Zwischenbericht
Projekt Elektronik

„Betty“

Gruppe 21 – WS 2001/2002

Betreuer: Jörg Folchert

26.November 2001

Projektteilnehmer:

Mohamed Akresh	TI	202078, (mohbelg@cs.tu-berlin.de)
Moulay Fatmi El Fadili	TI	170228, (myfatmi@hotmail.com)
Wolfram Fritzsch	TI	160463, (syncro@fritzsch.com)
René Gegusch	TI	181663, (ReneGegusch@t-online.de)
Christian Jung	ET	186917, (chrjung@gmx.de)
Jan Weil	ET	182969, (jan.weil@web.de)
Jens Zech	TI	182682, (jzech@gmx.de)

Inhalt

Abbildungen

1 Allgemeine Projektbeschreibung

Das Ziel dieses Projektes ist der erfolgreiche Entwurf und Aufbau einer komplexen, selbstzudefinierenden elektronischen Schaltung. Diese wird im Rahmen der Lehrveranstaltung Projekt Analog-/Digitalelektronik an der TU Berlin durchgeführt.

Dieser Zwischenbericht gibt einen Überblick über den derzeitigen Werdegang des Projekts. Er beinhaltet den Weg von gewählten Vorschlägen bis hin zur Definition des Projekts „*Betty*“.

Ziel ist die Entwicklung eines Mikrocontroller gesteuerten Fernbedienungselements. Hierfür wird eine kurze Einführung in die drei Teilaspekte angefügt, welche dem Leser einen groben Überblick über deren Funktionsweise verschafft.

Da es sich bei diesem Bericht nur um einen momentanen Zustand handelt, sind einzelne Teilbereiche nicht in umfassender Form erläutert bzw. spezifiziert.

In allen folgenden Kapiteln wird immer zwischen den drei Teilmodulen, Mikrocontroller, Display/Tastatur und Kommunikationsschnittstelle unterschieden.

1.1 Projektvorschläge

Während der Projektfindungsphase in den ersten beiden Projekttreffen, gab es folgende Vorschläge, welche genauer in den jeweiligen Protokollen (Protokoll I, Protokoll II) nachzulesen sind:

- *Funkuhr/Wetterstation*
    - bisherige Wetterstationen stellen immer wieder ein übergreifendes Projekt dar, wobei die Anforderungen bzw. die Aufgabenbereiche die gesamte Zielpalette des Projekts Elektronik beinhalten.
    - Ziel unseres Vorschlages sollte eine möglichst umfassende Datenerfassung mit der zusätzlichen Auswertung durch Software, hierfür wäre z.B. auch die Entwicklung mit einem RAM-Baustein erforderlich gewesen, um die Messwerte zu erfassen.
    - Ein Funkuhr zählt zum selben Vorschlag, da hier eine Teilaufgabe der Datenabgleich für eine interne Uhr gegeben wäre.
- *PC gesteuertes Netzteil*
    - ein PC-gesteuertes Netzteil kam als Diskussionsstoff von Jörg (Betreuer), da dieses in der Form nicht am Institut existiert und ein solches für etliche Aufgaben zur Entwicklung elektronischer Schaltungen sinnvoll wäre.
- *PWM-Steuerung (Pulsweitenmodulation)*
    - ebenfalls ein Vorschlag von Jörg, eine Steuerung für das am Institut existierende Kart.

  - Ziel sollte es sein, die momentane doch eher ruckhafte Stufenschaltung durch eine „softe“ Schaltung zu ersetzen

- *Stromsenke*
  - ist ein Vorschlag vom Institut, zum kontrolliertem Entladen von Akkus aller Art.
- *USB Funkübertragung*
  - eigentlich eine von uns als Voraussetzung zur Kommunikation mit einem PC gewählte Übertragungsmethode, da diese bisher nicht in Projekten verwandt wurde, und die Mehrheit sich dieser Schnittstelle annähern wollte (will).

1.2 Projektfindung

Durch die Aufzählung obiger Ideen, sind wir zu der Überzeugung gelangt, in unser zu entwickelndes System folgende Teilgebiete aufzunehmen:

- graphisches Display
- Mikrocontroller
- Kommunikation mit einem PC (seriell oder USB)
- Eingabe per Tastatur oder Touchscreen

Zur besseren Koordination haben wir uns dafür in Gruppen aufgeteilt, welche sich verstärkt um die Durchführung der einzelnen Teilgebiete kümmern.

1.3 Betty

Der Name „*Betty*“ ist ein einfach von uns gewählter Name für dieