene Studien mit dem Ziel durchgeführt worden, die Validität des neuen Ansatzes hinsichtlich einer automatisiert durchführbaren Überwachung und einer entsprechenden Therapieführung von Herzpatienten unter Verwendung von modernen Telekommunikationstechnologien zu ermitteln.

Die Ergebnisse dieser Studien haben gezeigt, dass die automatische Überwachung von Herzpatienten bzw. Patienten mit Herzunterstützungssystemen technisch möglich ist. Überdies wurde deutlich, dass auf die Mitwirkung des Patienten nicht verzichtet werden kann. Der Patient erhält aber dadurch die Möglichkeit, mehr Informationen über seinen Krankheitszustand und Therapieprozess zu bekommen, was wieder zu einem besseren Verständnis und zu einer höheren Zufriedenheit beim Patienten führen kann. Aus einem ähnlichen Gesichtpunkt heraus entstand die Idee meiner Arbeit, da die Überwachung durch die regelmäßige Abfrage und Auswertung der Herzunterstützungssysteme zu einer verbesserten Diagnose, Betreuung und Therapieführung von Patienten führt.

Der Ausgangpunkt dieser Diplomarbeit war der Gedanke, im klinischen Einsatz befindliche Herzunterstützungssysteme der Firma Berlin Heart AG [1] über eine längere Entfernung hinweg online zu überwachen und damit einen höheren Standard an Komfort und zugleich funktionaler Sicherheit und Zuverlässigkeit gewährleisten zu können. Durch diese Überwachungstechnologie würden Zeit und Kosten des technischen Supportes minimiert werden.

Diese Diplomarbeit beinhaltet die Konzeption und die Entwicklung dieser Überwachungs-technologie. Vor allem handelt es sich um die Hardware- und Softwareentwicklung eines Telematik-Moduls, das die Grundlage für die Realisierung dieser Technologie bildet. Es werden die Themen Telematik und Telemedizin behandelt sowie die Frage, in welcher Weise Telekommunikation, Informatik und Multimedia das Gesundheitswesen beeinflussen. Die rechtlichen Rahmenbedingungen telemedizinischer Anwendungen und ihre Anforderungen auf die Datensicherheit werden bearbeitet. Diese Diplomarbeit behandelt vor allem alle wichtigen Grundlagen der Internet-Netzwerke und TCP/IP, die zum besseren Verständnis und zur Nachvollziehbarkeit der behandelten Thematik führt. Die Arbeit erläutert dann die Arbeitsweise der Lösungselemente. Die Kryptographie bleibt nicht unerwähnt, sowohl in meinen Grundlagen-Abschnitten als auch bei der Realisierung von vor Fremdeinwirkungen sicheren Übertragungswegen.

Meine Diplomarbeit von der Idee bis zur Entwicklung und Realisierung war eine Zusammenarbeit zwischen den Firmen Berlin Heart AG und TeCNet GMBH [2]. An dieser Stelle möchte ich mich bei beiden Firmen für ihre Unterstützung und Betreuung sehr herzlich bedanken.

1.2 Definitionen, Akronyme, Abkürzungen

AES – Advanced Encryption Standard
ARP- Adress Resolution Protocol
API- Application Programming Interface
DHCP- Dynamic Host Configuration Protocol
FTP- File Transfer Protocol

ICMP- Internet Control Message Protocol
IEEE- Institute for Electric and Electronic Engineers
IP- Internet Protocol

ISDN- Integrated Services Digital Network
ISO- International Organisation for Standardization
ISP – Internet Service Provider

GPRS – General Packet Radio Service
HTTP- Hyper Text Transfer Protocol
LAN- Local Network Area
LCD – Liquid Crystal Display
NIST- National Institute of Standards and Technology
OSI- Open System Interconnection
PPP- Point to Point Protocol

RFC- Request For Comments
RJ11- Physikalischer Anschluss eines analogen Modems
RJ45- Physikalischer Anschluss für TCP/IP-Verbindungen
SCL- Serial Clock
SDA- Serial Data Clock
SMTP- Simple Mail Transfer Protocol
SMS- Short Message Service
TCP- Transmission Control Protocol
UDP- User Datagram Protocol
VAD- Ventricular Assist Device
WAN- Wide Area Net
WHO- World Health Organisation

1.3 Zielstellung

Ziel dieser Arbeit ist die Entwicklung einer Technologie zur permanenten Überwachung des diagnostischen Befundes von Patienten mit Herzunterstützungssystemen über das Internet, sowie der durchgehenden Dokumentation der diagnostischen Daten. Der Inhalt und die Ergebnisse dieser Arbeit sollen die Grundlage für die Entwicklung einer telemedizinischen Fernüberwachungstechnologie bilden, um eine Diagnose und Überwachung der Patienten zu gewährleisten.

Im Rahmen der Diplomarbeit soll sowohl die Netzwerkstruktur einer zertifizierungsfähigen Technologie gebildet werden als auch die Software und Hardware-Entwicklung eines Telematik-Moduls. Das Modul ermöglicht Patienten mit Herzunterstützungssystemen von ihren behandelnden Ärzten bzw. von den Clinical Affairs (siehe unten) online zu überwachen. Dadurch wird die ärztliche Betreuung der Herz-Patienten mit solchen Systemen unabhängig von ihrem Aufenthaltsort möglich. Durch diese Überwachungstechnologie würden auch Zeit und Kosten des technischen Supportes minimiert werden.

Die Clinical Affairs ist eine Abteilung in einer Firma. Sie stellt das Bindeglied zwischen der Firma und den Kunden/Anwendern dar, wenn es sich um technischen- und sachlichen Support handelt. Die Aufgaben beziehen sich auf die Systeme der Firma Berlin Heart AG. Folgende Punkte zählen zu den Aufgabengebieten der Clinical Affairs:

- Produkttraining
- Medizinischer Support
- Technischer Support
- Wissenschaftliches Marketing
- Studien
- Kontaktpflege zum klinischen Anwender
- 24h-Hotline

Bei der Planung der zu entwickelnden Technologie entstanden mehrere Anforderungen, die sowohl die Verfügbarkeit von Infrastrukturen und Verbindungselementen umfassen als auch die Anforderungen an die Zertifizierung erfüllen.
Die Anforderungen an die zu entwickelnde Technologie und das Telematik-Modul sind folgende:

1. Das Telematik-Modul ist ein Mikroprozessorsystem. Das System soll es ermöglichen, Daten aus der Steuereinheit des Herzunterstützungssystems des Patienten zu empfangen und diese an verschiedene Systeme weiterzuleiten.

2. Die Schnittstellen für die Kommunikation mit anderen Systemen sollten gemäß Abbildung 1.1 in das Telematik-Modul integriert werden. Folgende Schnittstellen sind für das System nötig:

    - Analoges Modem
    - Ethernet
    - RS232– serielle Schnittstelle
    - Bluetooth
    - GPRS – General Packet Radio Service

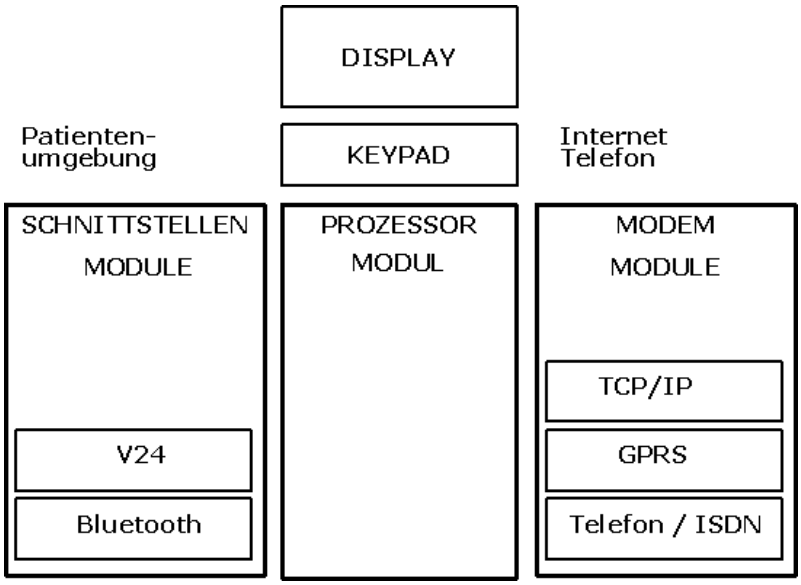


Abb. 1.1 Blockschaltbild der Schnittstellen des Telematik-Moduls

3. Die gesicherte Übertragung von aktuellen Daten (Online Monitoring) zu einem PC/Monitor bei Clinical Affairs.
4. Die gesicherte Übertragung von Daten (Log-Daten) zu und von einem zentralen File-Server.
5. Die Realisierung eines klinischen File-Servers zur Archivierung aller Log-Daten jedes in Verkehr gebrachten Systems.
6. Realisierung eines File-Servers (Klinischer Server) zur Archivierung aller Log-Daten (Log Files) jedes in Verkehr gebrachten Controllers.
7. Der klinische File-Server soll Linux als Betriebssystem verwenden.
8. Die Datenabfrage und die Verbindung zum klinischen Server muss als patientenbewusste Handlung durchgeführt werden, deshalb werden Schalter und Display für die Bedienung und die Anzeige benötigt.
9. Das Telematik-Modul soll Schalter beinhalten, um die Steuerung der wichtigsten Funktionen zu ermöglichen.
10. Das Telematik-Modul soll eine konfigurierbare Echtzeituhr haben, mit der aktuelle Datums- und Zeitanzeige angezeigt werden.

11. Mit einem Grafik-LCD-Modul, soll dem Patient folgendes zur Verfügung gestellt werden:

    - Der Patient startet sein Telematik-Modul
    - Der Patient wählt aus einem kleinen Menü "Controller auslesen" oder "Laptop auslesen"
    - Daten werden ausgelesen
    - Daten auslesen OK.- weiter mit Taste
    - Daten zur Klinik übertragen? " (J/N) "
    - Dann muss die Art der Übertragung ausgewählt werden (via Internet / Telefon?)
    - Anzeige "Verbindung mit Klinik hergestellt"
    - Abfrage "Daten übertragen? (J/N) "
    - Anzeige "Daten werden übertragen"
    - Anzeige "Datenübertragung OK! "
    - Anzeige "Verbindung wird getrennt"

12. Es müssen mehrere Verbindungen von und zum Server parallel möglich sein.
13. Bei der Softwareentwicklung vom Mikroprozessorsystem sollen mehrere Softwaremodule (Application Programming Interface) implementiert werden, um die Kommunikation zwischen der Patientenumgebung, Internet und der Clinical Affairs zu realisieren.
14. Die Datensicherheit muss durch die Verwendung eines Verschlüsselungsverfahrens gewährleistet sein.
15. Es muss aus Zertifizierungsgründen eine Kommunikationsmethode bzw. Kommunikationsregel mit dem klinischen Server entwickelt werden, die die Übereinstimmung zwischen den übertragenen Daten aus dem Herzunterstützungssystem mit den Daten, die auf dem Server ankommen, vergleicht.
16. Das Telematik-Modul soll über einen Anschluss zu einem Laptop/PC verfügen. Über diesen Anschluss soll das System konfigurierbar sein und in der Lage Daten (Log-Daten) aus dem Laptop/PC laden. Zur Konfiguration wird eine Konsole verwendet. Die Konsole ist vorerst auf Windows-Systemen zu implementieren. Es sollen folgende Parameter konfigurierbar sein:

    - Die Internetverbindungen entweder via Modem wie ISP-Einstellungen (Internet Service Provider Einstellungen) oder via TCP/IP-Einstellungen.
    - Datum und Uhrzeit
    - ID (Identifikationscode) des Systems (7-stellig)

1.4 Arbeitsmaterial

Für die Software- und Hardware-Implementierung dieser Arbeit wurden verschiedene Materialien und Tools benutzt. Anbei werden stichpunktartig alle verwendeten Materialien aufgelistet. Für die detaillierte Beschreibung dieser Tools wird auf die Literatur und Internet-Quellen verwiesen.

1. Zwei Rechner mit Windows 2000 als Betriebssystem dienten für Analyse, Testen sowie der Hardware- und Softwareentwicklung des Telematik-Moduls.
2. Ein Rechner mit Linux Suse 9.1 diente für die Server-Programmierung (Clinical Server).
3. RCM3200-Entwicklungsboard von Rabbit Semiconductor [49].
4. Dynamic C Version 8.51 für die Programmierung des Rabbit-Kern-Moduls von Z-World [44].
5. Kdeveloper Software für die Socket Programmierung am Linux (Klinischer Server)
6. Turbo C ++ Version 2.0 von BORLAND [57] für die Programmierung einer DOS-basierten Konsole
7. PCAD für das Schaltungsdesign des zu entwickelnden Telematik-Moduls [56].

2 Fachliches Umfeld

2.1 TCP/IP-Grundlagen

2.1.1 Kommunikationsprotokolle

Um Herzunterstützungssysteme über das Internet überwachen zu können bzw. einen Internet-Knoten realisieren zu können, bedarf es zunächst einigen Grundwissens über die Standard-Datenübertragungsprotokolle. Im Folgenden werden die einzelnen für diese Arbeit wichtigen Protokolle vorgestellt.

Moderne Kommunikationssysteme bestehen aus einer Sammlung von Protokollen (Protokollstack bzw. Protokollstapel). Jedes diese Protokolle enthält seine eigenen Funktionen, verwendet aber zur Bereitstellung seines Dienstes auch die Funktionen des darunter liegenden Protokolls. Das für das TCP/IP-Satck geltende, aus vier Schichten bestehende Modell ist eine Teilmenge des ISO/OSI-7-Schichtenmodells. In Abbildung 2.1 sind die beiden Referenzmodelle gegenübergestellt [4].

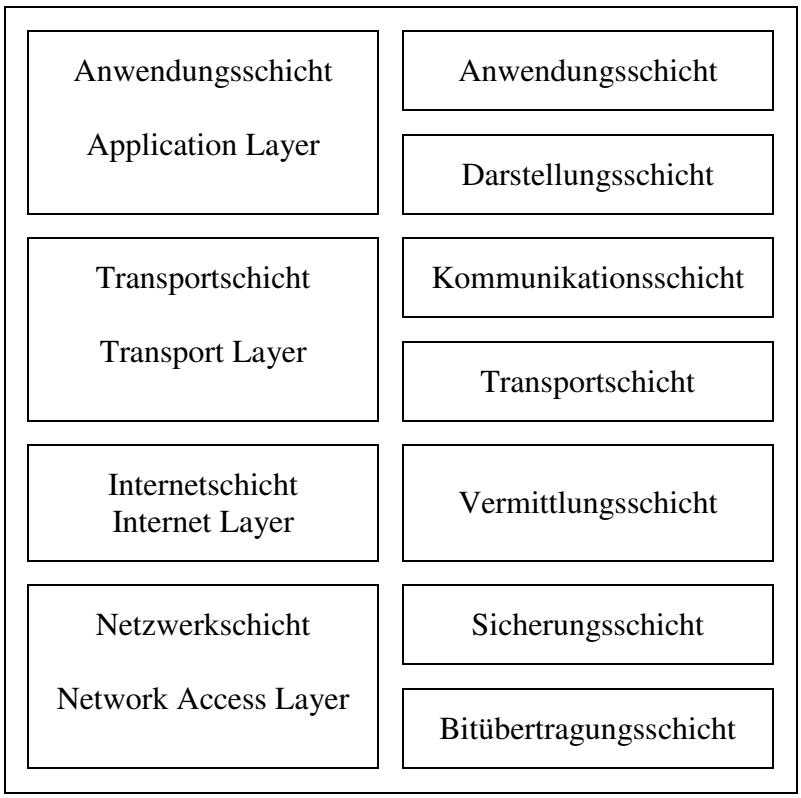


Abb. 2.1 ISO/OSI und Internet-Referenz-Modell

Die Funktion der einzelnen Schichten ist wie folgt:

1. Application Layer: Enthält eine Vielzahl von Protokollen, die Anwendungen zur Erbringung ihrer Dienste definiert haben „HTTP, FTP und SMTP “

2. Transport Layer: Ermöglicht Kommunikation zwischen Quell- und Zielhost (Kommunikation zweier Punkte) „TCP, UDP“

3. Internet Layer: Zustellung und Routing von Datagrammen zwischen den Internet-Netzkonten „ IP, ICMP, ARP“

4. Network Layer: Hostspezifische Implementierung der Übertragung von Datagrammen „ Ethernet (IEEE 802.3), PPP“

Für die weiteren Ausführungen in dieser Arbeit werden die Begriffe wie TCP/IP-Stack, Client/Server-Verhalten, IP-Adresse, Port-Nummer, Sockets, DHCP Client bzw. Server, eine Peer to Peer Verbindung, Ethernet und PPP benutzt. Aufgrund besseren Verständnisses der Arbeit werden diese Begriffe kurz erläutert. Da die Original-Spezifikationen teilweise mehrere hundert Seiten lang sind, sollte klar sein, dass diese Information nur einen Überblick darstellen kann. Für eine detaillierte Beschreibung wird auf entsprechende Quellen und Literatur verwiesen.

2.1.2 Der TCP/IP-Protokollstack

Der TCP/IP-Stack besteht aus drei Basisprotokollen (UDP, TCP und IP) in der Transport-und Netzwerksschicht, einigen Hilfsprotokollen in der Netzwerkschicht (ARP) und vielfältigen Applikationsprotokollen (den so genannten TCP/IP-Anwendungen im Applikations-Layer).

Von der Anwendung ausgehend, welche Daten man übertragen möchte (z.B. ein Webserver), wird durch jede Schicht des TCP/IP-Stacks ein entsprechender Protokollkopf (Header) mit Kontrollinformation vorangestellt (Abbildung 2.2). Dieser Vorgang wird Datenkapselung genannt (encapsulation). Beim Empfangen eines solchen Frames (z.B. aus Ethernet bzw. PPP) sind vom TCP/IP-Protokollstack dann von den einzelnen Softwareschichten die entsprechenden Header auszuwerten und zu entfernen, um letztlich die eigentlichen Nutzdaten zu gewinnen und an die entsprechende Applikation (z.B. Internet-Browser) weiter zu geben. D.h. ein TCP/IP-Stack muss seine Funktionalität anderen Programmen zur Verfügung stellen. Dafür ist eine Programmierschnittstelle (API) erforderlich. Die meisten Microcontroller und Prozessoren auf dem Markt bieten den TCP/IP-Stack in Programmbibliotheken mit Funktionen für den Zugriff auf TCP und UDP an.

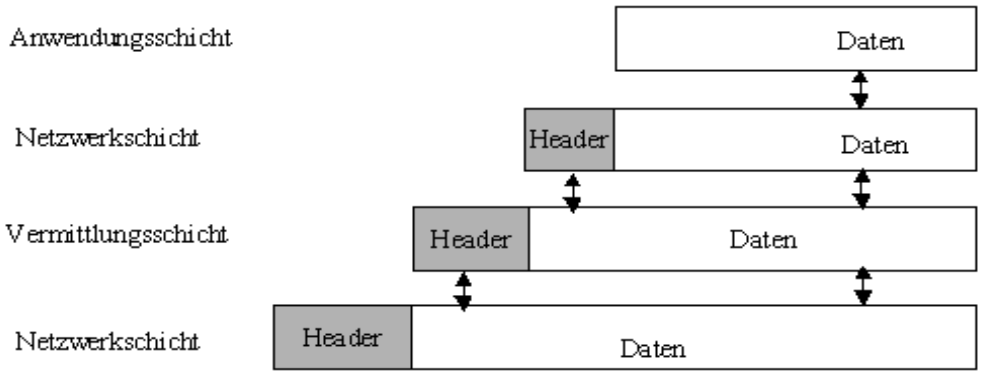


Abb. 2.2 Daten-Kapselung im Internet

Eine wichtige Frage bei der Kommunikation zwischen zwei Knoten im Internet-Netzwerk lautet:
Wer beginnt mit der Kommunikation?
Neben vorgegebenen Regeln durch die Protokolle nutzen TCP/IP-Protokolle ein Client/Server-Prinzip als Basis für die Kommunikation im Internet-Netzwerk.
Das Prinzip besagt, dass der Server immer eine passive Rolle einnimmt. Er wartet auf den Verbindungsaufbau und die Anforderung einer Dienstleistung (Service) durch einen Client. Der Client ist gegenüber dem Server immer der aktive Teil im Netzwerk. Er überträgt eine Anforderung (Request) an den Server und leitet damit den Verbindungsaufbau ein. Über diesen Request teilt er dem Server mit, dass er eine bestimmte Dienstleistung in Anspruch nehmen möchte. Der Server bietet normalerweise unterschiedliche Dienstleistungen an. Er antwortet dem Request vom Client mit einem Response.

Nachdem das Prinzip für die Kommunikation zwischen zwei Knoten im Internet-Netzwerk dargestellt wurde, sind die IP-Adressen dieser Knoten von großer Bedeutung.
Eine IP-Adresse ist eine logische Adresse und vergleichbar mit einer Telefonnummer im Telefonnetz. Über eine IP-Adresse (zum Beispiel 192.168.1.60) wird ein Knoten bzw. ein Rechner im TCP/IP-Netzwerk adressiert. Momentan ist die Verwendung der IP Version 4 (IPv4) üblich. Die IP-Adresse besteht aus 4 Zahlen (1 Byte groß, zwischen 0 und 255), durch Punkte getrennt [5].

Über Port-Nummern werden die einzelne TCP/IP-Anwendungen angesprochen.
Port-Nummern und IP-Adressen charakterisieren ein Socketinterface. Sockets sind Kommunikationsendpunkte, die aus IP-Adresse und Port-Nummer bestehen und dienen der bidirektionalen Kommunikation zwischen zwei Programmen, die nach dem Client/Server-Prinzip organisiert sind.

Eine IP-Adresse adressiert einen Rechner, die Port-Nummer eine bestimmte Anwendung. Es kann aber mehrere Anwendungen zum gleichen Zeitpunkt geben.
Jede Anwendung benötigt eine Port-Nummer. Port-Nummern sind Zahlen zwischen 0 und 65,535. Viele Port-Nummern sind für bekannte Netzwerksdienste bereits reserviert. Port-Nummern aus diesem Bereich wie die 21 für FTP (File Transfer Protocol) werden als privilegierte Ports bezeichnet.

Die Adressierung eines bestimmten Knoten im Internet-Netzwerk kann statisch oder dynamisch erfolgen. Die statische Adressierung eines Knoten im Netzwerk wird mit einem Konfigurationsprogramm eingestellt. Problematisch bei diesen Verfahren ist die Verwaltung der Knoten im Netzwerk. Für die dynamische Variante bieten TCP/IP-Protokolle Verfahren an, um einem Knoten in der Bootphase eine IP-Adresse, die erforderliche Subnet Maske sowie die IP-Adresse des jeweiligen Default Gateways automatisch zuzuweisen. DHCP (Dynamic Host Configuration Protocol) ist ein Verfahren für die dynamische Adressierung. Das Prinzip ist einfach. Ein Knoten im Netzwerk muss eine statische festgelegte IP-Adresse besitzen. Dieser Knoten wird zum DHCP-Server erklärt. Diesem wird von einem Administrator ein bestimmter Bereich von IP-Adressen (zum Beispiel 192.168.0.1 bis 192.168.0.100) zugewiesen. Der Server vergibt jeweils eine IP-Adresse als Antwort auf Anforderungen der angeschlossenen Knoten im Netzwerk. Ein Knoten ohne eigene IP-Adresse bildet den DHCP-Client.

2.1.3 Übertragungsmedien

Für die Entwicklung der Überwachungstechnologie und im Rahmen meiner Diplomarbeit wurden zwei Übertragungsstrecken vorgesehen. Die Datenübertragung wird sowohl über Ethernet, als auch über PPP bzw. die Modem-Einwahlverbindung erfolgen.

Ethernet ist ein herstellerneutrales und sehr weit verbreitetes Medium, um im lokalen Netzwerk (Local Area Network, LAN) und im WAN (Wide Area Net) Datenpakete zu übertragen. Es stellt im Internet-Referenzmodell ein Protokoll der Netzwerkschicht dar (Network Layer). Der Standard IEEE 802.3 (Institut for Electric and Electronic Engineers) definiert mögliche Übertragungsmedien, die physikalische Umsetzung der Informationen und die eigentliche Datenübertragung mit Geschwindigkeiten von 10Mbits/s bzw. 1Gbits/s (Fast Ethernet).
Ethernet basiert auf der gemeinsamen Nutzung eines Kanals, auf den alle angeschlossenen Netzwerkteilnehmer Zugriff haben. Dazu wird ein statisches Kanalzugriffsverfahren benutzt (CSMA/CD-Carrier Sense Multiple Acces with Collosion Detection). Die Übertragung der Bits erfolgt Manchester-codiert über übliche differentielle Zweidrahtleitungen (10Base-T, Twisted Pair, RJ45) [4].

Jedem Netzwerkknoten ist eine eindeutige, 48 Bits lange physikalische Hardwareadresse, die MAC-Adresse (Media Acces Control), zugeordnet. Die Länge eines mittels Ethernet übertragbaren Frames liegt zwischen 64 Byte und 1518 Byte. Diese Größenangabe bezieht sich auf den gesamten Rahmeninhalt, mit Ausnahme der Präambel. Ist die Datenmenge größer als 1500 Byte, so werden die Daten voneinander getrennt, und in 1500 Byte-Blöcken übertragen. Nach dem Senden eines Paketes erfolgt eine Pause von 9,6 µs. Diese Pause wird als Inter Frame GAP bezeichnet. Zur Verdeutlichung ist in Abbildung 2.3 ein Rahmen beschrieben [IEEE802.3].

Präambel	Zieladresse	Quelladresse	Typfeld	Datenfeld	Prüffeld
8 Byte	6 Byte	6 Byte	2 Byte	46 -1500 Byte	4 Byte

Abb. 2.3 Ethernet-Rahmen

Die Präambel besteht aus identischen Bytes (Binär: 10101010) und dient der Synchronisation des eingehenden Rahmens mit dem Zeitgeber des Empfängers. Direkt im Anschluss daran folgt der eigentliche Frame. Die ersten sechs Bytes (48 Bit) bilden die Zieladresse im LAN (auch als DA= Destination Address bezeichnet). Die folgenden zwei Byte geben den Typ der Frames an [5].

Diese Ethernet-Typnummern sind eindeutig und in der IEEE-Norm festgelegt. Sie dienen zur Unterscheidung, an welche höheren Protokolle der Rahmen weiterzuleiten ist. Der Abschnitt „LLC data“ enthält im Anschluss daran die zu übertragenden Nutzdaten (pay-load) der höheren Protokollschichten. Am Ende eines Ethernet-Rahmen wird eine vier Byte lange CRC-Prüfsumme übertragen, um den Rahmen auf Fehler oder Beschädigung prüfen zu können. Die maximale Anzahl der in einem Datenpaket übertragbare Nutzdaten-Bytes ergibt sich demnach aus 1518-6-6-2-4=1500. Da ein Ethernet Frame nicht kürzer als 64 Byte sein darf, werden eventuell fehlende Bytes durch „ Padding“ mit beliebigen Bytes aufgefüllt.

Das Point-to-Point Protocol (PPP) als zweite Übertragungsstrecke arbeitet gegenüber Ethernet auf der Schicht 2 des OSI-Schichtenmodells und wurde für die Übertragung von Schicht-3-Protokollen über eine Punkt-zu-Punkt-Verbindung entwickelt (siehe Abbildung 2.4). Punkt-zu-Punkt-Verbindungen sind z. B. Wählverbindungen über das analoge Telefonnetz.

Das PPP-Protokoll sieht eine Methode vor, die Datenpakete verschiedener anderer Protokolle einzukapseln und über eine physikalische Verbindung zu übertragen. Man spricht in diesem Zusammenhang auch von Tunneling.

Die Verbindung wird mit dem Link Control Protocol (LCP) aufgebaut, konfiguriert, getestet und auch wieder abgebaut. Außerdem enthält das PPP-Protokoll mehrere Network Control Protocols (NCPs), z. B. IPCP, die Schicht-3-Protokolle über eine Punkt-zu-Punkt-Verbindung aufbauen und konfigurieren können [7].

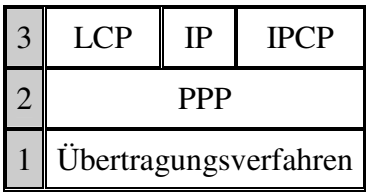


Abb. 2.4 PPP im OSI-Schichtenmodell

2.2 Kryptographie

Das Wichtigste für den Einsatz der Überwachungstechnologie ist, dass sie kein erhöhtes Risiko darstellt. Die Daten sind besonders sensibel und unterliegen daher besonderem Schutz. Werden diese Daten durch ein weltweit offenes Netz geschleust, so muss gewährleistet werden, dass sie nicht jeder sehen kann.
Kryptographie kommt aus dem Griechischen und heißt übersetzt „Geheimschrift“. Kryptographie ist gleichbedeutend mit der Verschlüsselung digitaler Nachrichten. Verschlüsselung kann auf verschiedenen Arten geschehen. Wichtigstes Unterscheidungsmerkmal zwischen einzelnen Verschlüsselungsverfahren ist die Anzahl der verwendeten Schlüssel [30].

Durch den Einsatz der Verschlüsselungsverfahren kann sichergestellt werden, dass die Daten nicht mitgelesen oder modifiziert werden können. Unter Verschlüsselung versteht man einen Prozess, bei dem aus einer offenen Nachricht durch einen speziellen Algorithmus unter Verwendung eines Schlüssels eine verschlüsselte Nachricht erstellt wird, die keine Rückschlüsse auf den Inhalt der offenen Nachricht zulässt.

Verschlüsselungsverfahren lassen sich in so genannte symmetrische und asymmetrische einteilen. Ein symmetrischer Verschlüsselungsalgorithmus verwendet im Gegensatz zu einem asymmetrischen denselben Schlüssel zur Chiffrierung und Dechiffrierung einer Nachricht [31].

Bei der Entwicklung der Überwachungstechnologie wurde sich für eine symmetrische Verschlüsselung der Daten entschieden, da man bei der Entwicklung den Zugriff auf die Komponenten besitzt. Aus diesem Grund wird in diesem Abschnitt nur diese Art der Verschlüsselung mit dem verwendeten Algorithmus erläutert.

Symmetrische Verschlüsselungsalgorithmen sind oft Blockverschlüsselungsalgorithmen (auch Blockchiffre). Bevor sie Daten verschlüsseln, werden diese zunächst in Blöcke unterteilt, deren Größe durch den Algorithmus vorgegeben ist. Die Blöcke werden anschließend unabhängig von einander verschlüsselt. Die Verschlüsselungsroutine setzt jedoch ganze Blöcke voraus, somit muss meistens der letzte Block mit Zufallsdaten aufgefüllt werden. Wichtige Parameter einer Blockchiffre sind somit die Blocklänge (auch Blockgröße) und die Schlüssellänge. Der große Nachteil symmetrischer Verfahren liegt in der Nutzung ein und desselben Schlüssels zur Chiffrierung und Dechiffrierung. Ist der Schlüssel einem Angreifer bekannt, ist es für ihn leicht, an Information zu gelangen und Fehlinformationen durch Veränderung der Originalnachricht zu verbreiten.

Bei der Entwicklung wurde die AES (Advanced Encryption Standard) ausgewählt.
Der Advanced Encryption Standard (AES) ist ein symmetrischer Verschlüsselungsalgorithmus, welcher als Nachfolger für Data Encryption Standard (DES) bzw. 3DES im Oktober 2000 vom National Institute of Standards and Technology (NIST) als Standard bekannt gegeben wurde. Nach seinem Entwickler wird er auch Rijndael-Algorithmus genannt [28].

Die wichtigsten Eigenschaften von AES sind:

- AES ist ein symmetrischer Algorithmus, und zwar eine Blockchiffre.
- AES verwendet mindestens 128 Bit lange Blöcke und setzt Schlüssel von 128, 192 und 256 Bit Länge ein.
- AES kann allen bekannten Methoden der Kryptoanalyse (die Kunst, einen Geheimtext ohne Kenntnis des Schlüssels zu dechiffrieren) widerstehen, insbesondere Power- und Timing-Attacken.
- Jeder Klartextblock wird zur Verschlüsselung mehrere Male mit einer sich wiederholenden Abfolge verschiedener Funktionen behandelt, in so genannte Runden. Die Anzahl der Runden ist von der Block- und der Schlüssellänge abhängig.
- Schnellste Erzeugung von Rundenschlüsseln.
- Rijndael erlaubt durch seine Einfachheit (die Referenz-Implementierung umfasst weniger als 500 Zeilen C-Code) eine der schnellsten Implementierungen und zeichnet sich durch seine gleichmäßige gute Performance über alle betrachteten Plattformen aus, wie z.B. 32-Bit-Prozessoren, 8-Bit Mikrocontroller [29].

AES ist eine Blockchiffre, dessen Blocklänge und Schlüssellänge unabhängig voneinander die Werte 128, 192 oder 256 Bit erhalten kann. Jeder Block wird zunächst in eine zweidimensionale Tabelle mit vier Zeilen geschrieben, dessen Zellen ein Byte groß sind. Die Anzahl der Spalten variiert somit je nach Blockgröße von 4 (128 Bit) bis 8 (256 Bit). Jeder Block wird nun nacheinander bestimmten Transformationen unterzogen. Aber anstatt jeden Block einmal mit dem Schlüssel zu verschlüsseln, wendet AES verschiedene Teile des Schlüssels nacheinander auf den Klartext-Block an. Die Anzahl dieser Runden variiert und ist von Schlüssellänge und Blockgröße abhängig.

Bei der Softwareentwicklung von den Elementen der Überwachungstechnologie werden in verschiedenen Applikationen AES mit 14 Runden, 128 Bit Schlüssel, 256 Bit Blocklänge und im Cipher feedback (CFB)- Betriebsmodus verwendet. Der Quellcode der AES-Bibliothek wurde in einer optimierten C/C++ Implementierung von Brain gladman. Dieser Code wurde in die Programmierumgebung von Dynamic C angepasst (siehe Kapitel 8.2). Der Sourcecode

ist auf der Internet Seite http://fp.gladman.plus.com unter dem Titel "A Specification for Rijndael, the AES Algorithm" zu finden.

3 Telemedizin

3.1 Geschichte und Definition der Telemedizin

Die Telemedizin ist keine Entwicklung der letzten Jahre. Die NASA hat schon in den 60er Jahren mit den ersten bemannten Raumfähren begonnen, ihre Astronauten auch über weite Distanzen medizinisch zu überwachen. Wörtlich bedeutet Telemedizin lediglich Fernmedizin. Unter diesem Begriff versteht man jegliches Behandlungsverfahren, das als Element die unmittelbare Überwindung räumlicher Distanzen zwischen Patient und Arzt enthält [20].

Unter Telemedizin versteht man die Tatsache, medizinische Daten, also Texte, Tabellen, Befunde sowie Bilder, über große Entfernungen hinweg elektronisch auszutauschen bzw. zu versenden, um eine diagnostische oder therapeutische Interaktion zu ermöglichen. Die Definition legt besonderes Augenmerk auf die räumliche Trennung der Teilnehmer an Erbringung und Unterstützung von Gesundheitsleistungen (siehe Abbildung 3.1).

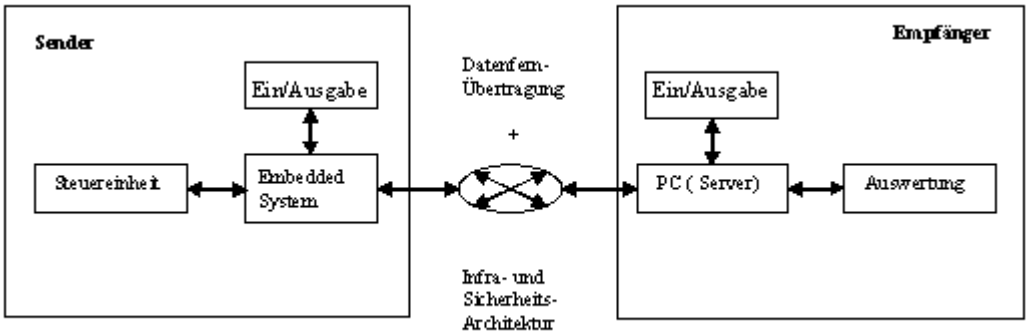


Abb. 3.1 Die technische Definition von Telemedizin mit einem DFÜ-Modul. DFÜ steht für die Datenfernübertragung.

Für die Definitionsreichweite ist es unwesentlich, ob die Übertragung online oder offline erfolgt, ob mit Interaktionsmöglichkeit oder ohne. Die Unterscheidung zwischen der Übertragung patientenbezogener und nichtpatientenbezogener Daten ist jedoch bei telemedizinischen Anwendungen wichtig. Für diese Daten gelten unterschiedliche Sicherheitsanforderungen.

Die sehr wichtige Unterscheidung zwischen patientenbezogenen Daten und nicht patientenbezogenen Daten ist von großer Bedeutung in dieser Arbeit, da die zu entwickelnde Überwachungstechnologie auf die Herzunterstützungssysteme des Patienten angewendet wird, indem die Parameter einer implantierten Herzpumpe überwacht werden. Von daher ist die Rede von nicht patientenbezogenen Daten, die zu einem zentralen Server übertragen werden sollen.
Die Bedeutung von Telematik bzw. Telemedizin in Bezug auf die Überwachung von Herzunterstützungssystemen wird deutlicher durch die Konzeption und die Ergebnisse dieser Arbeit.

Die Telemedizin ist einer der vier Bereiche der Telematik nach WHO (World Health Organisation)-Standard. Unter Telematik (Telematik= Telekommunikation und Informatik) versteht man einen Sammelbegriff für gesundheitsbezogene Aktivitäten, Dienste und Systeme, die über eine Entfernung hinweg mit Mitteln der Informations- und Kommunikations-Technologie ausgeführt werden, zum Zwecke globaler Gesundheitsförderung, Krankheits-Kontrolle und Krankenversorgung, sowie für Ausbildung, Management und Forschung für das Gesundheitswesen [WHO98]. Die restlichen drei Bereiche der Telematik sind das Gesundheitsmanagement (GM), Teleausbildung, Telematik für die Medizinische Forschung [11].
Nachfolgend sind weitere Begriffsverwendungen von Telematik oftmals auch in engerer Bedeutung:
Begriffsdefinition im weiteren Sinne: Teilgebiet der Informatik, das sich mit der Kommunikation von Daten unter Nutzung technischer Mittel über ("größere") räumliche Entfernungen befasst.
Meyers Lexikon: Kopplung von Informatik und Telekommunikation bzw. –wissenschaften.
Brockhaus: Forschungsbereich, der eine Synthese aus Telekommunikation und Informatik darstellt.
Telematik= Öffentliche Informationsdienste für den privaten und geschäftlichen Gebrauch, z.B. Verkehrstelematik = Telematik im Verkehrswesen (rechner- und kommunikationsgestützte Verkehrsleitsysteme) oder Telematik im Gesundheitswesen (z.B. Ferndiagnose, automatische Notrufsysteme mit Überwachung von Vitalfunktionen).

3.2 Die Anwendungsgebiete der Telemedizin

Obwohl die Telemedizin weitestgehend noch eine Zukunftsvision ist, zeichnen sich schon heute ihre konkreten Anwendungsgebiete ab. Diese sind entsprechend der weiten Definition der Telemedizin äußert vielseitig [12]. Die folgenden Anwendungsgebiete zählen aber zu den Wichtigsten:

- die medizinische Kommunikation (Teleoperation, Konsultation, Befundübertragung, das Elektronische Rezept),
- das Outsourcing von Patientendaten,
- die Gesundheitsnetze, insbesondere medizinische Informationssysteme und die Elektronische Patientenakte

Bei der medizinischen Kommunikation handelt es sich um die Übertragung von Patientendaten, um die Diagnose durch einen räumlich entfernten Spezialisten zu ermöglichen. Outsourcing bedeutet, dass zumindest Teilbereiche der Datenverarbeitung durch externe Unternehmen durchgeführt werden. Im Gesundheitswesen beinhaltet dies, dass Patientendaten nicht beim Arzt oder im Krankenhaus, sondern in einem externen Archive ausgelagert und archiviert werden.

Outsourcing kann auch in der Form möglich sein, dass digitalisierte verschlüsselte Patientendaten über Kommunikationsnetze zu einem Archiv geschickt werden. Je nach Bedarf kann der archivierende Arzt diese Online abrufen. Es wird im Prinzip dabei außerhalb der

ärztlichen Einrichtungen Rechenkapazität zur Verfügung gestellt und die Verwaltung der Patientendaten übernommen. Gesundheitsnetze stellen einen wesentlichen Baustein für den Aufbau einer informationstechnischen Gesundheitsplattform dar. Sie eröffnen viele Möglichkeiten zur Verbesserung der medizinischen Gesundheitsplattform und einer kosteneffizienten Behandlung.

Die Telemedizin ist deshalb daran zu messen, inwieweit sie in der Lage ist, einen Beitrag zur Verbesserung des Einsatzes der Ressourcen im Gesundheitswesen zu leisten, insbesondere hinsichtlich folgender Aspekte [15] [16]:

- Verbesserung der Qualität der Versorgung durch Standardisierung der medizinischen Dokumentation
- Verbesserung der Koordination der Behandlung innerhalb der ambulanten Versorgung und zwischen ambulanter und stationärer Versorgung durch Austausch von Informationen über Behandlungsdaten und Befunde,
- Bereitstellung von Daten über erbrachte Leistungen und deren Kosten für Entscheidungsträger des Gesundheitswesens,
- Erleichterung der Verfügbarkeit von medizinischem Wissen und Erfahrungen,
- Vereinfachung von Verwaltungs- und Abrechnungsabläufen

Die Ziele eines zukünftigen Gesundheitssystems für den Patienten sind sehr einfach zu postulieren, diese Ziele sind auch seit Jahren durch die Gesetzgeber anvisiert worden [19].

1. Der Patient trifft selbst alle Entscheidungen, die ihn und seinen Körper betreffen
2. Der Patient ist dafür ausgebildet und aufgeklärt (Patienten -Information)
3. Der Patient verfügt über seine Patientengeschichte (Informelle Selbstbestimmung)

Vielen Patienten, Medizinern, Pflegekräften, Sozialarbeitern etc. erscheint dies utopisch und nicht zu verwirklichen, weil, wie immer wieder zu hören ist, die Patienten überhaupt nicht in der Lage seien, den Sachverhalt zu verstehen und dann auch entsprechend zu entscheiden. Die Abbildung 3.2 zeigt im zukünftigen Gesundheitssystem das Wechselspiel und Gleichgewicht zwischen den individuellen Bedürfnissen einerseits und den Bedürfnissen der Gesellschaft mit den dafür erforderlichen (in den Ovalen stehenden) Voraussetzungen andererseits. Die Hauptvoraussetzung ist dabei vermutlich die Einsicht, dass Patienten, wenn sie ein gewisses Alter erreicht haben, für sich selbst verantwortlich sind. Dazu müssen sie auch die Einsicht bekommen, dass ihr Handeln in jedem Falle Auswirkung auf ihr Leben haben wird.

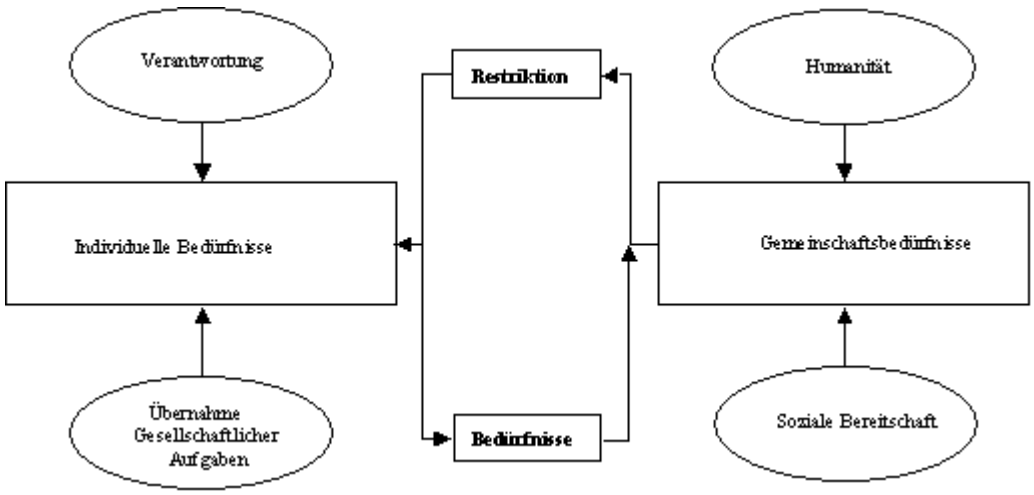


Abb. 3.2 Das Wechselspiel und Gleichgewicht im zukünftigen Gesundheitssystem zwischen den individuellen Bedürfnissen und den Bedürfnissen der Gesellschaft mit den dafür erforderlichen (in den Ovalen stehenden) Voraussetzungen.

Bei der täglichen gemeinsamen Arbeit an den Problemen des Patienten kommen die einzelnen Schritte des Arbeitsablaufs 'Vorschlag, Verstehen, Entscheiden und Handlung' überhaupt nicht zur Geltung, weil alle Beteiligten (mit Ausnahme des Patienten) es als selbstverständlich ansehen, dass der Patient unausgesprochen zustimmt, weil er ja sowieso keine Kenntnisse hat und schließlich einer Therapie in irgendeiner Form zustimmt.

3.3 Datensicherheit

Dass bei der derzeitigen Datenverarbeitung z. T. erhebliche Sicherheitsdefizite bestehen, zeigen u. a. die regelmäßigen Berichte in der Fachpresse und oft auch in anderen, nicht auf Fragen der Informationstechnik spezialisierten Medien.

Die Entwicklung einer Überwachungstechnologie für Patienten mit Herzunterstützungssystemen stellt eine hohe Anforderung an die Datensicherheit.
Der Sammelbegriff Datensicherheit umfasst im Zusammenhang mit der Überwachungstechnologie drei verschiedene und zugleich zusammenhängende Aspekte: Verfügbarkeit, Integrität, Vertraulichkeit [14]

a) Die **Verfügbarkeit** betrifft im Rahmen der Telemedizin vor allem die Stabilität und Ausfallsicherheit der Netze und der Serversysteme. Ohne Verfügbarkeit dieser Ressourcen rund um die Uhr ist der weitere Ausbau der Telemedizin nicht denkbar, insbesondere im Bereich der Kooperation. Patientendaten dürfen dagegen nur dort verfügbar sein, wo sie zur Behandlung des Patienten benötigt werden.

b) **Integrität** bedeutet Echtheit (Authentizität), Zuverlässigkeit und Fälschungssicherheit von Daten und Informationen, aber auch Fälschungssicherheit von Identitäten (Personen und Maschinen) bei der Kommunikation. Die Echtheit einer Personenidentität ist auch Grundlage für den Schutz des Zugangs zu vertraulichen Daten. Selbstverständlich müssen medizinische Daten vor Verfälschung geschützt werden.

c) Die **Vertraulichkeit** im Rahmen der Medizin soll die ärztliche Schweigepflicht und die Datenschutzanforderungen gewährleisten. Sie ist eine wesentliche Anforderung bei

Kommunikation und Kooperation in der Telemedizin. Jeder Beteiligte an einem medizinischen Betreuungsprozess soll nur die Daten sehen dürfen und können, die er im Interesse des Patienten sehen muss. Unbedingt verhindert werden muss, dass vertrauliche Daten bei der Übermittlung im Netz abgehört werden können, wie es heute in der Regel leicht möglich ist. Als Ausprägung der Vertraulichkeit kann auch die Anonymität gelten: Sie bedeutet die Vertraulichkeit einer Personenidentität. So würde die Krankenkassenabrechnung mit Hilfe von Pseudonymen verhindern, dass sich bei den Kassen umfangreiche Sammlungen personenbezogener Daten anhäufen.

Um die Datensicherheit gewährleisten zu können, müssen verschiedene Sicherungsinstrumente eingesetzt werden. Die wichtigsten technischen Sicherungsinstrumente sind die Kryptographie und das Firewall-System. Die Kryptographie wurde im letzten Kapitel behandelt. Firewall heißt übersetzt „Brandschutzwand“. Es handelt sich um eine Schutzwand im Netzwerk. Sie lässt, quasi wie eine Art bewachte Schranke, nur bestimmte Dienste durch. Firewall-Systeme mit einem Paketfilter schützen damit vor einem unbefugten Zugang zu einer Datenbank, auf die nur ein ausgewählter Personenkreis Zugriff haben soll.

3.4 Allgemeine datenschutzrechtliche Anforderungen

Der Einsatz eines Systems ist vor allem zwei Gefahren ausgesetzt. Es wird zum einen dann geschwächt, wenn die Nutzer der Technologien leichtfertig mit ihnen umgehen, z. B. die Schlüssel nicht sorgfältig verwahren oder die Geräte nicht vorschriftsmäßig warten. Eine zweite Gefahr liegt in der Zersetzung des Immunsystems von innen heraus, in dem immer wieder neue Techniken geschaffen werden, mit denen z. B. Signaturen nachgebildet werden, Verschlüsselungssysteme entschlüsselt und Schutzwälle durchlässig werden. Es bedarf hier der stetigen Überprüfung und Anpassung an den aktuellen, technischen Entwicklungsstand, um so resistent gegen derartige Angriffe zu bleiben.

Absolute Sicherheit gibt es nicht. Sorgfältig angewandt und überwacht bieten Kryptographien, digitale Signatur und Firewall-System einen gleichsam ausreichenden notwendigen Schutz. Die datenschutzrechtliche Anforderung an die Implementierungen und Telematikanwendungen solcher Art ist von großer Bedeutung, deshalb werden ein Überblick und eine Einführung dieser Problematik in diesem Kapitel kurz erläutert.

Für die Verarbeitung personenbezogener und nicht personenbezogener Patientendaten im Rahmen telemedizinischer Anwendungen gelten grundsätzlich die allgemeinen rechtlichen Rahmenbedingungen, die für die Verarbeitung personenbezogener Patientendaten außerhalb telemedizinischer Anwendungen gelten. Die Einführung telemedizinischer Anwendungen darf nicht zu einer rechtlichen oder faktischen Verschlechterung der Patientenrechte führen. Die Durchsetzung bzw. Konkretisierung der Patientenrechte unter den veränderten technischen Bedingungen bedarf teilweise neuer datenschutzrechtlicher Konzepte. Der Inhalt aller in diesem Abschnitt behandelten Thematik stammt zum größten Teil aus den Literatur-Quellen.

Rechtsgrundlagen

Für die Verarbeitung von Patientendaten durch niedergelassene Ärzte gelten die Vorschriften des BDSG (Bundesdatenschutzgesetz). Für die Verarbeitung von Patientendaten durch die Krankenhäuser gelten in Bund und Ländern unterschiedliche Rechtsvorschriften. In einzelnen

Ländern liegen sog. bereichsspezifische Regelungen der Verarbeitung personenbezogener Daten in Krankenhäusern (Landeskrankenhausgesetze, Gesundheitsdatenschutzgesetze etc.) vor. Soweit keine bereichsspezifischen Regelungen vorhanden sind, gelten die allgemeinen datenschutzrechtlichen Vorschriften [18].

Dokumentationspflicht

Nach der Berufsordnung ist der Arzt verpflichtet, die erforderlichen Aufzeichnungen über die in Ausübung seines Berufs gemachten Feststellungen und getroffenen Maßnahmen anzufertigen. Es handelt sich um eine unselbständige vertragliche Nebenpflicht aus dem Behandlungsvertrag. Ist die Dokumentation lückenhaft, kann dies im Haftungsprozess eine Umkehr der Beweislast zugunsten des Patienten nach sich ziehen, wenn die Aufklärung des Sachverhalts für den Patienten insgesamt erschwert wird.

Befugnis zur Übermittlung bzw. Weitergabe von Patientendaten

Nach den datenschutzrechtlichen Regelungen müssen Einwilligungen bestimmte Anforderungen erfüllen, um rechtswirksam zu sein. Insbesondere muss die Freiwilligkeit der Einwilligung gewährleistet sein und der Betroffene muss zuvor über Umfang und Zweck der geplanten Verarbeitung seiner Daten, die Freiwilligkeit der Einwilligung und die Möglichkeit des Widerrufs der Einwilligung informiert werden (vgl. z. B. § 4a Abs. 1 Satz 1 und 2 BDSG). Pauschale Einwilligungserklärungen, deren Tragweite der Betroffene nicht übersehen kann, sind daher unzulässig. Die Einwilligung bedarf der Schriftform, soweit nicht wegen besonderer Umstände im Einzelfall eine andere Form angemessen ist (vgl. z. B. § 4a Abs. 1 Satz 3, Abs. 2 BDSG). Die Landeskrankenhausgesetze enthalten bez. Datenübermittlungen an vor-, mit- und nachbehandelnde Ärzte und an Angehörige zum Teil hiervon abweichende Regelungen (z. B. Widerspruchsrecht des Patienten nach Information über die geplante Datenübermittlung). Spezialregelungen zur Einwilligung des Versicherten sind insbesondere im SGB V enthalten [19].

Informationsrechte des Patienten

Im Bereich der Telemedizin ist es besonders wichtig, dass der Patient in allen Verarbeitungsphasen ausreichend informiert ist über die Verarbeitung seiner personenbezogenen Daten. Dies setzt voraus, dass das ihn informierende Personal ebenfalls ausreichend informiert ist.

Es muss insbesondere auch gewährleistet sein, dass dem Patienten bei Vertragsabschluss bzw. Einwilligung Umfang, Zweck und Rechtsgrundlage der Verarbeitung seiner Daten sowie ggf. die Grundzüge des technischen Verfahrens der Verarbeitung bekannt gegeben worden sind.

Abruf von Patientendaten über ein Datennetz

Patientendaten können nach Erteilung einer Einwilligung des Patienten im Einzelfall für einen Zugriff durch den Berechtigten freigegeben werden. Ein Zum-Abruf-Bereitstellen (vgl. z. B. § 10 BDSG) von Patientendaten durch einen Arzt über ein Datennetz ist nach der gegenwärtigen Rechtslage grundsätzlich nicht zulässig. Ein Arzt ist verpflichtet, vor einer

Übermittlung zu prüfen, ob eine Befugnis zur Offenbarung der Daten an den Empfänger vorliegt.

Würde ein Arzt die Patientendaten für einen Abruf durch andere Behandlungseinrichtungen bereithalten und käme es dann zu einem Abruf, der rechtlich nicht (z. B. durch eine Einwilligung des Patienten) legitimiert ist, so hätte sich der speichernde Arzt nach § 203 StGB strafbar gemacht. Eine Offenbarung von Patientendaten kann auch dadurch vorgenommen werden, dass nicht verhindert wird, dass die Daten durch externe Dritte abgerufen werden können.

4 Die Konzeption der Überwachungstechnologie

Nachdem die technischen und rechtlichen Grundlagen für die Entwicklung telemedizinischer Anwendungen im vorigen Kapitel behandelt wurden, wird in diesem Kapitel die Konzeption der Überwachungstechnologie mit allen ihren Elementen behandelt.

Die Entwicklung einer Überwachungstechnologie für Patienten mit Herzunterstützungssystem hängt im Wesentlichen von drei Elementen ab. Das erste Element ist die Patientenumgebung als Voraussetzung für die Technologie. Das zweite Element ist der Arzt bzw. Clinical Affairs, wo die Ferndiagnose durchgeführt wird. Als drittes Element in diesem Dreieck ist die Netzwerkstruktur notwendig, die für die Herstellung einer Verbindung über das Internet zwischen der Patientenumgebung und dem Clinical Affairs dient.

Für die Entwicklung einer solchen Technologie sind die Grundlagen über die Herzunterstützungssysteme von großer Bedeutung. Aus diesem Grund wird das Thema Herzunterstützungssystem (VAD) kurz behandelt und vor allem das System der Firma Berlin Heart AG näher erläutert, da die zu entwickelnde Technologie auf Systemen von Berlin Heart angewendet wird. (Abschn. 4.1) Anschließend wird die Netzwerkarchitektur für die Überwachungstechnologie des Systems INCOR dargestellt. (Abschn. 4.2)

4.1 Definition des Begriffs Herzunterstützungssysteme

4.1.1 Definition

Herzunterstützungssysteme werden bei dekompensierter Herzinsuffizienz zur Überbrückung der Wartezeit auf ein Spenderorgan eingesetzt. Die üblichen Wartezeiten auf ein Spenderorgan betragen in Europa ca. 1 Jahr. Zur Überbrückung eignen sich Systeme, die das Patientenherz unterstützen und damit das Risiko, vor der Transplantation zu versterben,

minimieren [47]. Folgende sind im Allgemeinen die wichtigsten Anwendungsgebiete von Herzunterstützungssystemen:

- Überbrückung der Wartezeit bis zur Transplantation (Bridge to Transplant), wenn der Patient diese sonst nicht überleben würde. In 2003 wurden in Deutschland 369 Herzen transplantiert. Der Bedarf war aber etwa acht Mal so hoch. Weil Spender fehlen, sind Patienten, die noch auf der Warteliste stehen, verstorben. Künstliche Herzunterstützungssysteme können hier einen lebensrettenden Ausweg bieten, wobei die Prognose von einer rechtzeitigen Implantation abhängt. [Ärzte Zeitung, 22.04.2004]

- Vorübergehende Entlastung, wenn sich der Herzmuskel noch erholen kann (Bridge to Recovery. Z. B. bei schwerer Myokarditis. Unter Myokarditis versteht man eine Entzündung des Herzmuskels. Die Entzündung kann entweder nur den Herzmuskel betreffen oder auch den Herzbeutel miteinbeziehen [45].).

- Als Alternative zur Transplantation, wenn der Patient kein Organ empfangen kann.

„Unter mechanischer Kreislaufunterstützung versteht man die Aufrechterhaltung der Blutzirkulation durch partiellen oder totalen Ersatz der Pumpfunktion der linken, rechten oder beider Herzkammern mittels extra-oder intrakorporal gelegener mechanischer Pumpe. Als Oberbegriff für die mechanischen Kreislaufunterstützungssysteme hat sich die englische Bezeichnung Ventricular Assist Device (VAD) etabliert.“ [43].

Die unterschiedlichen Methoden der mechanischen Kreislaufunterstützung unterscheiden sich durch den Pumpmechanismus (Verdrängungsprinzip, axial, zentrifugal), Ort der Implantation (extrakorporal, intrakorporal) sowie durch die Dauer der Anwendung. Die Herz-Lungen-Maschine (HLM) ist ein Beispiel für extrakorporale mechanische Kreislaufunterstützung. Die HLM kann bis zu mehreren Stunden sowohl die Herzfunktion als auch die Lungenfunktion (mittels Oxygenator) übernehmen und ermöglicht so Operationen am nicht schlagenden, offenen Herzen [42].
Als implantierbares Herz-Unterstützungssystem wird hier das INCOR System der Berlin Heart AG näher betrachtet.

4.1.2 Implantierbares Herz-Unterstützungssystem INCOR

Die Berlin Heart AG entwickelt ein implantierbares VAD-System namens INCOR. INCOR kann langfristig die Kreislauffunktion des kranken Herzens stabilisieren. Die Einsatzmöglichkeiten reichen von der Herzunterstützung vor einer Transplantation (Bridge to Transplant) bis hin zur Erholung (Bridge to Recovery), die eine Transplantation gänzlich überflüssig machen kann. Langfristig kann ein solches System möglicherweise auch eine Alternative zur Herztransplantation (Alternative to Transplant) darstellen.

Der Controller (Steuereinheit gemäß Abbildung 4.1) kontrolliert und steuert das gesamte System. Er regelt die Stromversorgung und zeigt an, wann ein Akku ausgewechselt werden muss oder wenn eine Fehlfunktion vorliegt. An die Steuereinheit müssen immer die Pumpe sowie Haupt- und Reserveakku angeschlossen sein. Hinzu kommen das Netzgerät sowie bei Kontrolluntersuchungen der Laptop. Zu INCOR gehören je zwei Haupt- und Reserveakkus

mit identischen Leistungsdaten.

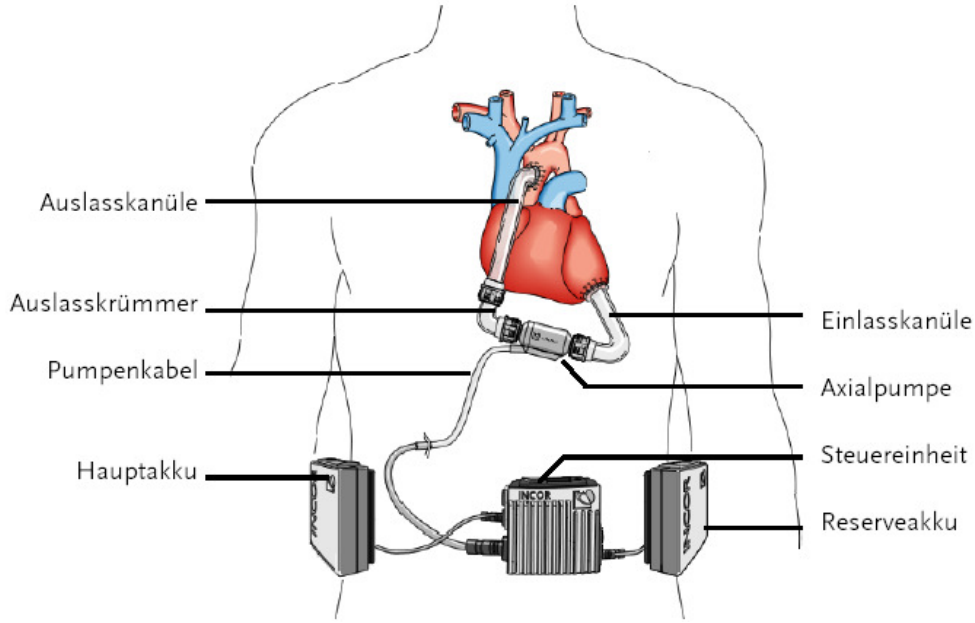


Abb. 4.1 INCOR der Firma Berlin Heart AG mit den zugehörigen Komponenten [Quelle: Benutzerhandbuch der INCOR-Systems der Firma Berlin Heart AG]

Herz-Unterstützungssysteme wie INCOR nehmen dem Herzen einen Teil dieser Pumparbeit ab. Bei INCOR fließt das Blut aus der linken Herzkammer durch die Einlasskanüle in eine elektrisch angetriebene Pumpvorrichtung und von dort aus durch die Auslasskanüle in den Körperkreislauf.

Die Pumpvorrichtung besteht aus einem Rohr, in dem sich ein Motor getriebenes Schaufelrad dreht und das Blut aktiv befördert. Das Schaufelrad dreht sich mit einer gleich bleibenden Drehzahl und fördert das Blut konstant. Anders als das menschliche Herz löst INCOR also keine Druckwelle aus.

Je nachdem, wie stark sich die linke Herzkammer zusammenzieht, ist ein stärkerer oder schwächerer Restpuls zu fühlen. Ein schwacher Restpuls weist jedoch nicht auf einen geringen Blutfluss hin, sondern zeigt, dass INCOR den größten Teil der Pumparbeit übernimmt. Von der Pumpe führt das Pumpenkabel durch die Haut nach außen und verbindet die Pumpe mit der Steuereinheit, die man außen am Körper trägt.

4.2 Die Netzwerkarchitektur für die Überwachungstechnologie

Ziel ist die Entwicklung einer zertifizierungsfähigen Technologie zur permanenten Überwachung des diagnostischen Befundes von Herz-Patienten über das Internet, sowie zur durchgehenden Dokumentation der diagnostischen Daten. Die Konzeption der Netzwerkstruktur soll die Grundlage für die Entwicklung einer telemedizinischen zertifizierungsfähigen Fernüberwachungstechnologie bilden, um eine Diagnose, Überwachung und Betreuung der Herzunterstützungssysteme zu gewährleisten.

4.2.1 Anwendungsbereich

Die zu entwickelnde Technologie wird auf Patienten mit Herzunterstützungssystemen angewendet. Der Patient ist mit einem Controller (Steuereinheit) zur Steuerung des jeweils

eingesetzten medizintechnischen Systems und einem Monitor (Laptop) zur Visualisierung und Speicherung der Betriebsdaten sowie zur Parametereingabe ausgestattet.

Der Anwendungsbereich der Überwachungstechnologie ist in dieser Arbeit auf dem Controller vom INCOR System vorgesehen. Die für die Überwachung relevanten Daten im Controller sind die Log-Daten (Log File) und die so genannten graphischen Daten. Beide Datenarten sollen mit der hier beschriebenen Technologie an den Arzt bzw. die Clinical Affairs übermittelt werden können. Die Log-Daten beinhalten unter anderem die Drehzahl der Pumpe, Temperatur und die Druckdifferenzen. Bei den graphischen Daten handelt es sich um die Aufzeichnung bestimmte Pumpenparameter. Das sind so genante Kurvenparameter, die von der Pumpe alle 9ms gesendet werden.

4.2.2 Anforderungen

Die Anforderungen an die zu entwickelnde Technologie sind folgende:

1. Die gesicherte Übertragung von aktuellen Daten (Online Monitoring) zu einem PC/Monitor bei den Clinical Affairs (passiv).
2. Die gesicherte Übertragung von Daten (Log-Daten) zu und von einem zentralen File-Server.
3. Die Entwicklung eines proprietären Protokolls, das zum einen die Übertragung der Log-Daten von der Patientenumgebung zum File-Server ermöglicht und zum anderen alle Anforderungen für die Zertifizierung und der Datensicherheit erfüllt.
4. Die Realisierung eines File-Servers zur Archivierung aller Log-Daten jedes in Verkehr gebrachten Controllers.

4.2.3 Das Konzept

Ausgehend von den genannten Anforderungen an die Überwachungstechnologie ist die in der Abbildung 4.2 dargestellte Netzwerkstruktur für die Überwachungstechnologie entstanden. Die Entwicklung dieser Technologie gemäß Abbildung gliedert sich in zwei Teile. Es handelt sich um die Entwicklung und Implementierung eines Telematik-Moduls zur permanenten Überwachung des diagnostischen Befundes von Patienten mit Herzunterstützungssystemen über das Internet und um die Entwicklung und Implementierung eines Servers zur Datenhaltung der medizinischen Daten seitens Clinical Affairs.

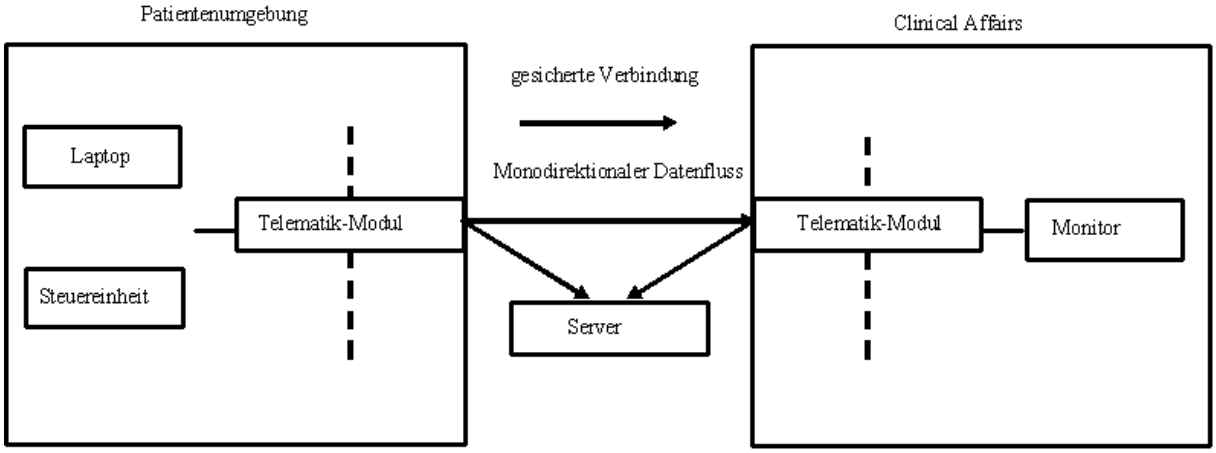


Abb. 4.2 Netzwerkstruktur der Überwachungstechnologie

Das Telematik-Modul ist der Kern der Technologie. Für die Kommunikation werden die verschiedenen Schnittstellen des Telematik-Moduls verwendet. Die folgenden Daten sind zu übertragen:

1. grafische Daten (Kurvenparameter) vom medizinischen Gerät,
2. die gesicherte Übertragung von aktuellen graphischen Daten (Online Monitoring) zu einem PC/Monitor,
3. die gesicherte Übertragung von Log-Daten zu und von einem zentralen File-Server,
4. die Übertragung von Log-Daten aus einem Laptop/PC.

Die Netzwerkstruktur, die die Patientenumgebung (medizinisches Gerät und Laptop/PC) mit dem Arzt bzw. Clinical Affairs über das Internet verbindet, beinhaltet drei Elemente. Zwei Telematik-Module werden die Verbindungen zu einem File-Server bei den Clinical Affairs ermöglichen. Der File-Server empfängt die Log-Daten von verschiedenen Patienten und archiviert sie unter der entsprechenden System-ID mit Datum und Zeit der versendeten Log-Daten. Die Datenübertragung muss verschlüsselt erfolgen, da die Anforderungen an die Datensicherheit sehr hoch sind.

Die Datenübertragung kann in drei unterschiedlichen Modi erfolgen, die nachfolgend dargestellt werden. Der erste Modus dient der Übertragung von graphischen Daten aus der Patientenumgebung zur Clinical Affairs. Der zweite Modus überträgt die Log-Daten aus dem Controller zum patientenseitigen Telematik-Modul. Im dritten Betriebsmodus werden die Log-Daten vom patientenseitigen Telematik-Modul zum klinischen Server übertragen.

Betriebsmodus „Online Monitoring“ („Online Beobachtung“)

Unter Online Monitorring versteht man die Übertragung aktueller Daten des Herzunterstützungssystems. Dabei handelt es sich um die so genannten graphischen Daten (Kurvenparameter), die der Controller von der Pumpe des Herzunterstützungssystems zum Laptop alle 9ms gesendet werden. Abbildung 4.3 zeigt die Technologie-Struktur einer solchen Online-Beobachtung.

Das System muss durch den Patienten (User) in den Online Beobachtungsmode gebracht werden. Das Telematik-Modul in der Patientenumgebung (Client) liest die seriellen Datenströme, die vom Controller zum Laptop fließen. Diese Datenströme werden über die serielle Schnittstelle via Internet direkt zum zweiten Telematik-Modul (serverseitig) weitergeleitet. Um den Online-Beobachtungsmode erfolgreich durchzuführen, müssen folgende Voraussetzungen erfüllt werden:

- Bereitschaft des Patienten.
- Patienten müssen für die Bedienung des Telematik-Moduls ausgebildet werden.
- Absprache zwischen Clinical Affairs und Patienten über den Zeitpunkt, zu dem beide Teilnehmer an dem Netzwerk bereit sind.
- Das Telematik-Modul (Server) muss eine eindeutige statische IP-Adresse bekommen.
- Das Telematik-Modul (Server) muss zuerst durch das Bedienungsmenü in den passiven Modus (listen mode) gebracht werden.
- Verfügbarkeit von TCP/IP bzw. Modem auf beiden Seiten der Kommunikation.

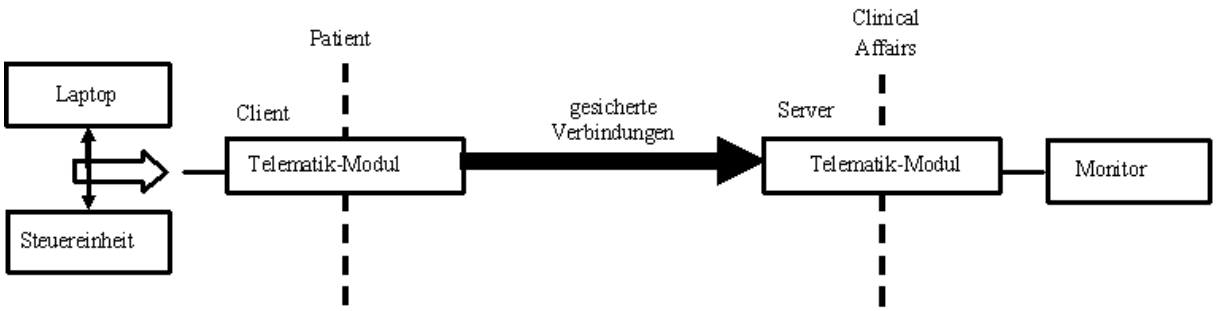


Abb. 4.3 Technologiestruktur der Online-Beobachtung

Eine weitere Variante für das Online Monitoring besteht in einer direkten seriellen bzw. drahtlosen Verbindung zwischen dem Telematik-Modul (Client) und dem Controller. Die serielle Verbindung kann drahtgebunden mit RS232 oder drahtlos mit Bluetooth Modul hergestellt werden.

In seiner aktuellen Ausstattung verfügt der Controller über eine RS232-Schnittstelle (für den Laptop-Anschluss) und lässt damit die drahtgebundene Kommunikation zu. Die Kommunikation mit dem Controller läuft mit einem bestimmten Protokoll und eindeutigen von der Anwendung abhängigen Befehlen. Die Übertragung der Kurven-Parameter wird im Controller durch eine Befehlssequenz ausgelöst, die vom Telematik-Modul über die serielle Schnittstelle an die Steuereinheit gesendet wird. Der Empfang der Kurvenparameter vom Telematik-Modul muss nicht quittiert werden, sondern wird über Internet zum zweiten Telematik-Modul (Server) übertragen. Voraussetzung hierfür ist die vorherige Herstellung der Sitzung einer Online-Beobachtung zwischen dem Patient und Clinical Affairs.

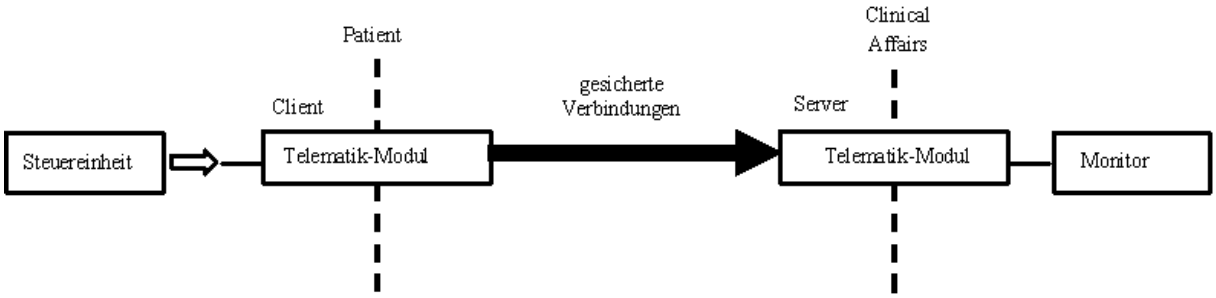


Abb.4.4 Online-Beobachtung mit direkter Verbindung zum Controller

Betriebsmodus „Log-Datenabfrage aus der Patientenumgebung“

Bevor die Log-Daten zum File-Server übertragen werden, müssen sie zuerst zum Telematik-Modul übertragen werden (siehe Abbildung 4.5). Die Log-Daten beinhalten wichtige Parameter über den Zustand der Pumpe und Akkus, der Druck, die Art der Betriebsspannung, Drehzahl der Pumpe und Controller-ID.

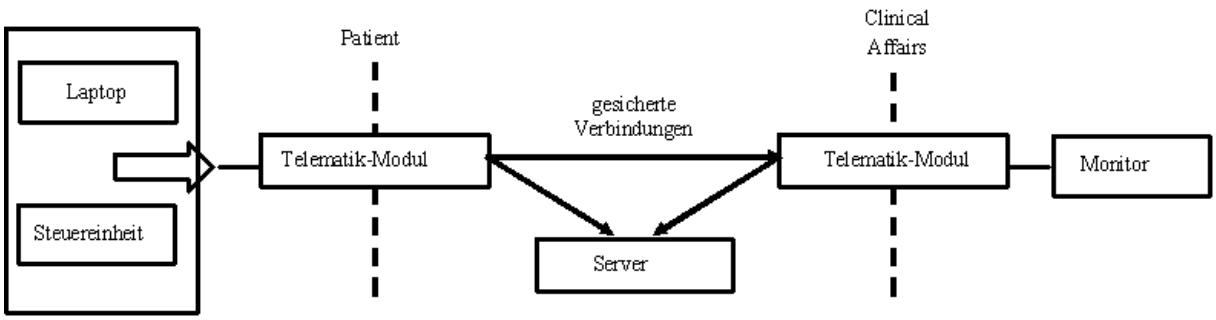


Abb. 4.5 Datenübertragung vom Controller bzw. Laptop zum Telematik-Modul

Die Log-Daten werden regelmäßig auf dem Laptop des Patienten in einem Log-File hinterlegt. Dieses ist 26065 Byte groß. Die Log-Daten sind technisch gesehen nicht anderes als ein Ereignisbuffer, den man aus dem Controller abliest. Der Ereignisbuffer besteht aus 400 Indizes. Jeder Index ist 65 Byte groß. Die Größe des Ereignisbuffers ist 400*65 Byte= 26000 Byte. Dazu kommen 65 Byte für die Steuerparameter. Dann ist die gesamte Größe der Log-Daten 26065 Byte.

Die Log-Daten sind Ascii-Zeichen, die nur unter Verwendung einer DOS-basierten Software in verständlichen Text umgewandelt werden können. Auf dem Laptop in der Patientenumgebung existiert eine benutzerfreundliche, DOS-basierte Bedienoberfläche, die mit dem Controller über die RS232-Schnittstelle kommuniziert. Auf dem Laptop werden diese Log-Daten regelmäßig abgespeichert.
Damit existieren zwei Quellen, aus denen die Log-Daten abgelesen werden können. Diese sind der Controller sowie der Laptop.

Für die Datenabfrage aus dem Laptop braucht man eine Applikation, die im selben Verzeichnis der Log-Daten liegt. Diese Applikation ermöglicht die serielle Datenübertragung zwischen dem Laptop und dem Telematik-Modul. Diese Kommunikationsmethode ermöglicht das Abgreifen der Log-Daten während der ununterbrochenen Kommunikation des Laptop mit dem Controller im klinischen Einsatz.

Für die Datenabfrage aus dem Controller sollte die entsprechende Befehlssequenz zum Controller über die RS232-Schnittstelle gesendet werden. Auf die Quittierung der Befehle muss geachtet werden, da die Quittierungsnummer des Controllers seine Bereitschaft und die fehlerfreie Verkablung wiedergibt. Bei Log-Daten- Abfrage bestätigt der Controller zuerst den Befehl mit einer Quittierungsnummer. Diese Kommunikationsmethode ist für den realen, mobilen Einsatz der Überwachungstechnologie nötig, da der Betrieb mit einem Laptop lediglich der Wartung und Anpassung des Controllers sowie der Speicherung von Betriebsdaten dient.

Betriebsmodus „Datenübertragung zum Server“

Das Ziel ist es, dem Arzt bzw. den Clinical Affairs die Möglichkeit anzubieten, mehrere Patienten gleichzeitig und regelmäßig zu überwachen. Dafür müssen erst einmal die Log-Daten gemäß Abbildung 4.6 zum Server übertragen werden. Die Anforderungen an die Datensicherheit sind gleichwertig denen beim Online-Banking. Es soll für die Datenübertragung zum Server ein Mechanismus entwickelt werden, der zum einen die Anforderung an die Datensicherheit erfüllt und zum anderen die byteweise Überprüfung der Übereinstimmung von empfangenen Daten zu den gesendeten Log-Daten gewährleistet und sichert.

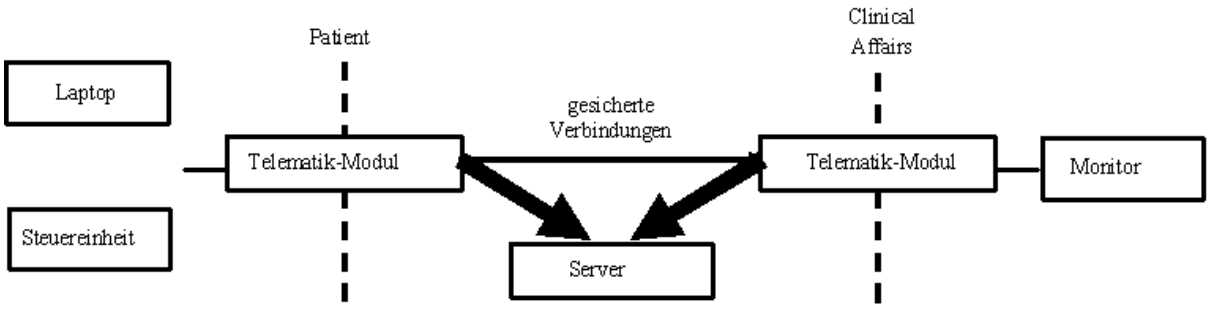


Abb.4.6 Übertragung von Log-Daten zum klinischen Server

Die Log-Daten sollen immer mit der aktuellen Zeitangabe (Datum und Uhrzeit) zum Server versendet werden. Dazu kommt die ID des Controllers. Auf dem Server sollen die Daten dementsprechend weiter verarbeitet werden, so dass sie in einer Datenbank gemäß der Controller-ID gespeichert werden. Ein in den klinischen Server integrierter Web-Server soll gleichzeitig mehrere Verbindungen über einen Standard-Browser zu den Clinical Affairs ermöglichen.

Die Zugänge zu den Log-Daten sollen mit Benutzernamen und Passwörtern geschützt sein. Die Übertragung der Daten vom klinischen Server zur Clinical Affairs soll mit SSL (Secure Socket Layer) gesichert sein.
Damit ergibt sich folgender Ablauf:

- Log-Daten werden zum Telematik-Modul (Client) geladen
- Den Log-Daten werden aktuelle Zeitangaben der auf dem Telematik-Modul laufenden Echtzeituhr angehängt
- Durch die Bedienung des Telematik-Moduls (Client) wird nach dem Laden der Log-Daten eine Internetverbindung zum Server hergestellt
- Der Server empfängt die Log-Daten und speichert sie
- Dem Clinical Affairs wird ein Zugang über Standard Web Browser zu den Log-Daten möglich sein

Um diese Betriebsmodi erfolgreich durchzuführen, müssen folgende Voraussetzungen erfüllt werden:

- Bereitschaft des Patienten
- Patienten müssen für die Bedienung des Telematik-Moduls ausgebildet werden
- Der Server muss eine eindeutige statische IP-Adresse bekommen.
- Verfügbarkeit von TCP/IP bzw. Modem auf beiden Seiten der Kommunikation

4.2.4 Schlussfolgerung

Das vorgestellte Konzept soll die Grundlage für eine Überwachungstechnologie für Patienten mit Herzunterstützungssystemen bilden. Dabei ist auf die Ausbildung des Patienten zu achten, da die Bereitschaft des Patienten im Wesentlichen das wichtigste Element für die Datenübermittlung ist, um die Ferndiagnose durchzuführen. Die technischen Voraussetzungen solcher Technologie werden mit der Entwicklung eines Telematik-Moduls und mit der Implementierung und Realisierung eines Servers (Clinical Server) geschaffen.

Die Hardware- und Software-Realisierung der Überwachungstechnologie wird im nächsten Kapitel dargestellt. Dabei handelt es sich um die Entwicklung des Telematik-Moduls (Client und Server) und um die Entwicklung eines Application Programming Interface (API) für den File-Server (Klinischer Server).

5 Realisierung der Fernüberwachungstechnologie

Nachdem das Konzept der Überwachungstechnologie behandelt wurde, wird die Implementierung aller am Netzwerk beteiligten Komponenten in diesem Kapitel beschrieben. Dabei handelt es sich um die Hardware- und Software-Implementierung eines Telematik-Moduls und um die technische Beschreibung der Arbeitsweise und der Realisierung der verschiedenen Betriebsmodi. Die Konfiguration des Moduls wird auch in diesem Kapitel behandelt. Des Weiteren wird die Implementierung und die Arbeitsweise der Schnittstelle (API) auf dem File-Server erläutert, da die Kommunikation mit dem Server bei der Übertragung von Log-Daten ein eigenes Protokoll bzw. Kommunikationsregelen auf dem TCP/IP-Stack besitzt.

5.1 Implementierung des Telematik-Moduls

Die Anforderungen an die Software und an die Hardware sind für die zu entwickelnde Technologie bzw. für das zu entwickelnde Telematik-Modul sehr umfangreich. Es war nicht so einfach, von den vorhandenen Angeboten auf dem Markt eine Wahl für die Entwicklungsumgebung und für das Entwicklungboard zu treffen.

Nach langer Recherche und Marktanalyse bezüglich Funktionalität, Speicherkapazität und Preis wurde die Entscheidung getroffen, das RCM3200 Kern-Modul [49] von Rabbit Semiconductor und Dynamic C [50] als Entwicklungsboard und Programmierumgebung für die Entwicklung des Telematik-Modul zu benutzen.

Das RCM3200-Modul hat eine integrierter 10/100Base-T Ethernet-Schnittstelle und wird bei 3,3 V mit 5-V-toleranten Ein/Ausgängen betrieben. Das Kern-Modul verfügt über 6 serielle Ports. Mithilfe von integrierten Low-EMI-Funktionen (Electro-Magnetic Interference), einschließlich Clock Spectrum Spreader, werden EMI-Probleme praktisch eliminiert, die die Durchführung von CE-und behördlichen RF-Emissionstests unterstützt.

Das RCM3200-Modul basiert auf dem Rabbit 3000-Mikroprozessor und läuft bei einer Taktgeschwindigkeit von 44,2 MHz. Es ist mit einem 512 K-Flash-Speicher, 512 K Programmausführungs-SRAM und 256 K Daten-SRAM, Quadratur-Decoder, PWM-Ausgängen und Impulserfassungsfunktionen ausgestattet. Der integrierte Ethernet-Port ermöglicht die lokale oder weltweite Konnektivität in Echtzeit.

Das RCM3200 verfügt außerdem über eine durch Batteriebetrieb gesicherte Echtzeituhr, Glueless-Speicher und E/A-Schnittstellen sowie Low-Power-„Sleepy“-Modi. Programme können mithilfe von Dynamic C von Z-World erstellt und mit einem Programmierkabel kompiliert, ausgeführt und debugged werden. Dazu ist kein schaltungsintegrierter Emulator erforderlich.

Dynamic C ist eine benutzerfreundliche C-Sprachenumgebung, die Editor, Compiler und Debugger enthält. Weitere entscheidende Vorteile von Dynamic C sind die mitgelieferten Bibliotheken, wie TCP/IP-Stack, PPP und weitere serielle Schnittstellen-Bibliotheken, die zur optimalen und schnellen Entwicklung beitragen.

5.1.1 Hardware des Telematik-Moduls

Das Blockbild gemäß Abbildung 5.1 zeigt die Komponenten der Hardware, die alle Anforderungen an die zu entwickelnde Technologie erfüllt. In diesem Abschnitt werden die einzelnen Komponenten mit deren Design und Bauelementen dargestellt. Im Anhang sind das Schaltungsdesign und das PCB-Design zu sehen. Für weitere Details wird man auf die Literatur, Datenblätter und die Internet-Seiten der Hersteller verwiesen.

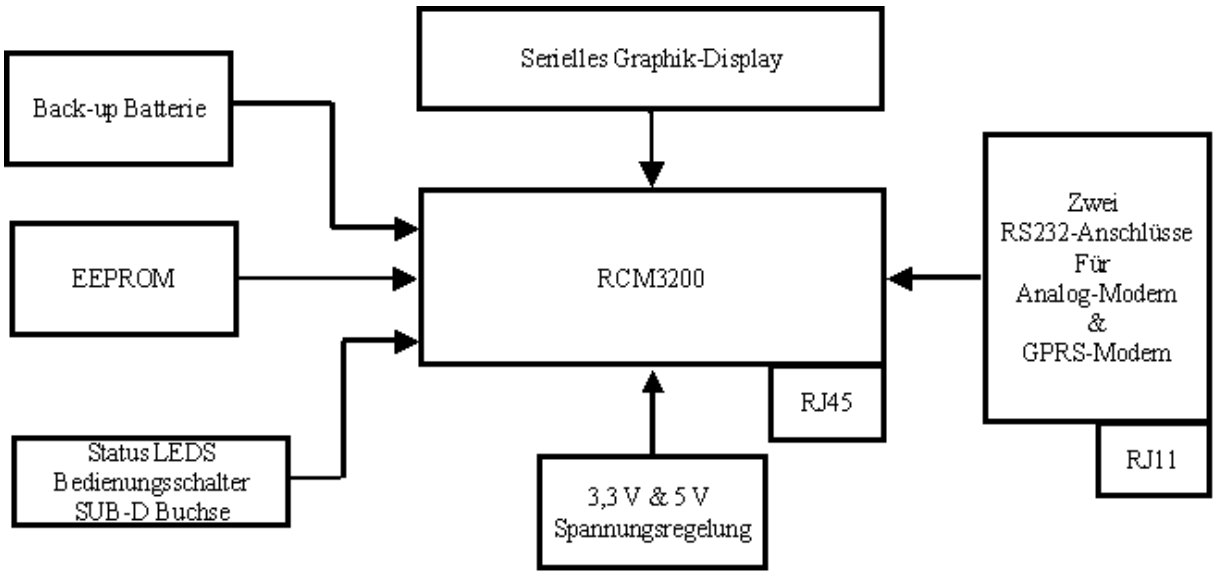


Abb. 5.1 Hardware-Blockschaltbild des Telematik-Moduls

Der Hauptteil der Hardware ist das RCM3200 Mikroprozessor-Kern-Modul. Dieses Modul kommuniziert über seine serielle Schnittstellen und Ein/ und Ausgänge mit den restlichen Komponenten auf der Trägerplatine. Die Verdrahtungen zum Kern-Modul ist durch die Dokumentation festgelegt. Alle Hardware und Software Dokumentation, User Manuals, Schaltungsdesigns von dem RCM3200 sind unter dem Internet-Link der Firma Rabbit Semiconductor zu finden. Die Abbildung 5.2 zeigt alle I/Os des Kern-Moduls.

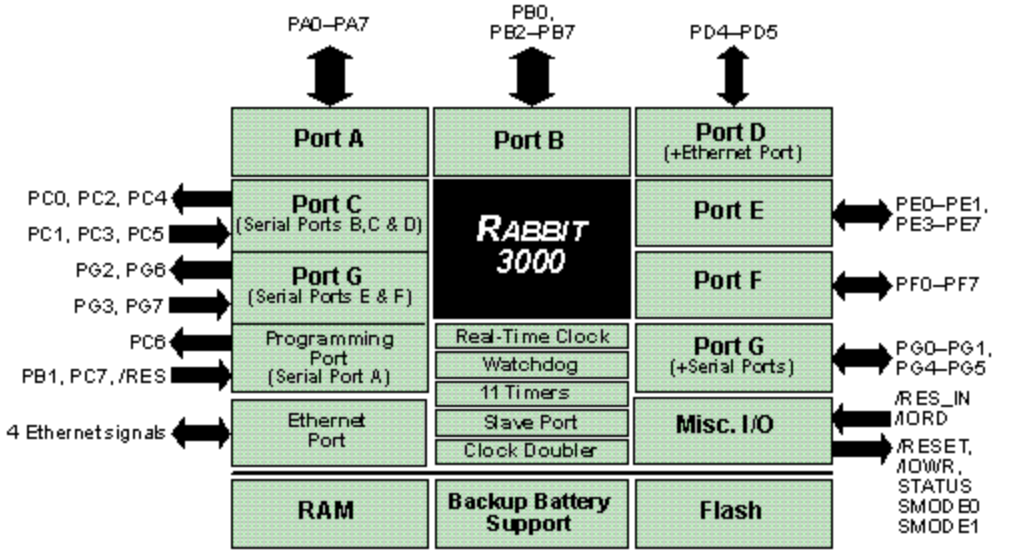


Abb. 5.2 Ports und Ein und Ausgänge vom RCM3200
[Das Bild ist aus dem Benutzerhandbuch vom RCM3200]

Für das gesamte Hardware-Design werden alle sechs seriellen Schnittstellen vom Kern-Modul benötigt. Zwei Ein- und Ausgänge für das EEPROM und acht weiteren für die Bedienungsschalter und Status-Leuchtdioden. Die Abbildung 5.2 zeigt auch alle seriellen Ports (jeweils mit den seriellen RX & TX-Leitungen) des Kern-Moduls. Die Ports werden folgendermaßen belegt:

1. Serial Port A ist für die Programmierung festgelegt.
2. Serial Port C wird für die Kommunikation mit dem analogen Modem verwendet
3. Serial Port F wird für die Kommunikation mit dem Controller und dem PC verwendet
4. Serial Port D wird für das graphische Display verwendet
5. Serial Port B wird für den zukünftigen Einsatz vom GPRS-Modem verwendet
6. Serial Port E wird für den zukünftigen Einsatz von Bluetooth verwendet

Die Verschaltung des seriellen analogen Modems ist gemäß der Abbildung 5.2 durchgeführt worden. Dies wird durch die Dokumentation vom PPP-Add-On-Modul von Dynamic C empfohlen. Am Anfang der Entwicklung der Einwahl-Verbindung wurde mit Standard Modem die Software getestet. Dann wurde das OEM-MODEM-56C/34C/32C/24C von der Firma Comtech [51] eingesetzt. Dieses embbeded Modem hat folgende für die Implementierung wichtige Eigenschaften:

- Hayes AT-command
- 300 bps bis 56,000 bps
- V90 Datenübertragungsverfahren
- TTL-Eingänge und 3,3 V serielle Ausgänge
- 5 V DC-Spannungsversorgung
- RJ11 physikalische Verbindung vorhanden

RS232 Signal	Rabbit Pin	Direction
DTR	PB6	out
RTS	PB7	out
CTS	PB0	in
DCD	PB2	in
RI	PB3	in
DSR	PB4	in
TD	PC2	out
RD	PC3	in

Abb. 5.2 Die Tabelle der Pinbelegung des analogen Modems
[Die Tabelle ist aus der Dokumentation des RCM3200 und Dynamic C]

Im Datenblatt des Modems wird das serielle Interface des Modems beschrieben. Die Verbindung zum RCM3200-Kern-Modul wird anhand der von dem Dynamic C empfohlen Pinbelegung gebaut. Das Modem bietet 3,3 V Ausgänge, die zur seriellen Kern-Modul direkt verbunden werden können. Die 5V-Modem-Eingängen brauchen entsprechende ein 5V-Pegel. Dafür wird eine weitere Komponente benötigt. Der SN74LVC1G07 Singel Buffer/Driver von Texas Instruments [52] wird für die Verbindung eingesetzt.

Für die Anzeige wird ein Grafik-Display von Electronic-Assembly [53] verwendet. Das Grafik-LCD EA GE128-6N3V24 ist an nahezu jedes Prozessorsystem anschließbar. Die Ansteuerung erfolgt über die Standard Schnittstelle RS-232. Das Display enthält komplette Grafikroutinen zur Displayausgabe sowie verschiedenste Schriftgrößen. Die Programmierung erfolgt über hochsprachenähnliche Grafikbefehle.

Das Display ist für +5V Betriebsspannung ausgelegt. Die Datenübertragung erfolgt seriell asynchron im RS-232 Format mit echten V.24 Pegeln (±10V) oder über 5V CMOS Pegeln. Das Übertragungsformat ist fest auf 8 Datenbits, 1 Stopbit, no Parity eingestellt. Die Baudrate kann über 3 Lötbrücken von 1200 Baud bis zu 115.200 Baud ausgewählt werden. Handshakeleitungen RTS und CTS stehen zur Verfügung. Bei kleinen Datenmengen ist eine Auswertung nicht erforderlich.

Die Verbindung zwischen dem LCD und dem RCM3200-Kern-Modul wird seriell mit RX und TX Leitungen aufgebaut. Dabei wird der serielle Port D vom Modul verwendet. Die RTS und CTS Leitungen werden durch Brücken entweder kurzgeschlossen oder mit dem Modul verbunden. Das LCD wird direkt an dem Kern-Modul angeschlossen. Das Datenblatt des Grafik-LCD beschreibt die Schritte, die durchgeführt werden sollen, wenn eine direkte Verbindung zu einem 5 V I/O eines Mikroprozessorsystems angestrebt wird. Dies kann über Lötbrücken ausgewählt werden.

Die Spannungsregelung des Telematik-Moduls wird gemäß der Schaltung in Abbildung 5.3 realisiert. Dabei wird die Netzteil-Spannung von 9 V auf 3,3 V und 5 V herabgesetzt. LM2673S50 regelt die Spannung auf 5 V und LM1117MP33 auf 3,3 V [58].
Die Spannungspegel werden das Kern-Modul, die beiden Modems (Analog, GPRS) und das Graphik-Display mit jeweils 3,3, 5 und 5 V versorgen.

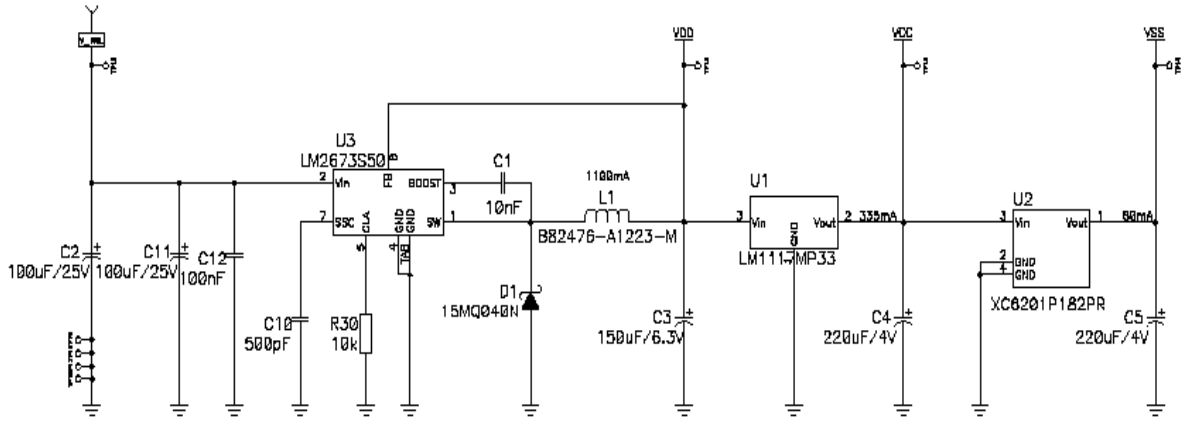


Abb. 5.3 Spannungsregelung des Telematik-Moduls

Eine weitere wichtige Komponente der Hardware ist die RS232-SUB-D-Buchse für die Verbindung zur Patientenumgebung. Dabei handelt es sich sowohl um die Verbindung zum Herzunterstützungssystem, als auch um die Verbindung zum Laptop bzw. zum PC. Die Baudrate zum PC wird auf 9600 Baud eingestellt. Die Baudrate zum Controller ist festgelegt auf 38400 Baud. Diese beiden Baudraten werden durch die Software eingestellt, dabei wird hardwaremäßig nur der serielle Port F des Kern-Moduls verwendet.
Es wird grundsätzlich ein Schnittstellenbaustein zur Pegelwandlung benötigt.

Der ICL3232CBN von Intersil [54] wurde ausgewählt. Der übernimmt die Umsetzung TTL-Pegel in V24-Pegel und umgekehrt[41]. Der Baustein arbeitet nach dem Ladungspumpenprinzip und liefert dementsprechend durch seine +/-6 V auch die benötigte Spannung, die den V24-Adpter des Controllers von Berlin Heart AG versorgt. Ein Schnittstellenmoduls V24_Protect ist eine weitere Komponente des INCOR-System. V24_Protect dient zur Potentialtrennung >4kV der seriellen Schnittstelle RS-232 zwischen dem Controller INCOR und dem Laptop.

Das PC-Interface des Telematik-Moduls wird durch ein Null-Modem mit Loop Back Handshaking gemäß der Abbildung 5.4 realisiert.

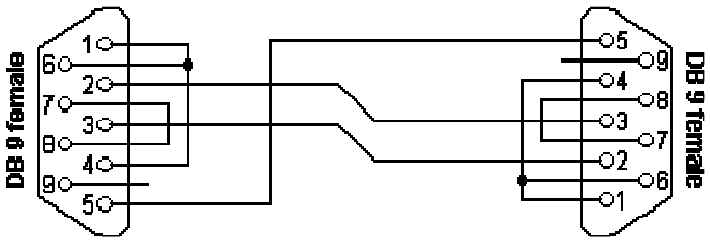


Abb. 5.4 SUB-D Anschluss für die PC-Kommunikation des Telematik-Moduls

Des Weiteren wird noch ein EEPROM (Electrically eraseable programmable-ROM) benötigt, um die Konfigurationsparameter des Telematik-Moduls zu speichern. Der 24LC16B/SN von Microchip [58] wird zu diesem Zweck verwendet. Der Baustein wird von Dynamic C empfohlen, da die I2C-Treiber bzw. Bibliothek für ihn angepasst sind. 24LC16B ist ein 16 KByte I2C serielles, elektrisch löschbares und programierbares ROM. Er ist in acht Speicherblöcke organisiert mit jeweils 256*8 Bit. Die Verbindung mit dem Kern-Modul wird wie in der Abbildung 5.5 durch zwei serielle I2C-Leitungen (SCL und SDA) realisiert [39].

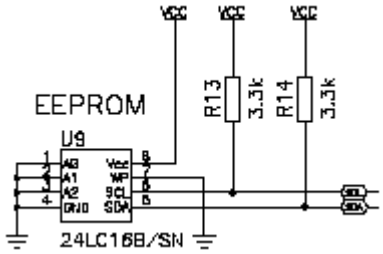


Abb. 5.5 Beschaltung des I2C EEPROMS vom Telematik-Moduls

Weiterhin wurde eine Backup-Batterie in das Design eingefügt, um die Echtzeituhr des RCM3200 im ausgeschalteten Zustand in Betrieb zu halten. Für die Bedienung des Telematik-Moduls werden Status-Leuchtdioden und Bedienungsschalter gebraucht. Auf die Bedeutung und Arbeitsweise dieser Schalter wird später darauf eingegangen. Es wurden folgende Pins des Kern-Moduls ausgewählt:

- Port F Pin 0 ist OK bzw. Ja-Taste
- Port F Pin 1 ist Nein-Taste
- Port F Pin 4 ist Konfigurationstaste
- Port F Pin 3 ist Server-Mode-Taste

5.1.2 Software des Telematik-Moduls

Im letzten Kapitel wurde die Konzeption der Überwachungstechnologie dargestellt. Dabei handelte es sich um die Netzwerkstruktur der Beteiligten gemäß der Abbildung 4.2 (siehe Seite 25). In diesem Abschnitt werden alle Betriebsmodi und die Software-Komponenten im Detail beschrieben und Arbeitsweise und Flussdiagramme der implementierten Software-Module dargestellt.

Die Abbildung 5.6 zeigt alle implementierten Software-Module im Telematik-Modul, um alle Anforderungen und Varianten der Datenübertragung gemäß dem Technologiekonzept zu realisieren.

Die Eingabe bezeichnet dabei die Bedienung des Telematik-Moduls durch die zur Verfügung stehenden Schalter. Eine Konfigurationstaste beginnt die Kommunikation zum PC im Fall, dass das Telematik-Modul durch eine auf dem PC laufende DOS-Basierten Konsole konfiguriert werden soll. Eine weitere Taste versetzt das System zu einem Server (Telematik-Modul bei Clinical Affairs), um die Online-Beobachtung von Seiten der Clinical Affairs zu starten. Die Verbindung zum Internet wird wahlweise entweder durch die Einwahl-Verbindung oder durch Ethernet hergestellt. Die dritte Variante wäre, das Modul als Client bei der Patientenumgebung in Betrieb zu nehmen.

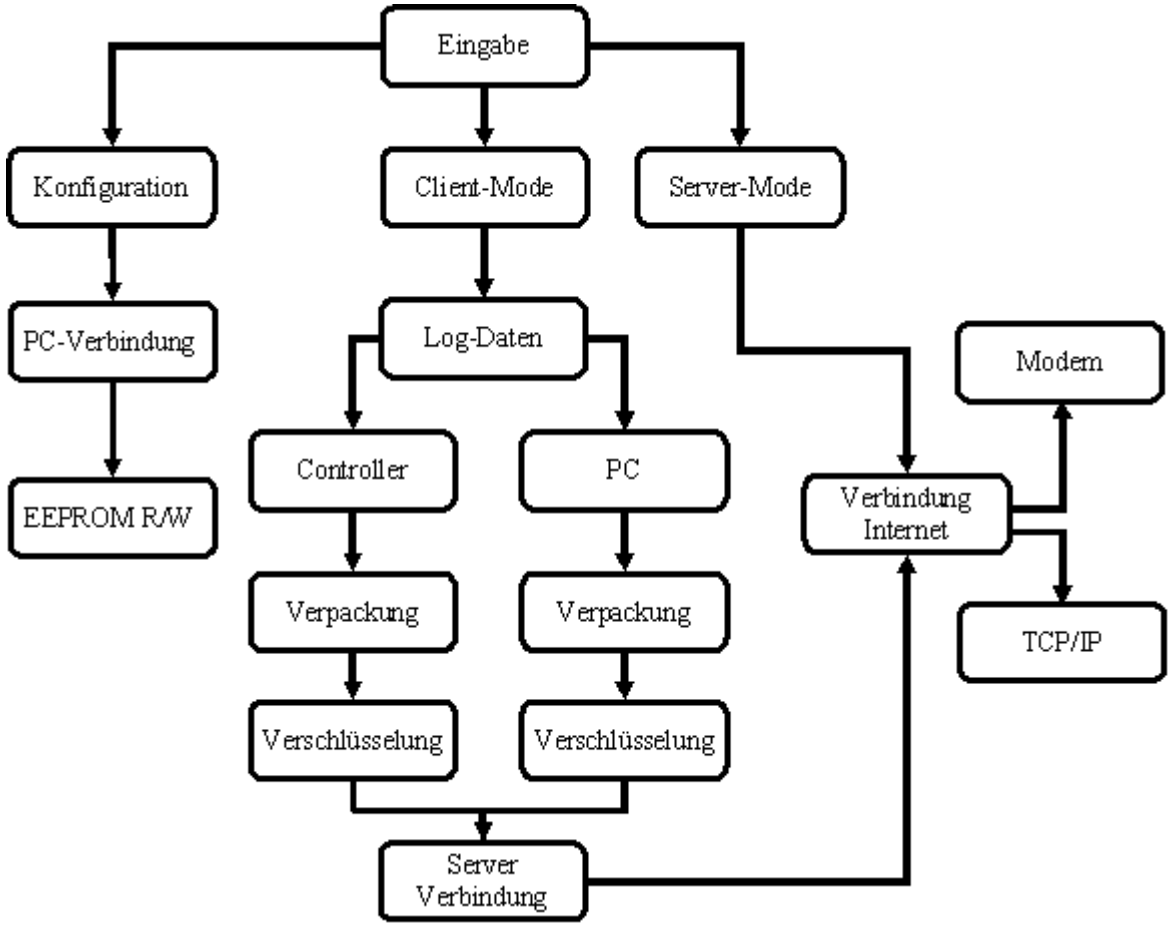


Abb. 5.6 Blockschaltbild der Software-Module des Telematik-Moduls

Beim Client-Mode handelt es sich um das Telematik-Modul in der Patientenumgebung. Bei diesem Modus werden abwechselnd eine Ja- bzw. OK- und Nein-Taste vom User (in dem Fall Patient) betätigt, um entweder die Online-Beobachtung zu starten oder Log-Daten abzufragen bzw. Log-Daten zum Server über eine Internet-Verbindung zu verschicken. Die Abbildung 5.7 zeigt den Ablauf der Bedienung des Telematik-Moduls in diesem Modus.

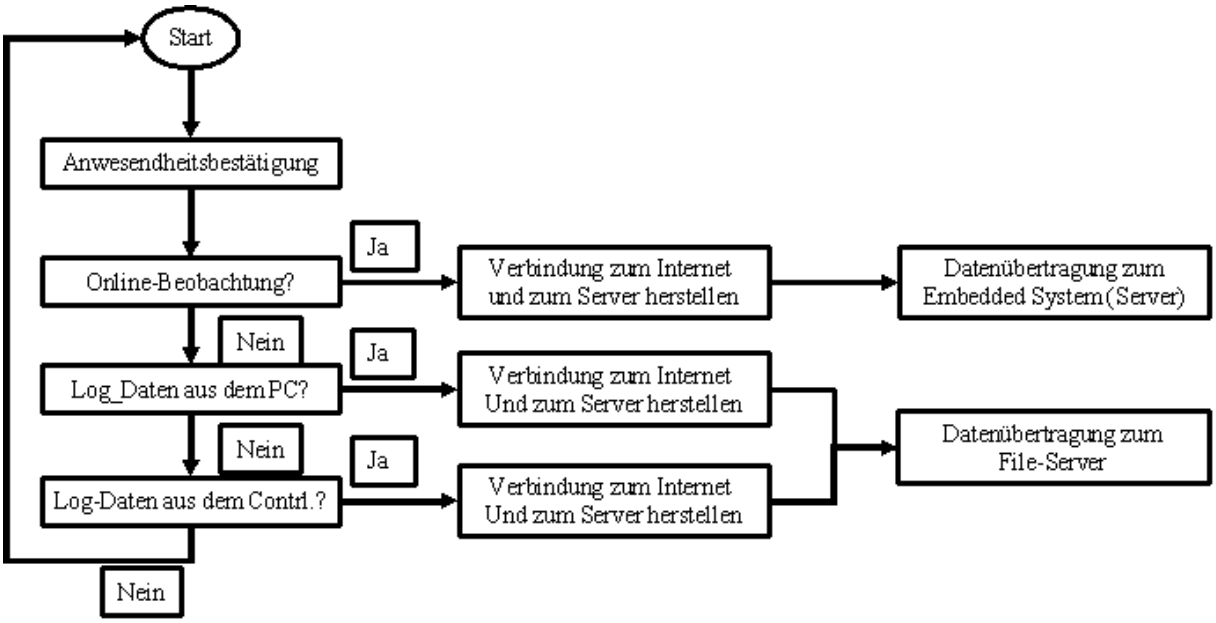


Abb. 5.7 Auswahlmenü des Telematik-Moduls in Client-Mode

Die Anzeige von Nachrichten und dem Auswahlmenü wird durch die Implementierung einer API für das Grafik-Display realisiert. Die Betätigung von Tasten wird bei der Abfrage ausgewählter Pins vom Kern-Modul wahrgenommen. Beide Quellcodes sind im Anhang meiner Arbeit unter den Funktionsnamen msg_n und detect_key zu finden. Der Quellcode der erzeugten Bibliothek für das Grafik-Display ist auch im Anhang eingefügt worden. Diese Bibliothek wurde für die Anwendung an den Dynamic C Standard angepasst. Der Code beinhaltet alle Funktionen, Grafiken und Nachrichten, die in das Telematik-Modul integriert worden. Die Verbindung zum Internet ist auch von großer Bedeutung, von daher wurde der Quellcode der beiden Verbindungsarten zum Internet im Anhang unter den Funktionsnamen Internet, ISP_ON und ISP_OFF eingefügt.

5.2 Das Online Monitoring (Online-Beobachtung)

Definition und Arbeitsweise dieser Betriebsmodi wurde im letzten Kapitel behandelt. Die Realisierung erfolgte aber gemäß dem in der Abbildung 5.8 dargestellten Ablaufsdiagramms. Erstens muss das Telematik-Modul mit statischer IP-Adresse auf Seiten der Clinical Affairs in den Server-Mode versetzt werden. Der Online-Beobachtungs-Modus wird durch das in der Abbildung 5.7 dargestellte Bedienungsschema gestartet.

Die Sicherheit der Verbindung zwischen Patientenumgebung und Clinical Affairs wird gewährleistet, indem vor der Datenübertragung ein Austausch von verschlüsselten geheimen Nachrichten auf beiden Seiten erfolgt. Ein symmetrisches Verschlüsselungsverfahren ist für diese Anwendung sehr gut geeignet, da nur ein einziger geheimer Schlüssel für die Verschlüsselung und Entschlüsselung auf den beiden Telematik-Modulen ausreicht.

Zur Datensicherheit gehört auch die Bandbreite des verwendeten Internetanschlusses.
Da die serielle Kommunikation immer einen Buffer zur Verfügung hat, muss immer darauf geachtet werden, dass keine Daten überschrieben werden. Dieser Fall tritt auf, wenn die Baudrate der seriellen Übertragung im Vergleich zur Bandbreite bzw. zur Internetverbindung sehr groß ist. Dann sind die ausströmenden Daten über das Internet vom Telematik-Modul (Client) viel geringer im Vergleich zu den einströmenden Daten vom seriellen Port. Es muss deswegen immer eine Abfrage des Buffers erfolgen, die bei einem buffer overflow die Sitzung auf beiden Seiten des Netzwerkes zum wiederholten Senden der Daten ermöglicht.

Des Weiteren sollen passende Nachrichten im Fehlerfall auf beiden Telematik-Modulen angezeigt werden, so dass beide Seiten den Fehlerfall bemerken und bewusst die Sitzung wiederholen. Die Online-Beobachtung wird beendet, wenn ausreichende Daten beim Clinical Affairs zur Analyse angekommen sind, um Patienten mit dem Herzunterstützungssystem zu betreuen. Dies wird durch eine passende Nachricht vom Auswahlmenü (Online-Beobachtung beenden?) angezeigt. Beim Betätigen der Ja-Taste wird die Sitzung beendet.

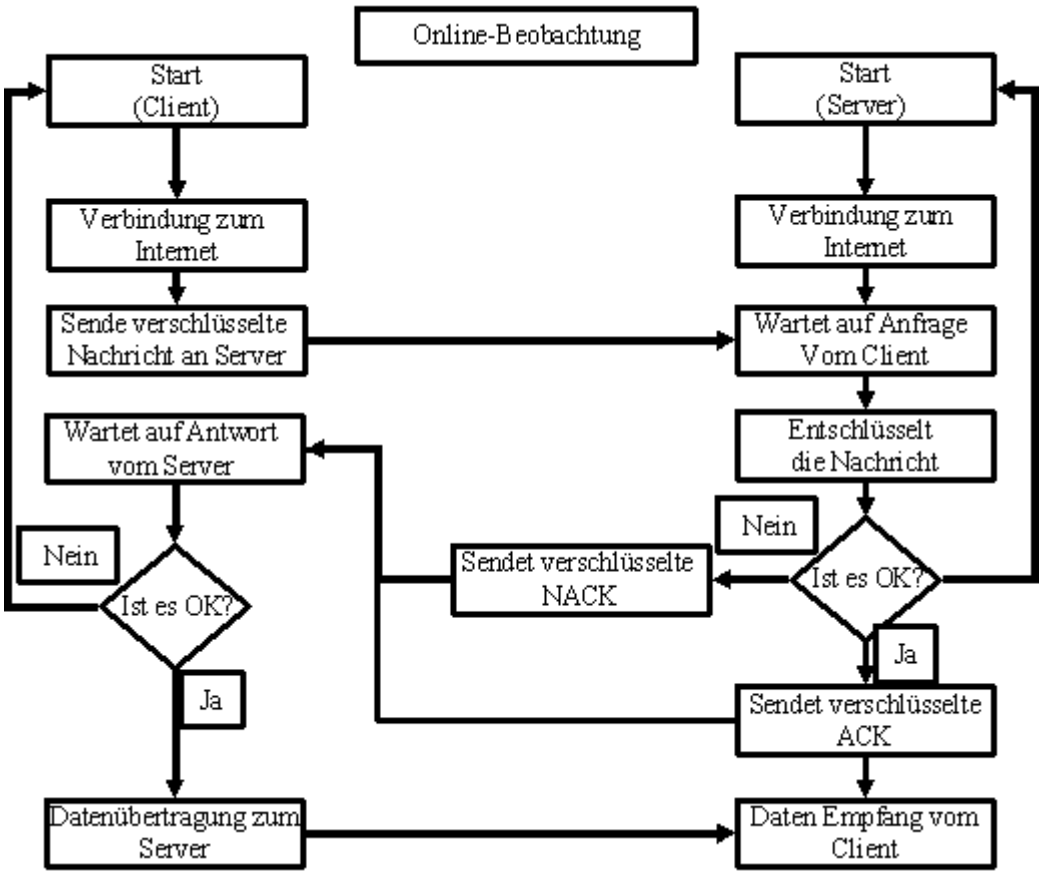


Abb. 5.8 Flussdiagramm der Online-Beobachtungs-Mode

5.3 Kommunikation mit dem PC/Laptop

Das Telematik-Modul erfüllt zwei Aufgaben bei seiner Kommunikation mit dem PC. Entweder um die Konfiguration durchzuführen oder um Log-Daten aus dem Patienten-Laptop abzufragen. Es ergeben sich an dieser Stelle folgende Fragen:
Woher und wie kommen die Log-Daten zum Patienten-Laptop?
Welche Parameter werden bei der Konfiguration des Telematik-Moduls eingestellt?

Die Kommunikation zum PC wird in den beiden Fällen durch die serielle Schnittstelle und mit einem Null-Modem-Kabel durchgeführt. Es wird eine auf dem PC laufende Applikation benötigt. Diese Applikation ermöglicht nicht nur die Übertragung der Log-Daten zum Telematik-Modul sondern auch, um die Konfigurations-Parameter zu verändern und dementsprechend die Änderungen in das Telematik-Modul abzuspeichern. Die Abbildung 5.9 zeigt das Flussdiagramm einer DOS-basierten Applikation bei der Kommunikation mit dem Telematik-Modul. Diese in TURBO C geschriebene Applikation muss über den vorhandenen seriellen Port gestartet werden.

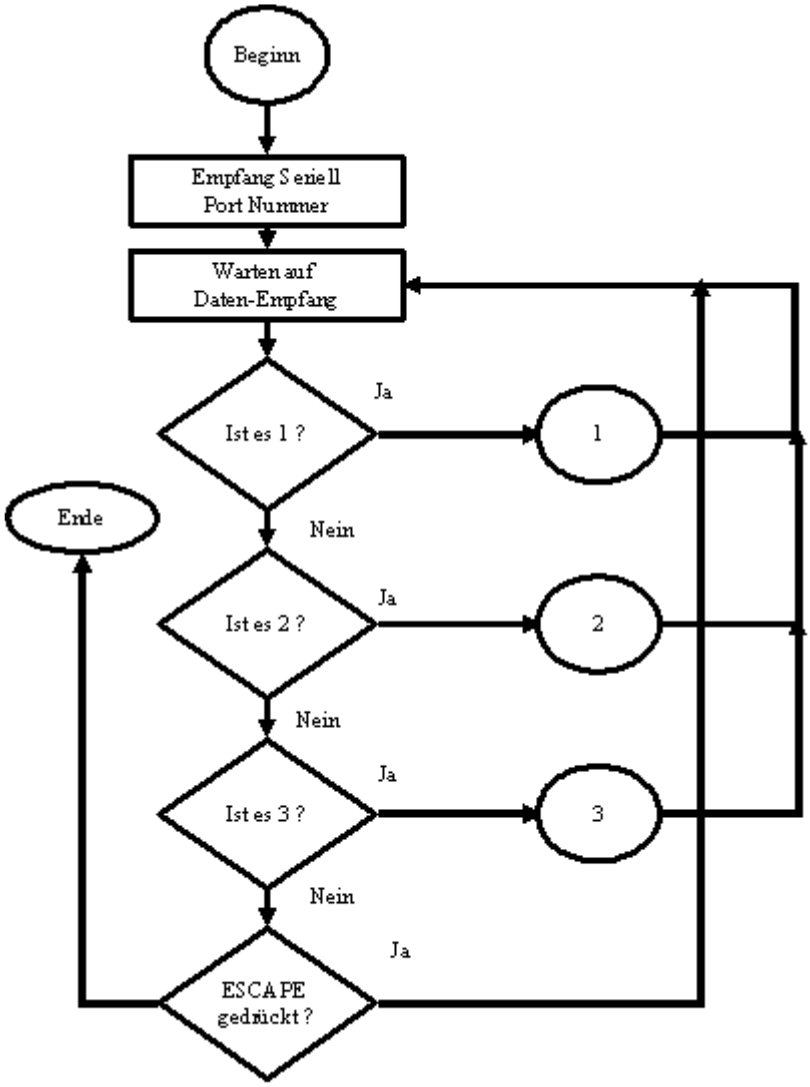


Abb. 5.9 Flussdiagramm der DOS-basierten Konsole

Nachdem die Applikation über eine auf dem PC vorhandene serielle Schnittstelle gestartet wurde, erwartet sie vom Telematik-Modul ein Zeichen (In dem Fall eine Zahl zwischen 1 und 3). Diese Zahlen werden dem PC über die serielle Schnittstelle durch die Eingabe vom User übertragen. Falls die empfangene Zahl 3 lautet, bedeutet dies, dass das Telematik-Modul die Konfigurationsmodi haben will. Die Konfiguration läuft wie in der Abbildung 5.10 dargestellten Flussdiagramms.

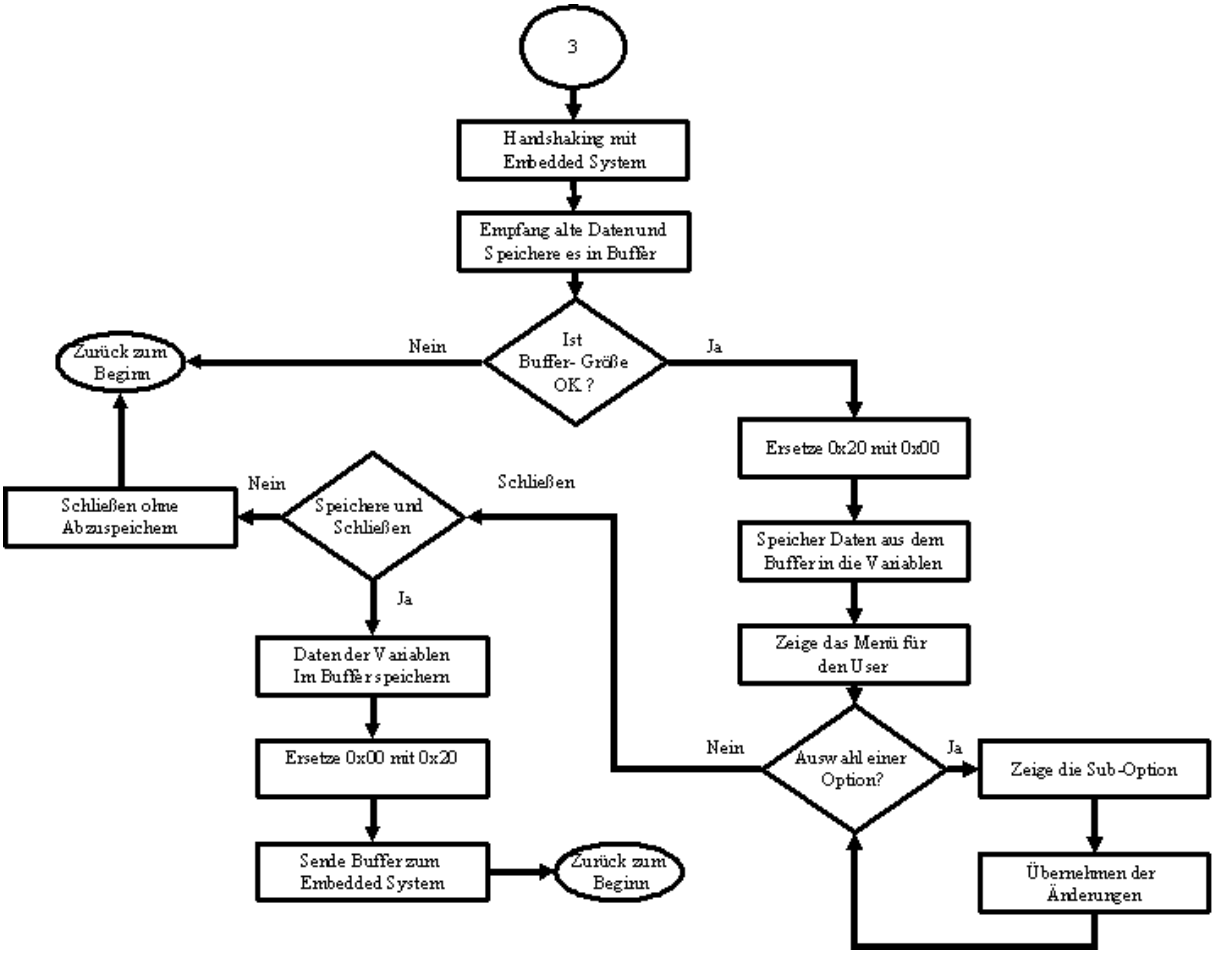


Abb. 5.10 Flussdiagramm der Konfiguration bei der DOS-basierten Applikation. Diese Abbildung hat einen Zusammenhang mit der Abbildung 5.9

Bei der Konfiguration handelt es sich um folgende Parameter:

1. Ethernet
- IP-Adresse
- Net Maske
- Standard Gateway
2. Modem
- ISP-Nummer
- Benutzername
- Passwort
3. Zeit, Datum
4. ID (7 Ziffern)

Die Konfigurationsparameter werden bei jeder Sitzung in einem Buffer immer zwischen dem I2C-EEPROM des Telematik-Moduls und dem PC hin und her übertragen. Dann wählt der User mittels der Konsole eine SUB-Option, die ihm die gewünschte Einstellung ermöglicht. Die Konsole mit allen Sub-Optionen ist in der Abbildung 5.11.a zu sehen. Die Auswahl einer SUB-Option erfolgt durch die Eingabe der jeweiligen Nummer der Option. Die Abbildung 5.11.b zeigt die SUB-Option der Ethernetkonfigurationsparameter des Telematik-Moduls. Die Ethernet-Konfigurations-Maske wird aufgerufen, wenn man die 1 in die Startmaske der Konfiguration eingegeben hat.

Die weiteren Konfigurationsparameter werden ähnlich wie die Ethernetmaske aufgerufen. Es handelt sich in dem Fall um die Einstellungen der Einwahlverbindungen, Eingabe eines Identifikationscodes für die zu überwachende Steuereinheit, Einstellung von Zeit & Datum und das Beenden der Sitzung. Beim Beenden ist deutlich in der Abbildung zu sehen, dass es sich entweder um den Abbruch der Sitzung oder um die Speicherung der geänderten Konfigurationsparameter in das Telematik-Modul handelt.

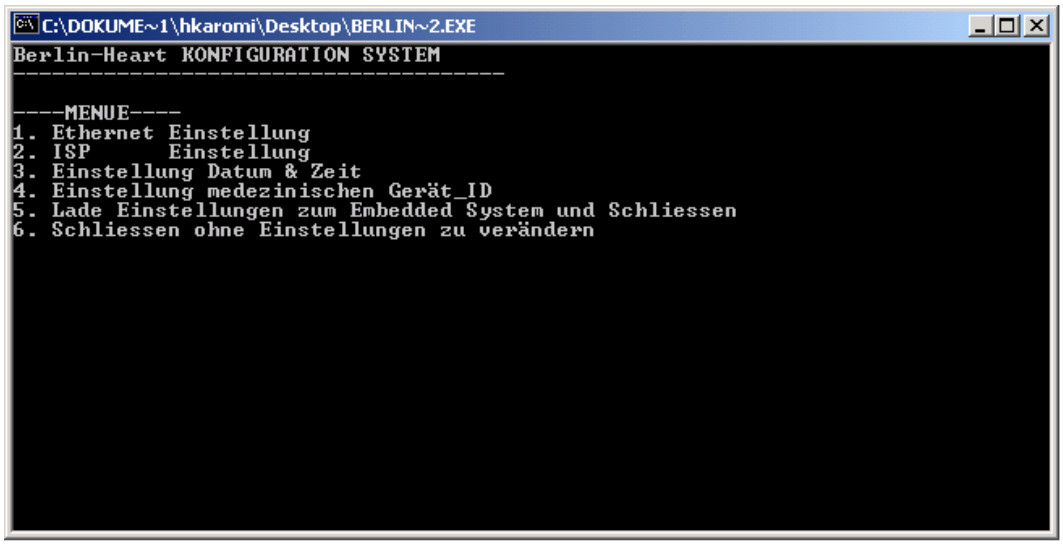


Abb.5.11.a SUB-Optionen der Konfigurationskonsole des Systems

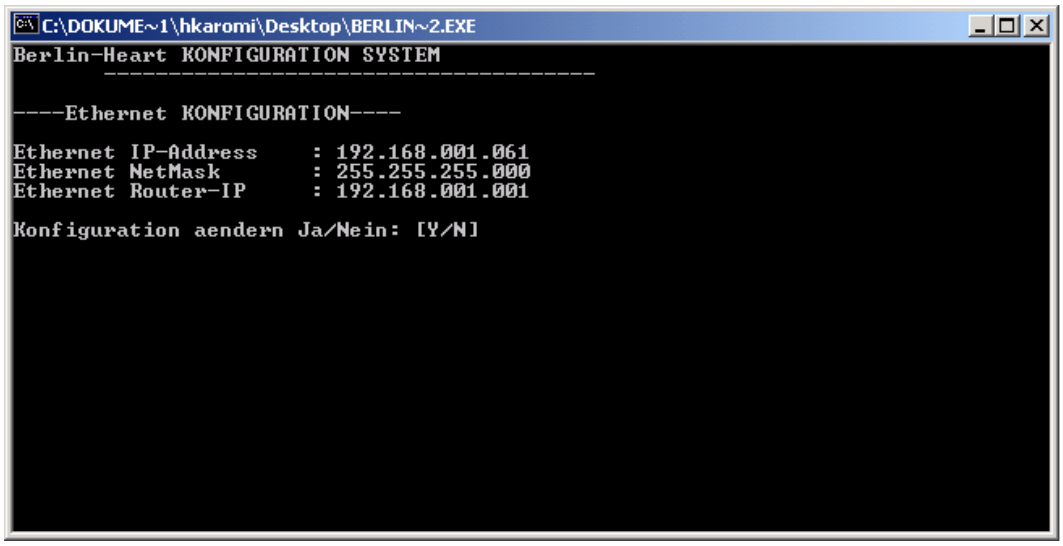


Abb. 5.11.b Die Ethernetkonfigurationsparameter des Systems

Die zweite Funktionalität der Konsole ist das Senden von den Log-Daten zum Telematik-Modul. Diese Log-Daten bezüglich des Herzunterstützungssystems (Controller) werden regelmäßig von einer existierenden Laptop-Software abgespeichert. Das Flussdiagramm der Datenübertragung stellt die Abbildung 5.12 dar.

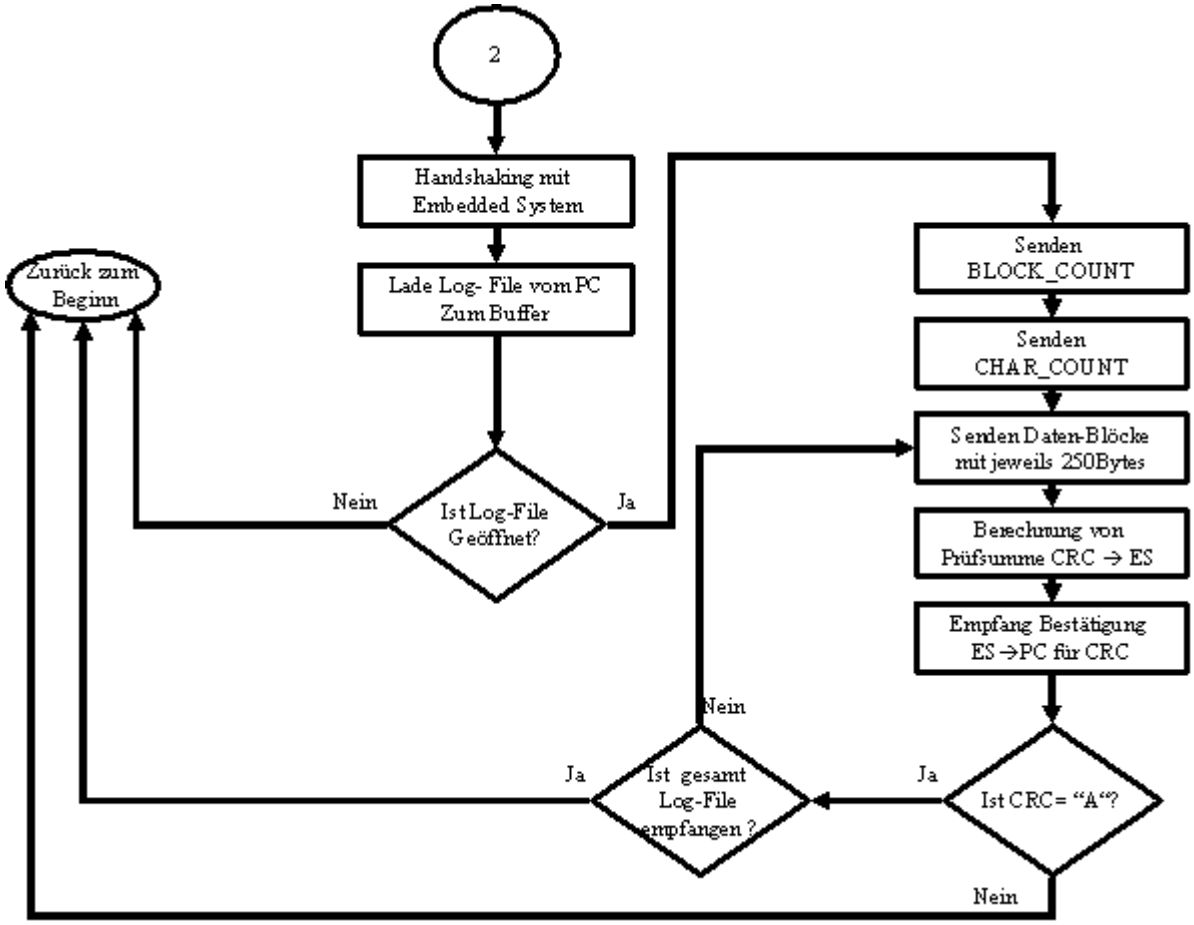


Abb. 5.12 Flussdiagramm der Log-Daten-Übertragung vom PC zum Telematik-Modul

Die Log-Daten haben immer eine maximale Größe von 26065 Byte. Diese maximale Größe wird bei der Entwicklung in 108 BLOCK_CONT von jeweils 250 Byte Blöcke geteilt. Diese Blöcke werden zum Telematik-Modul mit einer Prüfsumme (CRC-Cycel Redundancy Check) angehängt übertragen. Das Telematik-Modul bestätigt den Empfang jedes Blockes, indem es die Prüfsumme seiner Seite vergleicht. Falls die Prüfsumme nicht übereinstimmt, bricht er die Kommunikation ab und sendet ein NACK zur Konsole. Die Konsole zeigt dann eine Fehlermeldung und kehrt zu der Anfangsmaske zurück. Die CHAR_COUNT sind die letzten Bytes in dem letzten zu übertragendem Block. Auf diesem Mechanismus basiert auch die Übertragung dieser Log-Daten zum klinischen Server.

Die dritte Funktionalität der Konsole ist der Empfang von Log-Daten aus dem Telematik-Modul. Dies ist möglich durch den Empfang der Zahl 1. Diese Variante ist angebracht, wenn man in der Zukunft eine Punkt-zu-Punkt-Verbindung zwischen den beiden Telematik-Modulen (Client und Server) für die Übertragung von Log-Daten realisieren möchte, indem dann das Telematik-Modul (Server) die Daten wie im Falle der Online-Beobachtung empfängt und an den PC weiter leitet. Weiterhin wird diese Funktion bei der Analyse und beim Test des eingesetzten Mechanismus der Datenübertragung verwendet.

Der Ablauf ist in der Abbildung 5.13 dargestellt. Es handelt sich um die Übertragung von 250 Byte Blöcke, BOLCK_CONT und CHAR_COUNT, aber in die umgekehrte Richtung des in der Abbildung 5.12 dargestellten Diagramms. Hier wird ein File auf dem PC mit der Bezeichnung der aktuellen Zeitangabe erzeugt.

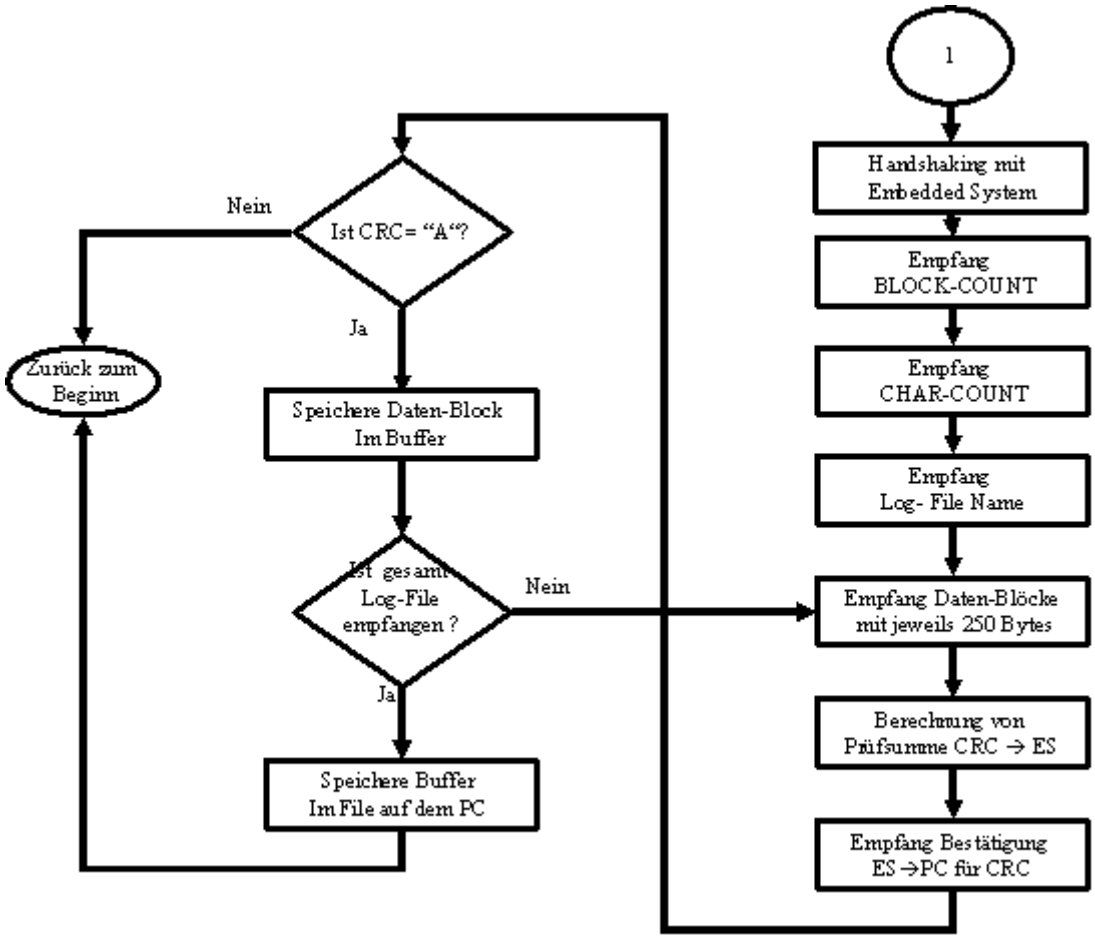


Abb. 5.13 Flussdiagramm der Übertragung der Log-Daten vom Telematik-Modul zum PC

Wenn bei allen vorhin erwähnten Varianten die Übertragung beendet oder abgebrochen wurde, wird die Applikation auf dem PC zu ihrem Ausgangspunkt zurückkehren. Auf dem Telematik-Modul wird eine passende Nachricht angezeigt. Diese Nachricht teilt dem User den Grund des Abbruches mit. Wenn die Daten aber in allen oben genannten Abläufen vollständig und fehlerfrei geladen worden sind, wird dem User des Telematik-Moduls eine Nachricht auf dem Grafik-Display angezeigt, die ihm den aktuellen Zustand mitteilt und über das Auswahlmenü mit Kombination der zur Verfügung stehenden Tasten (OK, JA, NEIN) weiter führt.

5.4 Abfragen der Log-Daten aus dem INCOR-Controller

Der Controller ist, wie bereits erwähnt, die Steuereinheit des Herzunterstützungssystems INCOR der Firma Berlin Heart AG. Da die zu entwickelnde Technologie an der Patientenumgebung solch eines Systems angewendet wird, wird eine weitere Abfrage der Log-Daten direkt aus dem Controller notwendig sein. Das ganze läuft wie in dem in Abbildung 5.14 dargestellten Flussdiagramm. Der Controller kommuniziert grundsätzlich durch Versenden von festgelegten Befehlen über seine serielle Schnittstelle. Diese Befehle veranlasst den Controller bestimmte Parameter wiederzugeben. Im Fall der Log-Daten handelt es sich nicht um die Kurvenparameter, sondern um die Abfrage eines Ereignisspeichers im Controller.

Die Befehle des Controllers sind in einer Form festgelegt, dass sie Anfang- und Endflag bzw. Anfang und Endzeichen beinhalten. Dazwischen sind Parameter und eine Prüfsumme eingefügt, die eine bestimmte Aktion im Controller auslösen.

Wenn man die Abbildung verfolgt, stellt man fest, dass die Interaktion mit dem Controller sequentiell abläuft. Wenn die Log-Daten vollständig und fehlerfrei übertragen wurden, werden sie dementsprechend in dem Telematik-Modul abgespeichert. Die Interaktion mit dem Controller läuft immer mit passenden Nachrichten an den User bzw. Patienten des Telematik-Moduls ab.

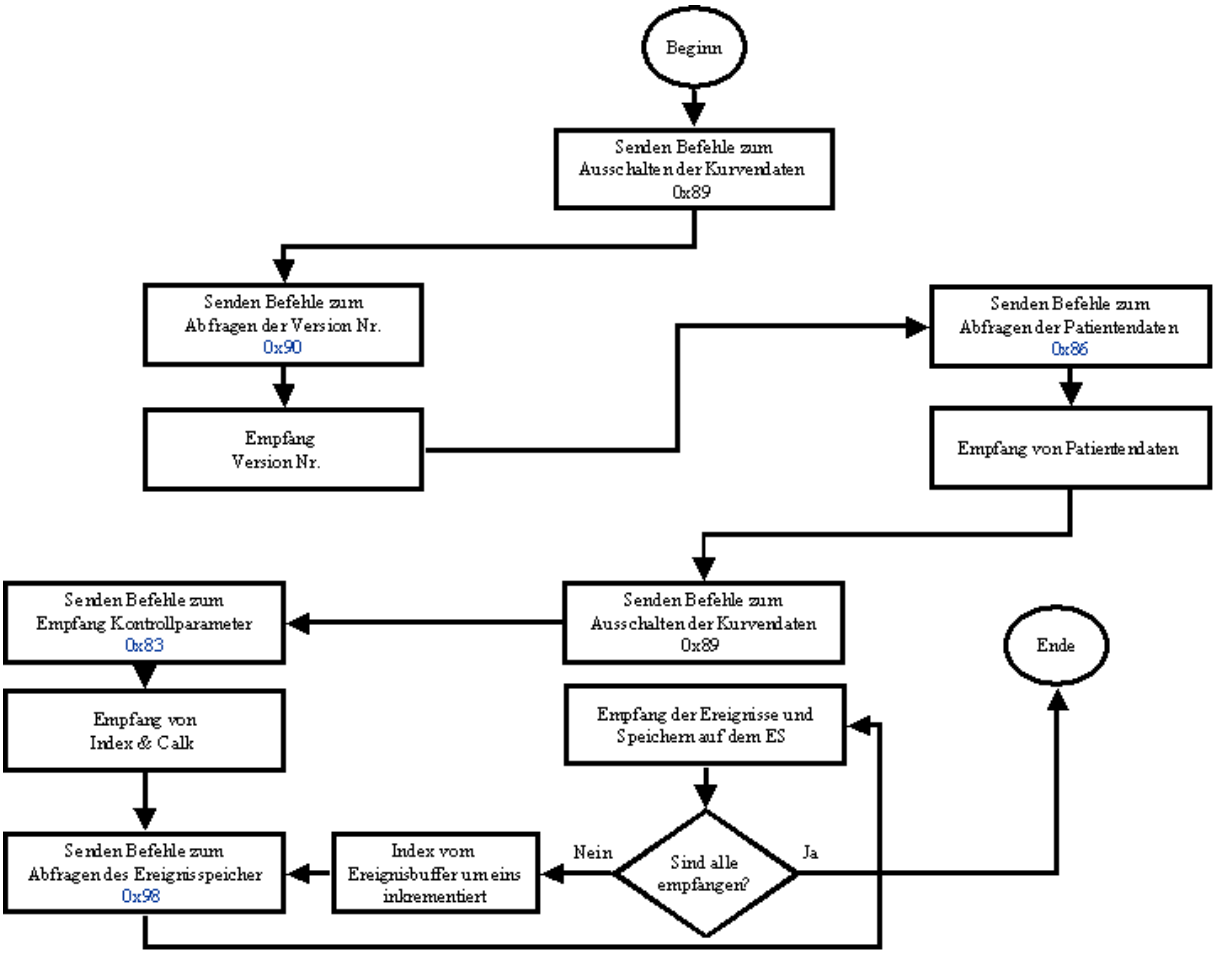


Abb. 5.14 Ablaufsdiagramm der Abfrage des Controllers

5.5 Betriebsmodus „Log-Daten zum Server übertragen“

Nachdem die Log-Daten aus einer Quelle der Patientenumgebung zum Telematik-Modul übertragen worden sind, muss gemäß Abbildung 5.7 eine Internet-Verbindung entweder über Modem oder über TCP/IP (Ethernet) zum File-Server hergestellt werden. Nachdem Aufbau bzw. der Herstellung der Internet-Verbindung sollen die Log-Daten zum Server übertragen werden.

Das Telematik-Modul stellt immer ein Auswahlmenü zur Verfügung, die eine Bedienung und Bestätigung von der Patientenumgebung benötigt, da eine der klaren Anforderungen bzw. Voraussetzungen für die Zertifizierung dieser Technologie darin besteht, dass die Überwachung als patientenbewusstes Handeln durchgeführt werden muss. Dieses Auswahlmenü führt einen Dialog mit der Patientenumgebung. Dieser Dialog wird die Bereitschaft, Anwesenheit und Aufmerksamkeit des Patienten überprüfen. Der Dialog gehört auch zur Bedienung des Systems. Diese Bedienung zeigt durch passende Nachrichten und Bestätigungen der Interaktion zwischen dem Telematik-Modul und der Patientenumgebung im fehlerbehafteten und fehlerfreien Fall an.

Die weiteren Anforderungen an die Überwachungstechnologie waren so umfangreich, dass klar wurde, dass neue und nicht standardisierte Kommunikationsregeln in der Anwendungsschicht für die verschlüsselte Übertragung der Log-Daten zum Server implementiert werden müssen. Diese Menge von Regeln, die das Format und die Bedeutung der von den gleichgestellten Einheiten innerhalb einer Schicht ausgetauschten Pakete oder Nachrichten festlegt, ist ein Protokoll [4].

In der Abbildung 5.15 ist die Stellung dieses Datenfernübertragungs-Protokolls (DFÜ-Protokoll) im ISO-Modell dargestellt. Der Name Datenfernübertragungsprotokoll ist improvisiert. Die Datensicherheit der Übertragung wird sowohl durch die Verschlüsselung der versendeten Pakete inklusive die entsprechende DFÜ-Header mit AES gewährleistet.

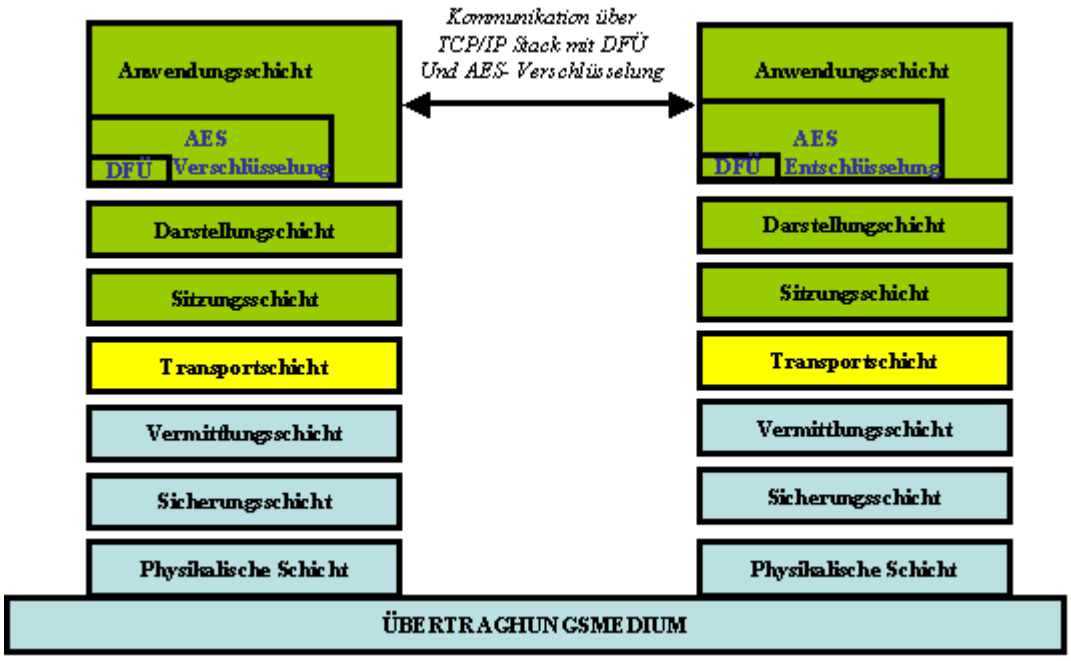


Abb. 5.15 Das Anwendungsschicht-Protokoll [DFÜ] im ISO-Modell

Die Bestandteile des DFÜ-Protokolls sind folgende:

DFÜ code	1 byte
DFÜ Header code	1 byte
DFÜ Data length	1 byte
DFÜ Data	0...to...250 bytes
DFÜ Stuffing	Restliche Daten bytes werden mit '0‘ ausgefühlt. [stuffing]

Das Paket-Format des DFÜ-Protokolls ist in der Abbildung 5.16 zu sehen. Der DFÜ-Code von 1 Byte wird für die Identifizierung beim Server verwendet. Die Header des Protokolls werden die Regeln der Kommunikation mit dem Server festlegen. Die Nutzdaten werden blockweise mit jeweils 250 Byte übertragen. Die Blockgröße ist auf 256 Byte festgelegt worden, da die Datenübertragung von der Patientenumgebung zum Telematik-Modul auch in 250 Blöcke zerlegt worden ist. Das Stuffing bedeutet nicht anderes als das Füllen vom letzten Paket mit Null, wenn die Nutzdaten im letzten Paket kleiner als 250 Byte sind.

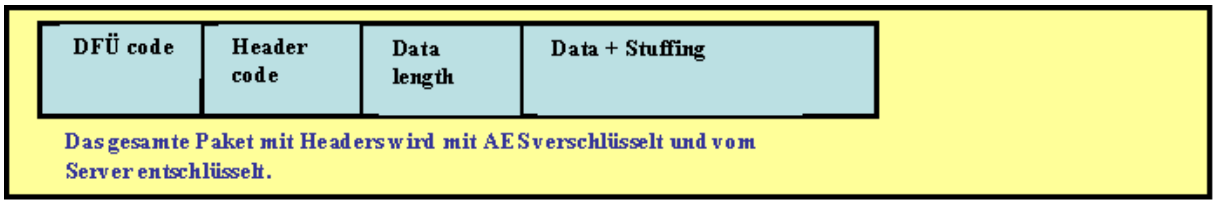


Abb. 5.16 Bestandteile des DFÜ-Protokolls

In der Software-Realisierung sind die Protokoll-Headers nicht anderes als Hash-Defines (Konstanten in C). Hier ist ein Ausschnitt der Definition von dem Source Code für das DFÜ-Protokoll, der für das Verständnis der Flussdiagramme und Arbeitsweise der Kommunikation notwendig ist:

```
#define DFÜ 88      //BHP protocol identity code

// protocol codes for DFÜ header

#define FILE_SEND_REQ 40    //request to send log file to server
#define FILE_SEND_ACK 41    //ACK for file send request to be received from server
#define FILE_SEND_NAK 42    //NAK for file send when the server/doctor is not ready.

#define FILE_DATA_SEND 50      //data packet sent to server
#define FILE_DATA_SEND_ACK 51   //ACK for conformed data packet to server
#define FILE_DATA_SEND_NAK 52   //NAK for error in data packet received
#define FILE_DATA_SEND_CONF 53   //data packet for conformation from server

#define FILE_GET_REQ 60   //request to server to send file [data: file name]
#define FILE_GET_ACK 61   //ACK for file request from server [data: file name& size]
#define FILE_GET_NAK 62   //NAK for file request when server cannot find file

#define FILE_DATA_GET 70      //data packet from server
#define FILE_DATA_GET_ACK 71   //data packet conformed by server
#define FILE_DATA_GET_NAK 72   //error in data packet sent by server
#define FILE_DATA_GET_CONF 73   //data packet sent to server for conformation
```

Um die Log-Daten zum Server übertragen zu können, stellt das Telematik-Modul mit dem klinischen Server über einen festgelegten Socket (Statische IP-Adresse plus Port) eine Internet-Verbindung her. Die mit AES verschlüsselte bidirektionale Kommunikation mit dem Server läuft über das DFÜ-Protokoll.

Bei jedem Request bzw. Replay des Telematik-Moduls (Client) handelt es sich um Senden oder Empfangen bestimmter Parameter bzw. Nutzdaten. Das Telematik-Modul muss das Paket-Format vom DFÜ-Protokoll und die Verschlüsselung des Paketes mit den Nutzdaten durchführen, bevor es zum Server überträgt. Diese Einkapselung vom DFÜ-Protokoll-Parameter und die Verschlüsselung bzw. Entschlüsselung des Paketes erfolgt gemäß in der Abbildung 5.17 dargestelltem Schema. Die Abbildung enthält das Schema in englischer Sprache, da die meisten Literaturquellen und das Sourcecode diesbezüglich in der englischen Sprache sind.

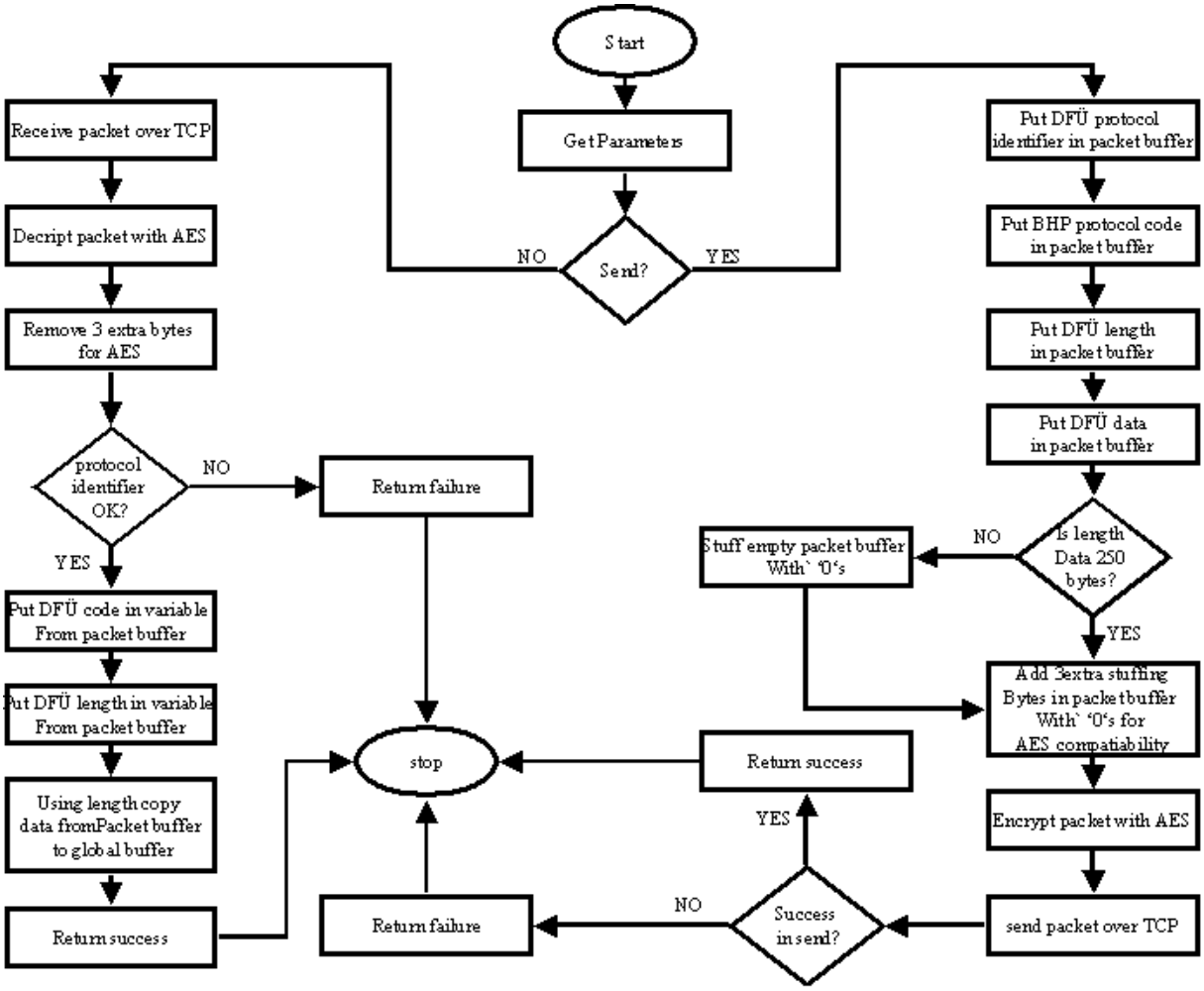


Abb. 5.17 Das Schema der Einkapselung vom DFÜ-Protokoll und die Verschlüsselung bzw. Entschlüsselung des Paketes.

Die Log-Daten werden nach dem in der Abbildung 5.18 dargestellten Ablauf zum klinischen Server übertragen. Das Telematik-Modul (Client) sendet seine Anforderung in dem DFÜ-Paket als z.B. hier FILE_SEND_REQ 40. Diese Anforderung bedeutet nicht anderes beim Server, als dass der Client die Log-Daten zum Server versenden will. Die Nutzdaten beim ersten Paket sind der Name der Log-Daten. Der Log-Daten-Name ist nicht anderes als die aktuelle Zeitangabe von der Echtzeituhr des Kern-Moduls.

Der Server antwortet mit einem NACK oder ACK. Falls die Reaktion vom Server FILE_SEND_ACK 41 lautet, überträgt das Telematik-Modul die ersten 250 Bytes Nutzdaten mit dem Header FILE_DATA_SEND 50 und wartet, bis der Server diese 250 Byte-Blocks wieder mit FILE_DATA_SEND_CONF 53 zurücksendet, damit sie byteweise beim Telematik-Modul verglichen werden können. Wenn der Vergleich erfolgreich durchgeführt wurde, schickt das Telematik-Modul ein FILE_DATA_SEND_ACK 51 mit den nächsten 250 Bytes Nutzdaten. Das würde so weiter laufen, bis die ganzen Log-Daten zum Server übertragen worden sind. In allen Fällen werden immer passende Nachrichten auf dem Display angezeigt, die Kommunikationsabbruch bzw. -erfolg symbolisieren. Wenn die Daten vollständig zum Server übertragen sind, wird die Verbindung zum Server und zur Internet beendet. Das Telematik-Modul kehrt dann zum Ausgangspunkt zurück.

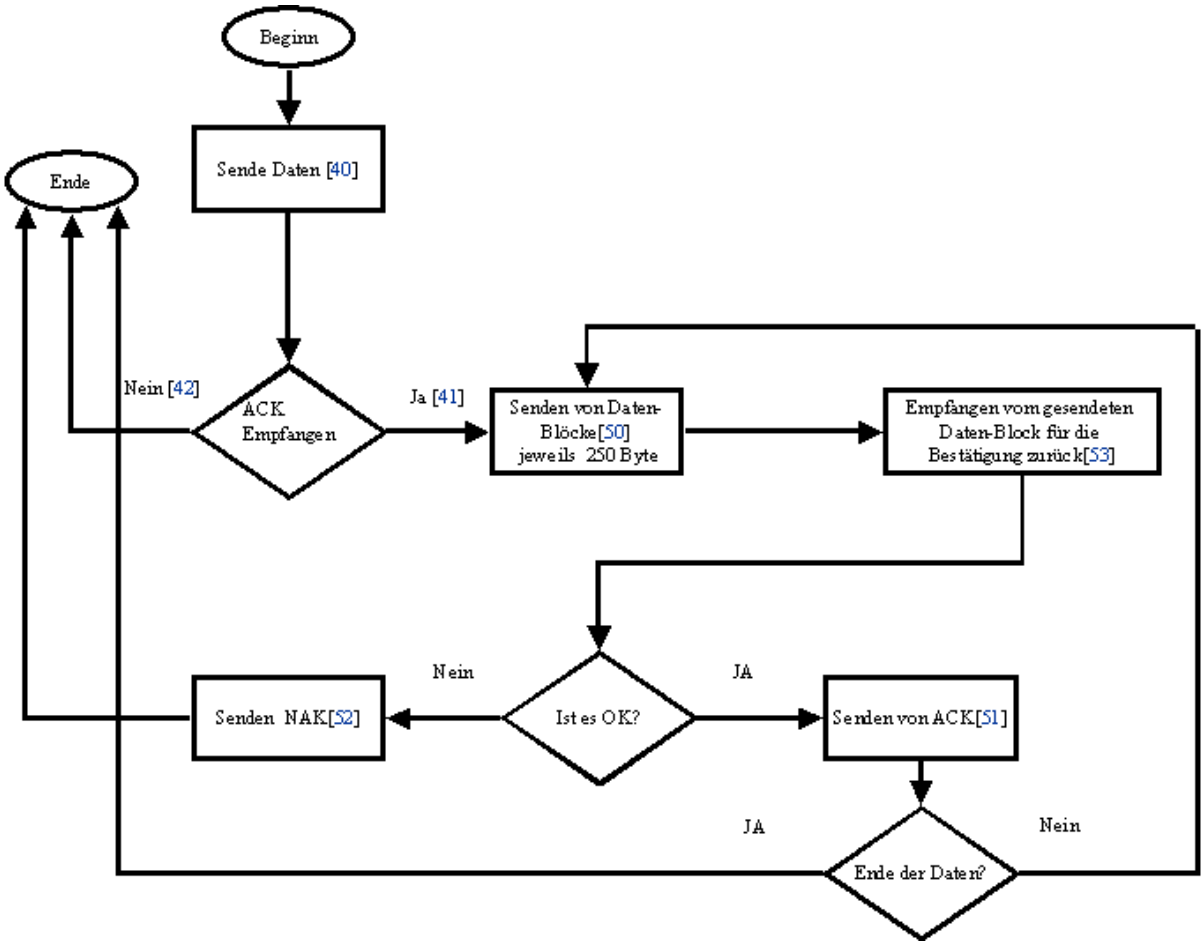


Abb. 5.18 Flussdiagramm der Kommunikation zwischen dem Telematik-Modul und dem Server

Die Arbeitsweise des Telematik-Moduls bei der Übertragung der Log-Daten zum File-Server wurde schon erklärt. Die Frage an der Stelle lautet: Wie verhält sich der Server?
Die Implementierung eines zentralen klinischen Servers sprengt den Rahmen dieser Diplomarbeit. Die Implementierung einer API, die mit einem Telematik-Modul über das DFÜ-Protokolls kommuniziert, wurde dagegen im Rahmen der Diplomarbeit erstellt. Das Werkzeug für die API ist KDeveloper unter Linux [8]. Die Abbildung 5.19 zeigt das Flussdiagramm der Serverseite bei der Kommunikation mit dem Telematik-Modul.
Die Server-Applikation ist immer in Wartezustand (Listen-Mode) und hat zwei Aufgaben. Diese sind der Empfang von Log-Daten vom Clients und das Senden der schon empfangenen Log-Daten zum Telematik-Modul abhängig von den Anfragen der Clients.

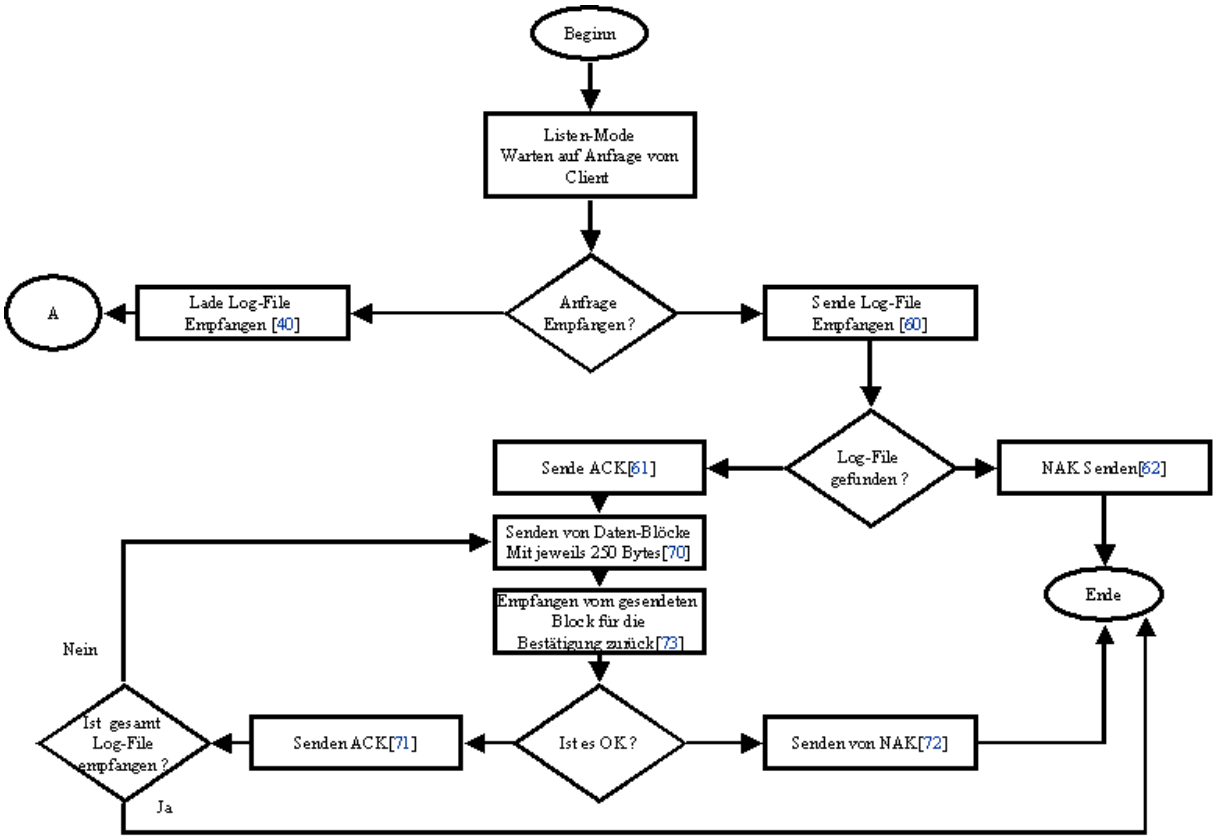


Abb.5.19 Flussdiagramm der Server-API bei der Kommunikation mit dem Telematik-Modul

In der Abbildung 5.19 wird sowohl die Abzweigung der API gemäß der Anfragen des Telematik-Moduls (Clients), als auch das Flussdiagramm der Server-API beim Senden von auf dem Server schon empfangenen und vorhandenen Log-Daten zum Telematik-Modul gezeigt. Wenn die Anfrage des Telematik-Moduls FILE_GET_REQ 60 lautet, verhält sich der Server in diesem Fall genau so ähnlich wie das Telematik-Modul (Client), wenn die Log-Daten zum Server mit dem DFÜ-Protokoll übertragen werden sollen.

Diese Funktion der Server-API wurde für Analyse und Testzwecke implementiert. Dazu gehört auch eine weitere Funktion des Telematik-Modul, wie das Laden einer der letzten 10 versendeten Log-Daten aus dem Server und das Weitersenden dieser Log-Daten an den angeschlossenen PC über die serielle Schnittstelle. Durch diese Funktionalität war es möglich, die gesendeten Log-Daten wieder aus dem Server zu empfangen, um sie dementsprechend an einen PC weiterzuleiten. Somit war es möglich, diese Log-Daten mit einer schon existierenden Software zu öffnen und zu lesen.

Falls die Anfrage des Clients FILE_SEND_REQ 40 lautet, bedeutet dies, dass der Server seine Bereitschaft zum Empfang von Log-Daten bestätigen muss. Der Server geht dann in den Modus A gemäß der Abbildung 5.20 und verhält sich genau so wie bei der Beschreibung der Übertragung der Log-Daten vom Telematik-Modul zum Server. Dabei spielt der Server die Empfängerrolle.

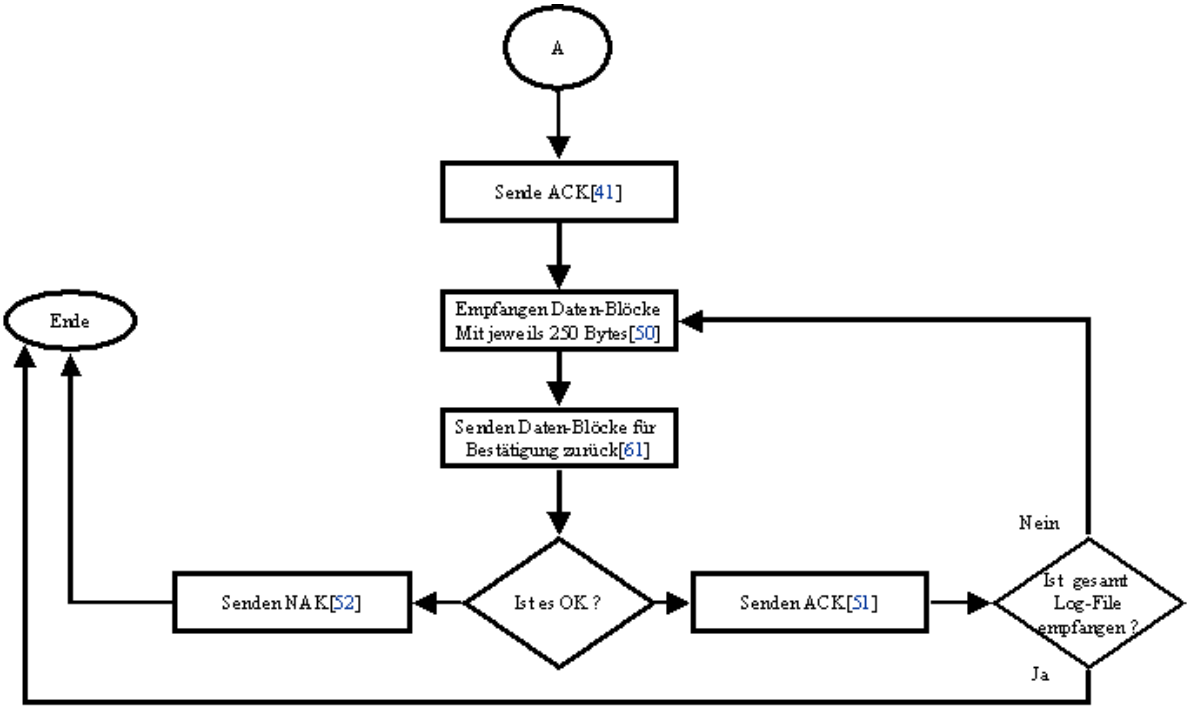


Abb. 5.20 Die Serverseite beim Empfang von Log-Daten

5.6 Probleme, Analyse und Schlussfolgerungen

Die Realisierung aller genannten Elemente der Überwachungstechnologie waren, sowohl in der Software als auch in der Hardware, nicht ohne kritische Entwicklungspunkte, die eine sehr schnelle Entscheidung benötigten, um die Entwicklung korrekt und fehlerfrei fortzuführen. Die Entwicklung erfolgte mit dem Entwicklungsboard RCM3200. Alle anderen Komponenten bzw. externen Module wurden durch Leitungen und Schnittstellen verbunden. Hatte die Software ein bestimmtes positives Stadium erreicht, wurde an einer Trägerplatine und der Auswahl von Embedded Modems gearbeitet.

Die Abbildung 5.21 zeigt die Trägerplatine mit allen Komponenten. Die Platine wurde mit vier Schichten (Layer) entwickelt. Auf dem Board sind noch eine GPRS- und Bluetooth-Schnittstellen für zukünftige Aufgaben (siehe Kapitel Sieben) integriert.

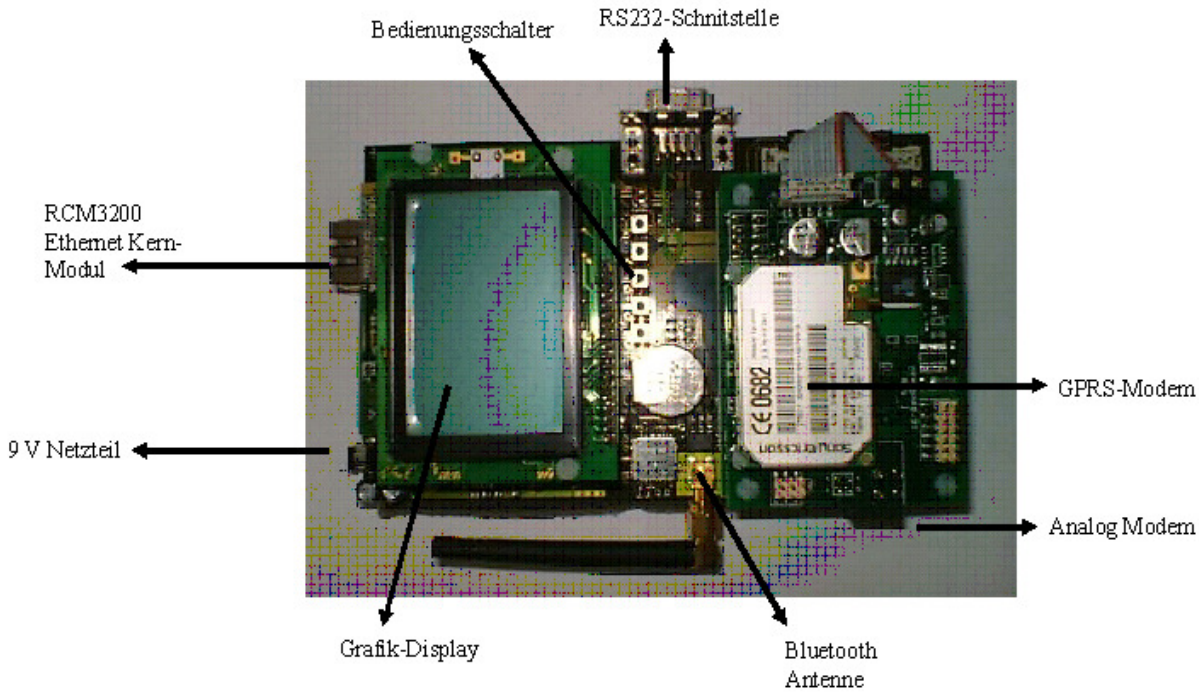


Abb.5.21 Die Trägerplatine vom Telematik-Modul

Die Abbildung 5.22 zeigt das Telematik-Modul als Prototyp für die klinischen Untersuchungen. 5.22. ist die Vorderseite des Telematik-Moduls, die das Display mit den drei Bedienungsschaltern beinhaltet. Unter dem Display sind die beiden Ja- bzw. OK-und Nein-Tasten zu sehen. Die Beschriftung der Tasten wurde durch die Software auf dem Grafik-Display integriert. Links oben in dem Bild ist eine GPRS-Antenne eingebettet, um eine zukünftige SMS-Notruf-Funktion zu entwickeln.

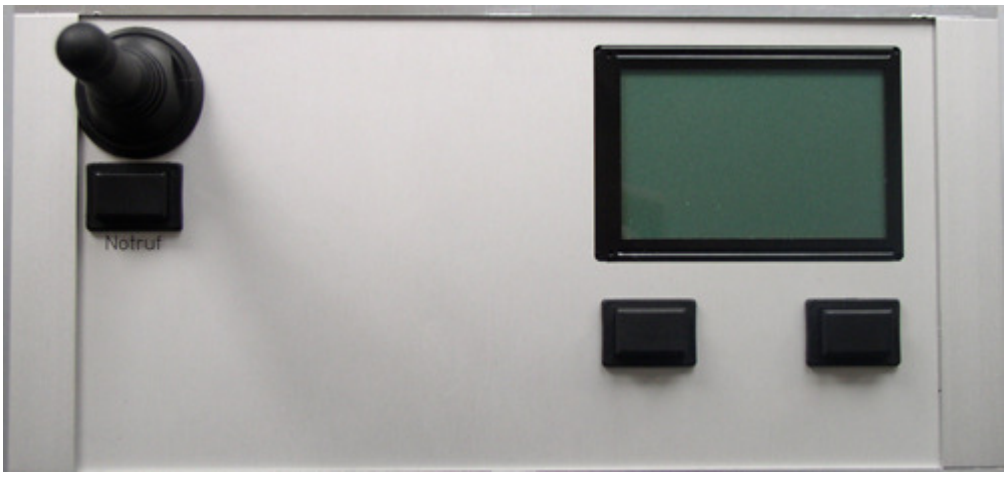


Abb. 5.22 Die Vorderseite des entwickelten Telematik-Moduls

Die Abbildung 5.23 zeigt die Rückseite des entwickelten Telematik-Moduls. Die Schnittstellen sind deutlich mit ihren Beschriftungen zu erkennen. Es sind zwei serielle SUB-D-Buchsen, 9 V Netzteil, Modem und Ethernet Anschlüsse. Eine Bluetooth-Antenne ist auch mit einem Bluetooth Chip für zukünftige Zwecke in dem Telematik-Modul integriert. Zwei weitere Bedienungstasten sind auf der Rückseite integriert worden. Die Servertaste dient dafür, dass das Telematik-Modul für die Online-Beobachtung in die Server-Mode-Funktion versetzt werden kann. Durch die Server-Mode-Taste wird es möglich sein, immer dasselbe Telematik-Modul mal als Client und mal als Server, je nach Einsatzbedarf und Einsatzort zu benutzen. Die Konfigurationstaste dient zur Konfiguration des Telematik-Moduls über seinen PC-Anschluss.

Abb. 5.23 Die Rückseite des entwickelten Telematik-Moduls

In diesem Abschnitt werden im Allgemeinen die Problematik und ihre Lösungen und anschließend die entsprechenden Schlussfolgerungen behandelt.

Bei der Entwicklung der Hardware ergaben oft Probleme. Es gab Probleme und Fehler bei der Anzeige von Text und Grafiken auf dem Grafik-Display, da bei der Entwicklung eine große Datenmengen über die seriellen Leitungen geschleust worden sind. Diese Datenmenge war größer als die vom Datenblatt festgelegte Größe von 63 Byte. Die serielle CTS & RTS-Leitungen wurden bei der Entwicklung kurzgeschlossen. Die Lösung dafür, nach Absprache des Online-Supports vom Hersteller, lag in der Software, indem softwaremäßig Verzögerungen (Delays) eingefügt wurden, wenn eine große Datenmenge über die serielle Leitung des Grafik-Displays übertragen werden sollte.

Bei der Software gab es im Vergleich zur Hardware viel mehr Änderungs- und Korrekturbedarfs. Allein die Entwicklung des DFÜ-Protokolls brachte einen großen Aufwand mit sich.
Die Fehler äußerten sich in geringfügigen Änderungen, angefangen von einer Abbruchbedienung in einer FOR-Schleife bis zur Änderungen einer Hash Define-Konstante.

Ein Administrationsfehler, der frühzeitig entdeckt wurde, war die Benutzung der mitgelieferten AES-Bibliothek von Dynamic C für die Verschlüsselung und Entschlüsselung der Daten. Die Zielsetzung der Entwicklung war die Datenübertragung zu einem Linux-File-Server. Diese mitgelieferte AES-Bibliothek enthält Assembler Code, das an das Rabbit-Kern-Modul angepasst wurde und nicht auf andere Systeme eingesetzt werden konnte. Die Lösung war der Einsatz der frei zur Verfügung stehenden Quellcode von AES von der NIST.

Dabei ergab sich ein weiteres Problem. Bei der Datenübertragung zum klinischen File-Server über das DFÜ-Protokoll waren die Pakete auf 253 festgelegt worden. Drei Bytes für das DFÜ-Protokoll und 250 Bytes waren die Nutzdaten. Bei der Verschlüsselung bzw. Entschlüsselung dieser Datenpakete mit dem verwendeten AES musste der DFÜ-Pakete auf drei weitere Bytes vergrößert werden, da das verwendete AES Code 16 Byte große Pakete verarbeiten kann. Damit aber die 256 Bytes DFÜ-Pakete mit dem AES Free Quellcode verschlüsselt bzw. entschlüsselt werden kann, wurden Änderungen am Quellcode benötigt.
Die Funktionen AESencrypt und AESdecrypt wurden für diese Anwendung implementiert und in den Quellcode des AES integriert. Der Quellcode beider Funktionen ist im Anhang zu finden.

Für die Analyse der Überwachungstechnologie sind dementsprechend verschiedene Tools und Geräte zum Einsatz gekommen. Die Analyse der Log-Datenübertragung und die Online-Beobachtung wurden mit einer Netzwerk-Schnüffel-Software durchgeführt. Des weiteren wurde das empfangene Log-File mit dem gesendeten verglichen, um die Übereinstimmung beider Files und damit die fehlerfreie Datenübermittlung zu überprüfen.

Das Programm Etherreal wurde für die Analyse der Pakete über das Internet sowohl für die Online-Beobachtung als auch für die Übertragung der Log-Daten zum File-Server verwendet. Etherreal protokolliert die gesamte Datenübertragung auf dem Ethernet in allen Protokollschichten und ist ein hervorragendes Mittel zur Analyse der Datenübertragung über das Netz. Die Protokollebenen lassen sich wie in einem Dateiverzeichnis des Win-Explores öffnen bzw. schließen. Das Programm ist auf der Internet Seite http://www.ethereal.com/ zum kostenlosen Download veröffentlicht.

Der Source-Code auf dem Telematik-Modul hat einschließlich aller eingefügten Bibliotheken eine Länge von 42588 Code-Zeilen. Im Anhang dieser Arbeit ist die an Dynamic C angepasste AES-Bibliothek zu finden. Bei der Softwareentwicklung war es aufgrund der besseren Lesbarkeit des Codes notwendig, die großen Module in Dynmaic C Bibliotheken zu programmieren. Dabei handelte es sich um die AES, Grafik-Display und die Konstanten.

6 Zusammenfassung

Vorgestellt wurde ein Prototyp einer Überwachungstechnologie von Patienten mit Herzunterstützungssystemen. Diese Arbeit befasste sich mit der Zielstellung, den Voraussetzungen, den fachlichen und rechtlichen Grundlagen, der Konzeption und der Realisierung dieser Technologie.

Dabei handelte sich im Wesentlichen um die Hardware- und Software-Entwicklung eines Telematik-Moduls, das sowohl eine Verbindung zur Patientenumgebung über die seriellen Schnittstellen darstellte, als auch die Herstellung einer Internet-Verbindung zu einem klinischen File-Server bzw. zum Clinical Affairs.

Es handelte sich um die Diagnose des Patienten mittels der Abfrage seines Herzunterstützungssystems, indem man die medizinischen Daten über den Zustand seines Herzunterstützungssystems direkt an einen klinischen File-Server (eine Überwachungszentrale im übertragenen Sinne) überträgt. Dort werden diese dann von Fachleuten diagnostiziert und bewertet, um entsprechende Maßnahmen durchzuführen.

Weiter betrachtete ich die Überwachung der Patienten durch eine direkte Punkt-zu-Punkt-Verbindung zwischen der Patientenumgebung und den Clinical Affairs, indem die Parameter des Herzunterstützungssystems durch die Online-Beobachtung direkt analysiert werden. Dann wird anhand der Diagnose bestimmter Parameter dieser Daten eine entsprechende Maßnahme für die Behandlung des Patienten abgeleitet und durchgeführt.

Meine Diplomarbeit hat gezeigt, dass eine Überwachung von Patienten mit Herzunterstützungssystemen technisch möglich ist. Aber bevor das vorgestellte Konzept und die Ergebnisse meiner Arbeit zum realen Einsatz an Patienten mit solchen Systemen umgesetzt werden können, müssen sowohl viele Voraussetzungen erfüllt, als auch viele kritische Punkte überwunden werden.

Unter den Voraussetzungen zählen die Bereitschaft des Patienten, seine Ausbildung und die Absprache zwischen der Patientenumgebung und den Clinical Affairs. Die kritischen Punkte auf der anderen Seite gehören zur selben Kategorie aller Internet-Anwendungen. Dabei handelt es sich um die Verfügbarkeit und Zuverlässigkeit der verwendeten Infrastruktur. Der Einfluss dieser kritischen Punkte auf die Überwachungstechnologie hängt sehr stark davon ab, wo und wie die Beteiligten am Netzwerk miteinander verbunden sind. Ein Beispiel dafür wäre der Vergleich der Datenübertragung mittels analogem Modem und Ethernet. Die Kommunikation zwischen den Teilnehmern mit Modems läuft im Wesentlichen langsamer als die Kommunikation mittels Ethernet (ISDN).

Die vorgestellte Überwachungstechnologie meiner Arbeit wird an die zurzeit in den klinischen Untersuchungen befindlichen Herzunterstützungssystemen der Firma Berlin Heart AG eingesetzt und auf Dauer getestet. Es werden viele Integrationen in der Zukunft dazu addiert und implementiert werden müssen, bis das Überwachungsnetzwerk an Patienten angewendet werden kann.

7 Ausblick

7.1 Zukünftige Entwicklungen an die Überwachungstechnologie

Nachdem das Konzept und die Realisierung einer Überwachungstechnologie als Prototyp für die klinischen Untersuchungen in dieser Diplomarbeit behandelt wurde, sollen in diesem Abschnitt die zukünftigen Schritte einer Weiterentwicklung der Überwachungstechnologie genannt werden.

Die API des File-Servers ist im Rahmen meiner Diplomarbeit lediglich für den Empfang von Log-Daten aus einem einzigen Telematik-Modul (Client) implementiert worden. Für den zukünftigen Einsatz des Servers muss man mit mehreren Clients rechnen, die gleichzeitig einen Zugriff auf den Server beanspruchen. Aus diesen Anforderungen ergibt sich die Notwendigkeit der Implementierung einer Datenbank und eines Web-Servers, um die empfangenen Log-Daten gemäß ID, Datum und Uhrzeit abzuspeichern und um die über eine mit SSL gesicherte Internet-Verbindung über Webbrowser dem Clinical Affairs zur Verfügung zu stellen.

Es ergeben sich folgende Anforderungen an einen derartigen multi-userfähigen Server:

1. Der Server muss die Möglichkeit bieten, dass mehrere Verbindungen zu ihm aufgebaut werden können.
2. Der Server sollte rund um die Uhr erreichbar sein.
3. Auf dem Server muss eine Datenbank integriert sein, welche die Daten zu Patienten, Ärzten und Clinical Affairs erfasst und in Verbindung miteinander bringt. Eine relationale Datenbank sollte für den Gebrauch ausreichend Funktionalität mit sich bringen.
4. Das Serversystem muss Mechanismen enthalten, welche einen unerlaubten Zugriff unterbinden.
5. Es ist ein http-Server zu integrieren, um den Benutzern die Möglichkeit zu geben, Daten einzusehen und bestimmte Aktionen durchzuführen.
6. Der Server dient neben dem Speichern der Log-Daten und Benutzerdaten ebenfalls zur Dokumentation für die Vorgänge von Benutzern. Der Benutzer muss je nach Status verschiedene Masken vom http-Server erhalten.
7. Es gibt drei verschiedene Möglichkeiten des Status eines Benutzers. Diese sind einzuteilen in Patient, Arzt und Clinical Affairs.
8. Jeder Benutzer muss eindeutig identifiziert werden können und sich durch ein Passwort authentifizieren.
9. Jeder Patient muss einem Arzt sowie einem medizinischen Herzunterstützungssystem zugewiesen werden.
10. Eine Herzunterstützungssystemsnummer kann nur einmal im Gebrauch bzw. vergeben sein.
11. Ein Arzt kann mehrere Patienten behandeln.
12. Jeder Arzt ist einem Krankenhaus zuzuweisen.
13. Von jedem Benutzer müssen seine Telefonnummer, seine Adresse und sein Name hinterlegt sein.
14. Log-Daten werden nach der medizinischen Gerätenummer, dem Datum und der Uhrzeit abgelegt.
15. Log-Daten sind einem Patienten zuzuweisen.

16. Das Clinical Affairs-Personal kann Log-Daten von Patienten herunterladen und einsehen.
17. Auf dem Server soll eine Statistik integrieret werden, die die Analyse aller Funktionen der gesamten Überwachungstechnologie ermöglichen soll. Folgende Anforderungen sind an eine solche Statistik zu stellen:

    - Es soll anhand der Herzunterstützungssystemsnummer immer bei jedem Log-Datenempfang ein Eintrag in die Statistik entstehen.
    - Die Überprüfung der Patientenbereitschaft durch seine tägliche Mitwirkung beim Versenden der Log-Daten bezüglich seines Herzunterstützungssystems.
    - Die Überprüfung der Bereitschaft bzw. Teilnahme von Clinical Affairs durch ihre tägliche Mitwirkung beim Überprüfen der Log-Daten vom Server bzw. von der Datenbank bezüglich ihrer Patienten mit Herzunterstützungssystemen.
    - SMS-Versand bzw. E-Mails sollen dokumentiert werden
    - Die Statistik soll in Grafiken dargestellt werden.
    - Die Grafiken sollen verschiedene Achsen-Bezeichnungen haben, je nach Bedarf der Statistik.
    - Auf dem Server neu eingehende Patientendaten werden automatisch mit dem aktuellen Datum und Uhrzeit versehen und an den Datenbestand angehängt. Die eindeutige Identifizierung der Daten erfolgt anhand der Herzunterstützungssystemsnummer (Controller-ID).

Die weitere Entwicklung der Überwachungstechnologie betrifft nicht nur die Implementierung restlicher Komponenten des File-Servers, sondern die Implementierung der Bluetooth und GPRS-Strecken.

Die Hardware-Implementierung des Telematik-Moduls war mit Absicht in der Form entwickelt worden, dass eine Bluetooth-Schnittstelle und eine Schnittstelle für das GPRS-Modem in das Design integriert wurden, da beide Schnittstellen für zukünftige Zwecke der Überwachungstechnologie dienen.

Die Idee hinter der Bluetooth-Schnittstelle ist die, dass man die Kommunikation zur Patientenumgebung in einer einfacheren Form gestallten will. In dem Fall handelt es sich um eine drahtlose Verbindung zwischen dem Herzunterstützungssystem und dem Telematik-Modul. Das setzt voraus, dass das zu überwachende System schon so einen Schnittstellen-Treiber bzw. Anschluss besitzt.

Die Variante mit dem GPRS-Modem hängt auch sehr stark mit der drahtlosen Verbindung zur Patientenumgebung zusammen. Dabei beschränkt sich die Funktion des GPRS-Modems auf einen Notfall-Dienst. Es ist in der Form gedacht, dass beim Betätigen einer Notfalltaste bestimmte Parameter des Herzunterstützungssystems drahtlos über die Bluetooth-Schnittstelle abgefragt werden. Diese Parameter werden in eine SMS eingekapselt und an die Überwachungszentrale bzw. an die Clinical Affairs gesendet. Diese zwei weiteren Implementierungen sind von großer Bedeutung, da von der Benutzerfreundlichkeit des Telematik-Moduls der Erfolg der Überwachungstechnologie abhängt. Weiterhin muss immer an die Ausbildung und die Bereitschaft der Patienten und aller Beteiligten der Überwachungstechnologie in jedem Entwicklungsschritt geachtet werden.

7.2 Konzeption einer Studie für Patienten mit Herzunterstützungssystemen

Um die Ergebnisse dieser Arbeit auf Dauer evaluieren und testen zu können, wird eine Studie benötigt. Mit dieser sollen verschiedene Aspekte der Überwachungstechnologie behandelt und berücksichtigt werden. Dabei werden die regelmäßige Übermittlung der Patientendaten und die Bereitschaft der Patienten zur kontinuierlichen Mitarbeit überprüft.

Das angestrebte Resultat dieser Studie ist im Allgemeinen ähnlich wie alle anderen Studien für die Überwachung von Patienten mit Herzimplantaten. Es handelt sich um die Verkürzung der Frühhospitalphase, um die Verminderung von Mortalität und Morbidität zu erreichen. Aber auch unnötige Krankenhauseinweisungen bzw. Krankenhausaufenthalte werden u. U. vermieden, was wiederum Kosteneinsparungen bewirken könnte. Dabei stellt die Technologie, die in dieser Arbeit behandelt wurde, eine wichtige wesentliche Voraussetzung für die Umsetzung dieser Studie dar. Wie in den Abbildungen zu sehen war, müssen mehrere Teilnehmer ans Netzwerk angeschlossen werden, um eine erfolgreiche Datenübertragung und -auswertung zu erzielen.

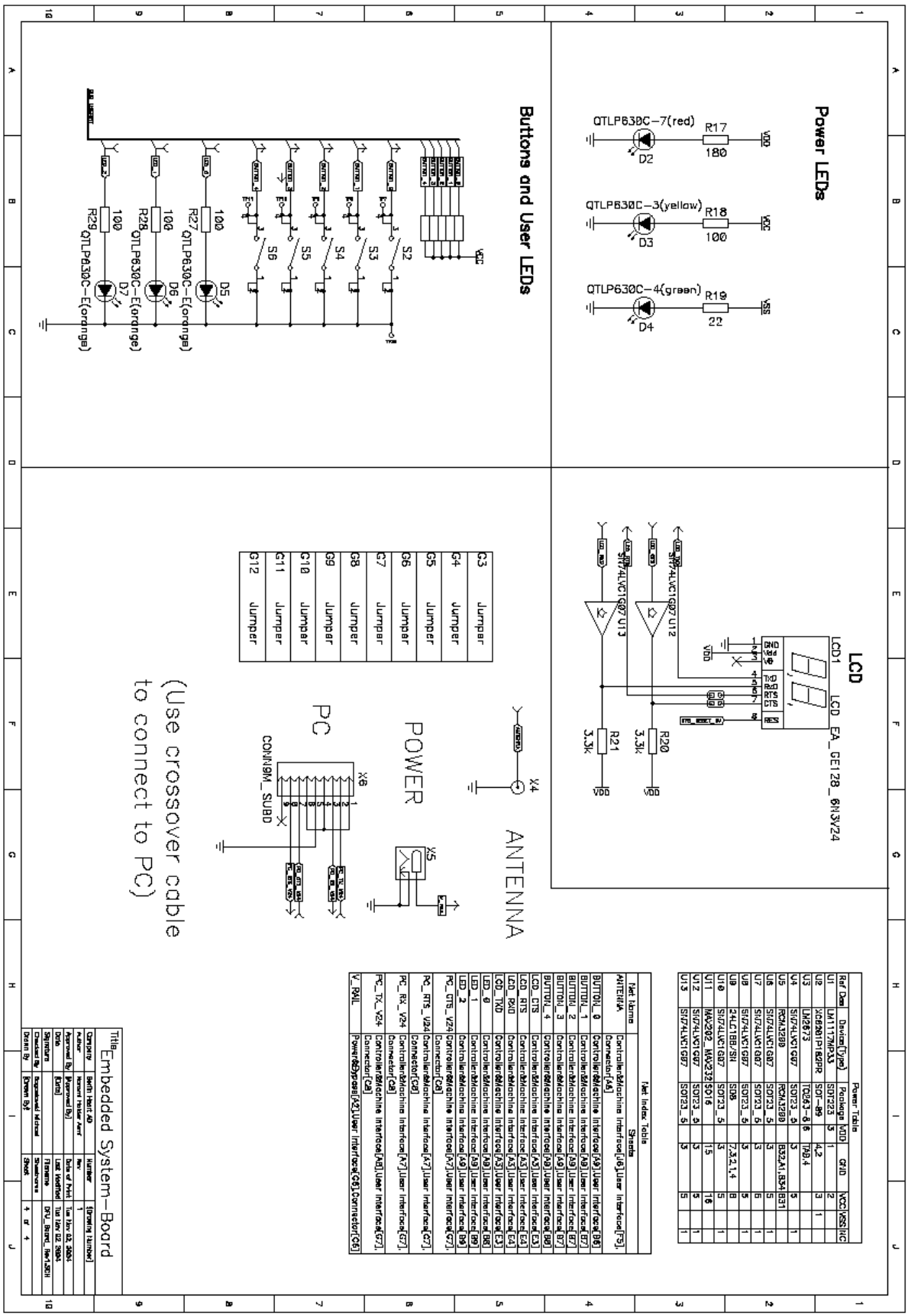
Die an der Studie teilnehmenden Patienten werden verpflichtet, ihre Log-Daten täglich zu übermitteln. Der Empfang der Log-Daten wird beim klinischen Server registriert. Bei jedem Empfang erfolgt ein Eintrag in die Statistik, abhängig vom Datum und Patientennamen. Durch die regelmäßige Kontrolle der Übertragung der Log-Daten wird sowohl die Bereitschaft der Patienten als auch die einwandfreie technische Funktionsweise kontrolliert. Wenn Patienten die Log-Daten für mehr als zwei bis drei Tage nicht versenden, wird dies beim klinischen Server registriert. Diese Patienten werden entsprechend benachrichtigt. Wenn Patienten nicht mehr zur Datenübermittlung bereit sind, werden sie von der Studie nicht mehr berücksichtigt.

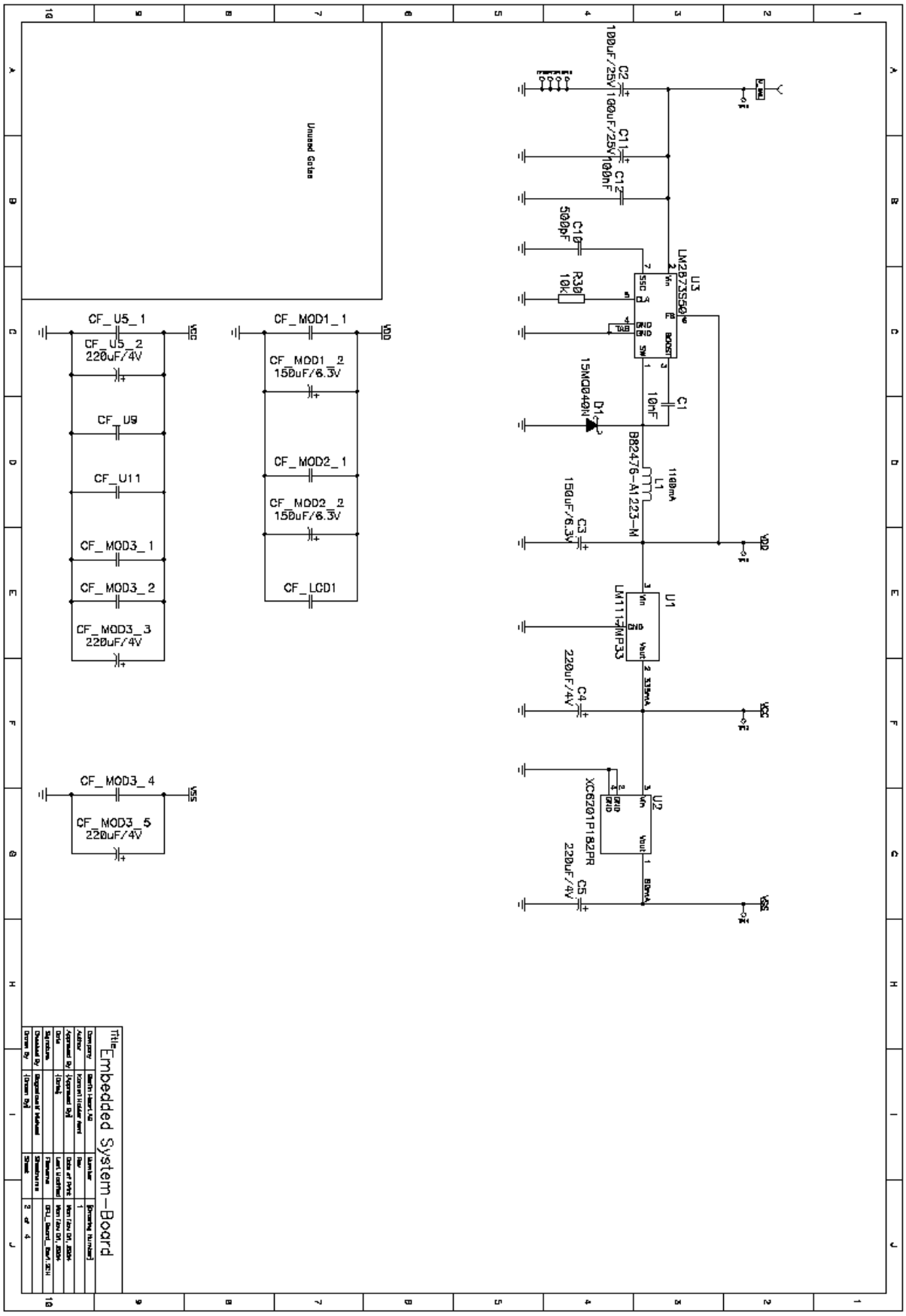
Die Ergebnisse dieser Diplomarbeit haben gezeigt, dass die automatische Überwachung von Patienten technisch möglich ist. Zudem wird deutlich, dass auf die Bereitschaft und Mitarbeit des Patienten zwar nicht vollständig verzichtet werden kann, die diesbezüglichen Anforderungen an die Kooperationsbereitschaft jedoch vergleichsweise nicht gering sind. Zudem erhält der Patient jedoch durch seine Mitwirkung die Möglichkeit, mehr Informationen über seinen Krankheitszustand und Therapieprozess zu bekommen, was wieder zu einem besseren Verständnis und zu einer höheren Zufriedenheit beim Patienten führen kann. Durch die regelmäßige Abfrage und Auswertung der Herzunterstützungssysteme kann zu einer verbesserten Therapieführung von Patienten mit implantierten Pumpen beitragen.

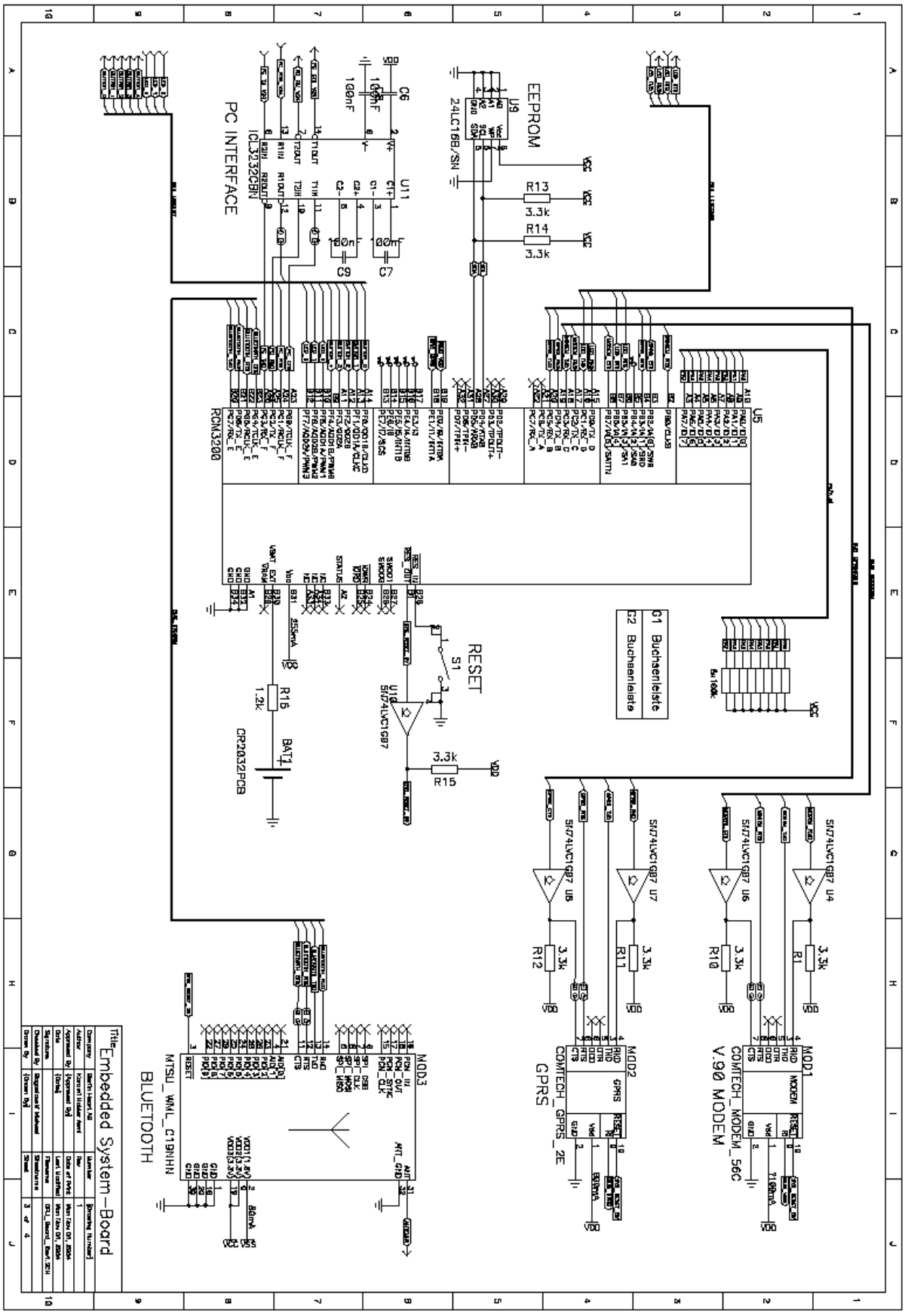
Diese Arbeit stellte die Grundlage einer Überwachungstechnologie vor. Die angestrebten Resultate dieser Studie würden dazu beitragen, dass mehre Schlussfolgerungen über den Erfolg der Ergebnisse dieser Arbeit gezogen werden. Es können dadurch neue Erfahrungen in diesem Bereich gesammelt werden, die den zukünftigen Einsatz einer Technologie dieser Art vereinfacht.

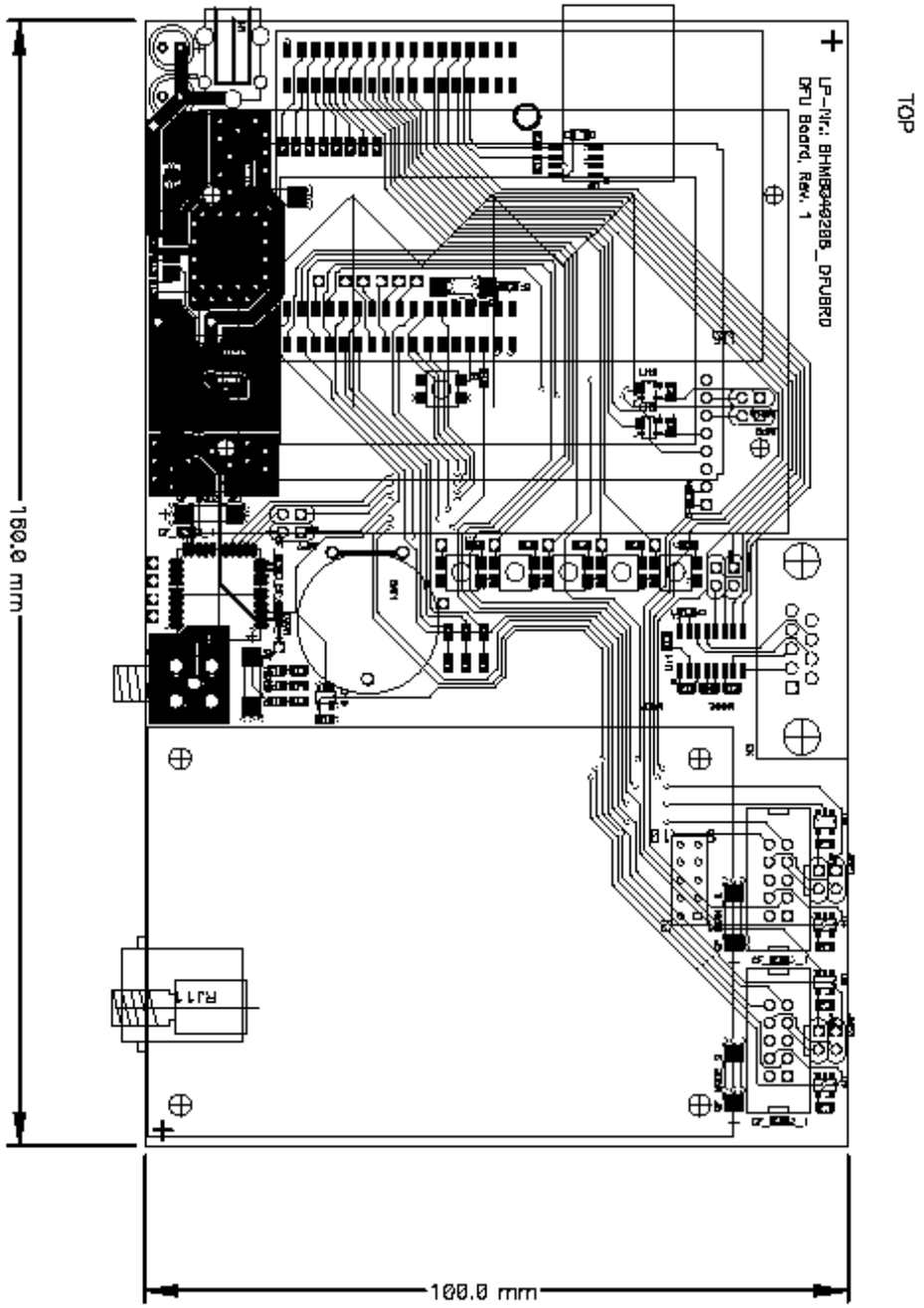
8 Anhang

8.1 Schaltungsdesign




Unused Gates
CF_U5_1
CF_U5_2
220uF/4V
CF_U9
CF_U11
CF_MOD3_1
CF_MOD3_2
CF_MOD3_3
220uF/4V
CF_MOD1_1
CF_MOD1_2
150uF/6.3V
CF_MOD2_1
CF_MOD2_2
150uF/6.3V
CF_LCD1
CF_MOD3_4
CF_MOD3_5
220uF/4V
VCC
VDD
VSS
C2
100uF/25V
C11
100uF/25V
C12
100nF
C10
500pF
R30
10k
U3
LM2623SD
C1
10nF
D1
15MQ040N
B82476-A1223-M
L1
1100mA
C3
150uF/6.3V
U1
LM1117MP33
C4
220uF/4V
U2
XC6201P182PR
C5
220uF/4V
Title
Embedded System-Board
Company
Author
Approved By
Date
Checked By
Revision
Drawn By
Sheet
Number
Rev
Date of Print
Last Modified
File Name
Drawing Number
2 of 4


PC INTERFACE
ICL3232CBN
EEPROM
24LC16B/SN
RCM3200
RESET
S1
BAT1
CR2032PCB
R15
3.3k
R16
1.2k
GPRS
COMTECH_GPRS_2E
V.90 MODEM
COMTECH_MODEM_56C
BLUETOOTH
MTSU_WML_C19NHN
G1 Buchsenleiste
G2 Buchsenleiste
Embedded System-Board



NOTE: GPRS-Module and MODEM share the PCB area and use the same mounting holes. GPRS-Module is mounted over MODEM

NOTE: CORE-Module and LCD share the PCB area. Mount LCD over CORE-Module

8.2 Quellcode von AES Bibliothek für Dynamic C

```
/**********************//#030877#//
BHP_AES_CRYPT.LIB

Implementation of the 16-byte-block based Rijndael AES cipher with 128 bit Key
for Berlin-Heart-Protocol packets of 256 bytes.

For more information on this AES implementation please refer to:
"A Specification for Rijndael, the AES Algorithm" by-Joan Daemen & Vincent Rijmen.

Acknowledgement - Brian Gladman  (http://fp.gladman.plus.com)

**********************/

/*** BeginHeader */
#ifndef __BHP_AES_CRYPT_LIB
#define __BHP_AES_CRYPT_LIB
/*** EndHeader */

/*** BeginHeader */
//AES defines
#define n_row 4 // the number or rows in the state for AES
#define n_col 4 // the number or columns in the state for AES
#define n_maxr 14  // the maximum number of cipher rounds for AES

//AES global vars:
typedef unsigned char aes_elem; // a finite field element in GF(256) for AES
typedef aes_elem aes_col[4]; // a column of four GF(256) elements for AES
typedef aes_col aes_state[4];// an array of columns for the state for AES
aes_state key_sch[15]; // the key schedule for AES
int key_len; //key length for AES

const aes_elem s_box[256] = // the S box
{
0x63, 0x7c, 0x77, 0x7b, 0xf2, 0x6b, 0x6f, 0xc5,
0x30, 0x01, 0x67, 0x2b, 0xfe, 0xd7, 0xab, 0x76,
0xca, 0x82, 0xc9, 0x7d, 0xfa, 0x59, 0x47, 0xf0,
0xad, 0xd4, 0xa2, 0xaf, 0x9c, 0xa4, 0x72, 0xc0,
0xb7, 0xfd, 0x93, 0x26, 0x36, 0x3f, 0xf7, 0xcc,
0x34, 0xa5, 0xe5, 0xf1, 0x71, 0xd8, 0x31, 0x15,
0x04, 0xc7, 0x23, 0xc3, 0x18, 0x96, 0x05, 0x9a,
0x07, 0x12, 0x80, 0xe2, 0xeb, 0x27, 0xb2, 0x75,
0x09, 0x83, 0x2c, 0x1a, 0x1b, 0x6e, 0x5a, 0xa0,
0x52, 0x3b, 0xd6, 0xb3, 0x29, 0xe3, 0x2f, 0x84,
0x53, 0xd1, 0x00, 0xed, 0x20, 0xfc, 0xb1, 0x5b,
0x6a, 0xcb, 0xbe, 0x39, 0x4a, 0x4c, 0x58, 0xcf,
0xd0, 0xef, 0xaa, 0xfb, 0x43, 0x4d, 0x33, 0x85,
0x45, 0xf9, 0x02, 0x7f, 0x50, 0x3c, 0x9f, 0xa8,
0x51, 0xa3, 0x40, 0x8f, 0x92, 0x9d, 0x38, 0xf5,
0xbc, 0xb6, 0xda, 0x21, 0x10, 0xff, 0xf3, 0xd2,
```

```
0xcd, 0x0c, 0x13, 0xec, 0x5f, 0x97, 0x44, 0x17,
0xc4, 0xa7, 0x7e, 0x3d, 0x64, 0x5d, 0x19, 0x73,
0x60, 0x81, 0x4f, 0xdc, 0x22, 0x2a, 0x90, 0x88,
0x46, 0xee, 0xb8, 0x14, 0xde, 0x5e, 0x0b, 0xdb,
0xe0, 0x32, 0x3a, 0x0a, 0x49, 0x06, 0x24, 0x5c,
0xc2, 0xd3, 0xac, 0x62, 0x91, 0x95, 0xe4, 0x79,
0xe7, 0xc8, 0x37, 0x6d, 0x8d, 0xd5, 0x4e, 0xa9,
0x6c, 0x56, 0xf4, 0xea, 0x65, 0x7a, 0xae, 0x08,
0xba, 0x78, 0x25, 0x2e, 0x1c, 0xa6, 0xb4, 0xc6,
0xe8, 0xdd, 0x74, 0x1f, 0x4b, 0xbd, 0x8b, 0x8a,
0x70, 0x3e, 0xb5, 0x66, 0x48, 0x03, 0xf6, 0x0e,
0x61, 0x35, 0x57, 0xb9, 0x86, 0xc1, 0x1d, 0x9e,
0xe1, 0xf8, 0x98, 0x11, 0x69, 0xd9, 0x8e, 0x94,
0x9b, 0x1e, 0x87, 0xe9, 0xce, 0x55, 0x28, 0xdf,
0x8c, 0xa1, 0x89, 0x0d, 0xbf, 0xe6, 0x42, 0x68,
0x41, 0x99, 0x2d, 0x0f, 0xb0, 0x54, 0xbb, 0x16
};
const aes_elem is_box[256] = // the inverse S box
{
0x52, 0x09, 0x6a, 0xd5, 0x30, 0x36, 0xa5, 0x38,
0xbf, 0x40, 0xa3, 0x9e, 0x81, 0xf3, 0xd7, 0xfb,
0x7c, 0xe3, 0x39, 0x82, 0x9b, 0x2f, 0xff, 0x87,
0x34, 0x8e, 0x43, 0x44, 0xc4, 0xde, 0xe9, 0xcb,
0x54, 0x7b, 0x94, 0x32, 0xa6, 0xc2, 0x23, 0x3d,
0xee, 0x4c, 0x95, 0x0b, 0x42, 0xfa, 0xc3, 0x4e,
0x08, 0x2e, 0xa1, 0x66, 0x28, 0xd9, 0x24, 0xb2,
0x76, 0x5b, 0xa2, 0x49, 0x6d, 0x8b, 0xd1, 0x25,
0x72, 0xf8, 0xf6, 0x64, 0x86, 0x68, 0x98, 0x16,
0xd4, 0xa4, 0x5c, 0xcc, 0x5d, 0x65, 0xb6, 0x92,
0x6c, 0x70, 0x48, 0x50, 0xfd, 0xed, 0xb9, 0xda,
0x5e, 0x15, 0x46, 0x57, 0xa7, 0x8d, 0x9d, 0x84,
0x90, 0xd8, 0xab, 0x00, 0x8c, 0xbc, 0xd3, 0x0a,
0xf7, 0xe4, 0x58, 0x05, 0xb8, 0xb3, 0x45, 0x06,
0xd0, 0x2c, 0x1e, 0x8f, 0xca, 0x3f, 0x0f, 0x02,
0xc1, 0xaf, 0xbd, 0x03, 0x01, 0x13, 0x8a, 0x6b,
0x3a, 0x91, 0x11, 0x41, 0x4f, 0x67, 0xdc, 0xea,
0x97, 0xf2, 0xcf, 0xce, 0xf0, 0xb4, 0xe6, 0x73,
0x96, 0xac, 0x74, 0x22, 0xe7, 0xad, 0x35, 0x85,
0xe2, 0xf9, 0x37, 0xe8, 0x1c, 0x75, 0xdf, 0x6e,
0x47, 0xf1, 0x1a, 0x71, 0x1d, 0x29, 0xc5, 0x89,
0x6f, 0xb7, 0x62, 0x0e, 0xaa, 0x18, 0xbe, 0x1b,
0xfc, 0x56, 0x3e, 0x4b, 0xc6, 0xd2, 0x79, 0x20,
0x9a, 0xdb, 0xc0, 0xfe, 0x78, 0xcd, 0x5a, 0xf4,
0x1f, 0xdd, 0xa8, 0x33, 0x88, 0x07, 0xc7, 0x31,
0xb1, 0x12, 0x10, 0x59, 0x27, 0x80, 0xec, 0x5f,
0x60, 0x51, 0x7f, 0xa9, 0x19, 0xb5, 0x4a, 0x0d,
0x2d, 0xe5, 0x7a, 0x9f, 0x93, 0xc9, 0x9c, 0xef,
0xa0, 0xe0, 0x3b, 0x4d, 0xae, 0x2a, 0xf5, 0xb0,
0xc8, 0xeb, 0xbb, 0x3c, 0x83, 0x53, 0x99, 0x61,
0x17, 0x2b, 0x04, 0x7e, 0xba, 0x77, 0xd6, 0x26,
0xe1, 0x69, 0x14, 0x63, 0x55, 0x21, 0x0c, 0x7d
```

```
};

const aes_elem ff_tab[8] =
{
                0x00, 0x1b, 0x36, 0x2d, 0x6c, 0x77, 0x5a, 0x41
};

/*** EndHeader */

/*** BeginHeader AESmakeKey */
void AESmakeKey(void* key);
/*** EndHeader */

/* START FUNCTION DESCRIPTION
*********************************************
AESmakeKey                    <BHP_AES_CRYPT.LIB>

SYNTAX:                       void AESmakeKey(void* key);

KEYWORDS:                     AES, key, aes_col, aes_elem, aes_state, key_sch, BHP.

DESCRIPTION:        This function makes cipher key using the string passed to it.
            Acknowledgement - Brian Gladman  (http://fp.gladman.plus.com)

PARAMETER1:         16 byte(unsigned char) string to be used to make the key.

RETURN VALUE:  none

SEE ALSO:      AESmakeKey, AESdecryptPacket, AESencryptPacket

END DESCRIPTION
*********************************************************/
// set the cipher key
void AESmakeKey(void* key)
{
  int hi,i,k;
  aes_col *kp;
  aes_elem rc;
  aes_col temp;

  key_len = 128 / 32;
  kp = (aes_col*)key_sch;
        hi = n_col * (key_len + 7);
        i = -1, k = 0;
        while(++i < key_len)
        {
                kp[i][0] = ((unsigned char *)key)[k++];
                kp[i][1] = ((unsigned char *)key)[k++];
                kp[i][2] = ((unsigned char *)key)[k++];
                kp[i][3] = ((unsigned char *)key)[k++];
        }
```

```
        --i;
        rc = 1;
        while(++i < hi)
        {
        //temp = { kp[i - 1][0], kp[i - 1][1], kp[i - 1][2], kp[i - 1][3] };
    temp[0] = kp[i - 1][0];
    temp[1] = kp[i - 1][1];
    temp[2] = kp[i - 1][2];
    temp[3] = kp[i - 1][3];

                if(i % key_len == 0)
                {
                        rot_column(temp);
                        sub_column(temp);
                        temp[0] ^= rc;
                        rc = gf_mul2(rc);
                }
                else if (key_len > 6 && (i % key_len == 4))
                        sub_column(temp);
                kp[i][0] = kp[i - key_len][0] ^ temp[0];
                kp[i][1] = kp[i - key_len][1] ^ temp[1];
                kp[i][2] = kp[i - key_len][2] ^ temp[2];
                kp[i][3] = kp[i - key_len][3] ^ temp[3];
        }
}
```

```
/*** BeginHeader AESencryptPacket */
int AESencryptPacket(unsigned char *data);
/*** EndHeader */

/* START FUNCTION DESCRIPTION
*********************************************
AESencryptPacket              <BHP_AES_CRYPT.LIB>

SYNTAX:                       int AESencryptPacket(unsigned char *data);

KEYWORDS:                     AES, encrypt, BHP.

DESCRIPTION:        This function encrypts the 256 byte packet passed to it in place.

PARAMETER1:         256 byte(unsigned char) packet string.

RETURN VALUE:  SUCCESS / FAILURE code i.e. 0 / 1

SEE ALSO:      AESmakeKey, AESdecryptPacket, AESencryptPacket

END DESCRIPTION
*********************************************************/
int AESencryptPacket(unsigned char *data)
{
        int i,j,k;
  unsigned char indata[17];
  unsigned char outdata[17];
  j=0;
  k=0;
  while(k<256)
  {
        memset(indata, '\0', 17);
        memset(outdata, '\0', 17);
        for(i=0;i<16 && j<256;i++,j++)
                {
                indata[i]=data[j];
                }
        if(encrypt(indata, outdata)==FAILURE)
        {
                return FAILURE;
        }
        for(i=0;i<16 && k<256;i++,k++)
                {
                data[k]=outdata[i];
                }
  }
  return SUCCESS;
}
```

```
/*** BeginHeader AESdecryptPacket */
int AESdecryptPacket(unsigned char *data);
/*** EndHeader */

/* START FUNCTION DESCRIPTION
*********************************************
AESdecryptPacket              <BHP_AES_CRYPT.LIB>

SYNTAX:                       int AESdecryptPacket(unsigned char *data);

KEYWORDS:                     AES, decrypt, BHP.

DESCRIPTION:        This function decrypts the 256 byte packet passed to it in place.

PARAMETER1:         256 byte(unsigned char) packet string.

RETURN VALUE:  SUCCESS / FAILURE code i.e. 0 / 1

SEE ALSO:      AESmakeKey, AESdecryptPacket, AESencryptPacket

END DESCRIPTION
**********************************************************/
int AESdecryptPacket(unsigned char *data)
{
        int i,j,k;
  unsigned char indata[17];
  unsigned char outdata[17];
  j=0;
  k=0;
  while(k<256)
  {
        memset(indata, '\0', 17);
        memset(outdata, '\0', 17);
        for(i=0;i<16 && j<256;i++,j++)
                {
                indata[i]=data[j];
                }
        if(decrypt(indata, outdata)==FAILURE)
        {
                return FAILURE;
        }

        for(i=0;i<16 && k<256;i++,k++)
                {
                data[k]=outdata[i];
                }
  }
  return SUCCESS;
}
```

```
/*** Beginheader gf_mul2 */
// multiply a GF(256) element by {02}
aes_elem gf_mul2(aes_elem s);
/*** EndHeader */

aes_elem gf_mul2(aes_elem s)
{
		return (s << 1) ^ ff_tab[s >> 7];
}

/*** Beginheader gf_mul4 */
// multiply a GF(256) element by {04}
aes_elem gf_mul4(aes_elem s);
/*** EndHeader */

aes_elem gf_mul4(aes_elem s)
{
		return (s << 2) ^ ff_tab[s >> 6];
}

/*** Beginheader gf_mul8 */
// multiply a GF(256) element by {08}
aes_elem gf_mul8(aes_elem s);
/*** EndHeader */

aes_elem gf_mul8(aes_elem s)
{
		return (s << 3) ^ ff_tab[s >> 5];
}

/*** Beginheader rot_column */
// rotate bytes within a column
void rot_column(aes_col in);
/*** EndHeader */

void rot_column(aes_col in)
{
	aes_elem t;
	t = in[0];
	in[0] = in[1];
	in[1] = in[2];
	in[2] = in[3];
	in[3] = t;
}
```

```
/*** Beginheader sub_column */
// forward byte substitution for a column
void sub_column(aes_col in);
/*** EndHeader */

void sub_column(aes_col in)
{
        in[0] = s_box[in[0]];
        in[1] = s_box[in[1]];
        in[2] = s_box[in[2]];
        in[3] = s_box[in[3]];
}

/*** Beginheader inv_sub_column */
// inverse byte substitution for a column
void inv_sub_column(aes_col in);
/*** EndHeader */

void inv_sub_column(aes_col in)
{
        in[0] = is_box[in[0]];
        in[1] = is_box[in[1]];
        in[2] = is_box[in[2]];
        in[3] = is_box[in[3]];
}

/*** Beginheader sub_bytes */
// forward byte substitution transformation
void sub_bytes(aes_state s);
/*** EndHeader */

void sub_bytes(aes_state s)
{
        int row;
        for(row = 0; row < n_row; ++row)
        sub_column(s[row]);
}

/*** Beginheader inv_sub_bytes */
// inverse byte substitution transformation
void inv_sub_bytes(aes_state s);
/*** EndHeader */

void inv_sub_bytes(aes_state s)
{
        int row;
        for(row = 0; row < n_row; ++row)
        inv_sub_column(s[row]);
}
```

```
/*** Beginheader shift_rows */
// forward shift row transformation
void shift_rows(aes_state s);
/*** EndHeader */

void shift_rows(aes_state s)
{
        int row;
  aes_elem t[n_row];
        for(row = 1; row < n_row; ++row)
        {
    //  aes_elem t[n_row];
                t[0] = s[(row + 0) % n_col][row];
                t[1] = s[(row + 1) % n_col][row];
                t[2] = s[(row + 2) % n_col][row];
                t[3] = s[(row + 3) % n_col][row];
                s[0][row] = t[0];
                s[1][row] = t[1];
                s[2][row] = t[2];
                s[3][row] = t[3];
        }
}

/*** Beginheader inv_shift_rows */
// inverse shift row transformation
void inv_shift_rows(aes_state s);
/*** EndHeader */

void inv_shift_rows(aes_state s)
{
        int row;
  aes_elem t[n_row];
        for(row = 1; row < n_row; ++row)
        {
        //aes_elem t[n_row];
                t[(row + 0) % n_col] = s[0][row];
                t[(row + 1) % n_col] = s[1][row];
                t[(row + 2) % n_col] = s[2][row];
                t[(row + 3) % n_col] = s[3][row];
                s[0][row] = t[0];
                s[1][row] = t[1];
                s[2][row] = t[2];
                s[3][row] = t[3];
        }
}
```

```
/*** Beginheader mix_columns */
// forward mix column transformation

void mix_columns(aes_state s);
/*** EndHeader */

void mix_columns(aes_state s)
{
        int col;
  aes_elem ad, bc, abcd;
        for(col = 0; col < n_col; ++col)
        {
        //aes_elem ad, bc, abcd;
                ad = s[col][0] ^ s[col][3];
                bc = s[col][1] ^ s[col][2];
                abcd = ad ^ bc;
                s[col][0] ^= abcd ^ gf_mul2(s[col][0] ^ s[col][1]);
                s[col][1] ^= abcd ^ gf_mul2(bc);
                s[col][2] ^= abcd ^ gf_mul2(s[col][2] ^ s[col][3]);
                s[col][3] ^= abcd ^ gf_mul2(ad);
        }
}

/*** Beginheader inv_mix_columns */
// inverse mix column transformation
void inv_mix_columns(aes_state s);
/*** EndHeader */

void inv_mix_columns(aes_state s)
{
        int col;
  aes_elem ad, bc, p, q;
        for(col = 0; col < n_col; ++col)
        {
        //aes_elem ad, bc, p, q;
                ad = s[col][0] ^ s[col][3];
                bc = s[col][1] ^ s[col][2];
                q = ad ^ bc;
                q ^= gf_mul8(q);
                p = q ^ gf_mul4(s[col][0] ^ s[col][2]);
                q = q ^ gf_mul4(s[col][1] ^ s[col][3]);
                s[col][0] ^= p ^ gf_mul2(s[col][0] ^ s[col][1]);
                s[col][1] ^= q ^ gf_mul2(bc);
                s[col][2] ^= p ^ gf_mul2(s[col][2] ^ s[col][3]);
                s[col][3] ^= q ^ gf_mul2(ad);
        }
}
```

```
/*** Beginheader add_round_key */
// add key transformation (same in both directions)
void add_round_key(aes_state s, aes_state k);
/*** EndHeader */

void add_round_key(aes_state s, aes_state k)
{
        int col;
        int row;
        for(col = 0; col < n_col; ++col)
                for(row = 0; row < n_row; ++row)
                        s[col][row] ^= k[col][row];
}

/*** Beginheader state_in */
// trabsfer the input to the state array
void state_in(aes_state s, void* in);
/*** EndHeader */

void state_in(aes_state s, void* in)
{
        int col;
  int row;
  int i;
        for(col = 0, i = 0; col < n_col; ++col)
                for(row = 0; row < n_row; ++row)
                        s[col][row] = ((unsigned char *) in)[i++];
}

/*** Beginheader state_out */
// trabsfer the state array to the output
void state_out(aes_state s, void* out);
/*** EndHeader */

void state_out(aes_state s, void* out)
{
        int col;
  int row;
  int i;
  for(col = 0, i = 0; col < n_col; ++col)
                for(row = 0; row < n_row; ++row)
                        ((unsigned char *) out)[i++] = s[col][row];
}
```

```
/*** Beginheader encrypt */
// encrypt a single block of 16 bytes
int encrypt(void *pt, void* ct);
/*** EndHeader */

// encrypt a single block of 16 bytes
int encrypt(void *pt, void* ct)
{
	int r;
	aes_state s;
	if(key_len)
	{
		state_in(s, pt);
		add_round_key(s, key_sch[0]);
		for(r = 1; r < key_len + 6; ++r)
		{
			sub_bytes(s);
			shift_rows(s);
			mix_columns(s);
			add_round_key(s, key_sch[r]);
		}
		sub_bytes(s);
		shift_rows(s);
		add_round_key(s, key_sch[key_len + 6]);
		state_out(s, ct);
		return SUCCESS;
	}
	else
		return FAILURE;
}

/*** Beginheader decrypt */
// decrypt a single block of 16 bytes
int decrypt(void* ct, void* pt);
/*** EndHeader */

int decrypt(void* ct, void* pt)
{
	aes_state s;
  int r;
	if(key_len)
	{
		state_in(s, ct);
		add_round_key(s, key_sch[key_len + 6]);
		for(r = key_len + 5; r > 0; --r)
		{
			inv_shift_rows(s);
			inv_sub_bytes(s);
			add_round_key(s, key_sch[r]);
			inv_mix_columns(s);
		}
```

```
                inv_shift_rows(s);
                inv_sub_bytes(s);
                add_round_key(s, key_sch[0]);
                state_out(s, pt);
                return SUCCESS;
        }
        else
                return FAILURE;
}

/*** BeginHeader */
#endif
/*** EndHeader */
```

9 Quellenverzeichnis

Betreuung

1. Berlin Heart AG: **www.berlinheart.de**
2. TeCNeT GMBH: **www.tecnet.de**

TCP/IP-Protokolle

3. TCP/IP-Grundlagen Protokolle und Routing, Heise, Gerhard Lienemann
4. Computernetzwerke, Andrew S. Tanenbaum, Person Studium
5. Virtuelle Private Netzwerke, VDE, Gerhard Lienemann
6. Messen, Steuern und Regeln per Internet, Franzis, Klaus-Dieter Walter
7. TCP/IP LEAN, Web Server for Telematik-Moduls, CMP Books, Jeremy Bentham
8. UNIX Netwrok Programming, Networking API, W. Richard Stevens

Telemedizin

9. Telemedizinführer Deutschland, Ausgabe 2004, Minerva
10. Medizinische Elektronik, Josef Eichmeier, Springerverlag
11. Telemedizin -Grundlagen -Perspektiven -Systeme -Anwendungen Proceedings Heinz Handels, Shaker Verlag
12. Rechtliche Rahmenbedingungen der Telemedizin, C.H. Beck Verlag, Angelika E., Hermeler
13. Studie der Firma Roland Berger & Partner GmbH
14. Forum der Medizin-Dokumentation und Medizin-Informatik Heft 1 / März 2004 Telematik im Gesundheitswesen, Bei Dr.rer.nat.habil Günter Steyer
15. Health Academy 01/2003, Telemedizin und Ökonomie
16. Health Academy 01/2002, Tele Home Care
17. Rechtfragen der Telemedizin, Springerverlag, Herausgeber: Ch. Dierks. , H. Feussner., A. Wienke,
18. Tagungsbänder 5,7 und 8 zur Fortbildung und Arbeitstagung, TELEMED 00,02 und 03
19. Chancen, Risiken und Gefahren der Telemedizin, TELEMED 98, Kongressdokumentation
20. **http://telemed-atlas.de/**
21. **http://www.telemedsys.de/de/index.shtml**
22. **http://www.m-ww.de/enzyklopaedie/diagnosen_therapien/telemedizin.html**
23. **http://www.ads.tuwien.ac.at/teaching/ws03/PSEinf/2003-12-02-b.pdf**
24. **http://www.imbi.uni-freiburg.de/medinf/it-eductra/p313ge/sld001.htm**
25. **http://www.informatikmed.de/telemedizin.htm**
26. **http://www.fzi.de/mit/fachgebiet.html**
27. **www.telemedizin.philips.de**

Kryptographie

28. The Design of Rijndael, AES-The Advanced Encryption Standard, Springer, Joan Daemen, Vincent Rijmen
29. Kryptographie in C und C++, Springer, Michael Welschenbach
30. Kryptographie, Grundlagen, Algorithmen, Protokolle, Spektrum akademischer Verlag Dietmar Wätjen
31. Internet-Sicherheit, Browser, Firewalls und Verschlüsselung, Hanser, Kai Fuhrberg, Dirk Häger, Stefan Wolf
32. Kryptografie und Public-Key-Infrastrukturen im Internet, Dpunkt.Verlag Klaus Schmeh

Hardware & Software

33. Softwareentwicklung in C and C++, Springer, Klaus Schmaranz
34. C The complete Reference, McGraw-Hill, Herbert Schildt
35. Programming Telematik-Moduls in C and C++, O'REILY, Michael Bar
36. Software Engeineering, Pearson Studium, Ian Sommerville
37. Embedded Ethernet and Internet Complete, Lakeview Research, Jan Axelson
38. Das Embedded PC Handbuch, Franzis, Hans-Joachim Blank
39. Designing Embedded Hardware, O'REILLY, John Catsoulis
40. Coputerschnittstellen, Hanser, Preuß/Musa
41. Halbleiter-Schaltungstechnik, Springer, U.Tietze, CH.Schenk

Herzunterstützungssysteme

42. Handbuch der Kardiotechnik, Gerhard Lauterbach, Gustav Fischer, Springer
43. Diplomarbeit: Modellierung und Simulation des Systems »Herz -Kreislauf -Kunstherz« mit Matlab / Simulink, Dr. med. Peter Göttel
44. **http://www.herzersatz.de/**
45. **http://www.netdoktor.de/**

Hersteller

46. RCM3200: **www.rabbitsemiconductor.com**
47. Dynamic C: **www.zworld.com**
48. Comtech: **www.comtech.uk.com**
49. Texas Instruments: **www.ti.com**
50. Electronic-Assembly: **www.electronic-assembly.de**
51. Intersil: **www.intersil.com**
52. Microchip: **www.microchip.com**
53. PCAD: **www.pcad.com**
54. BORLAND: **http://www.borland.com/**
55. National Semiconductors: **http://www.national.com/**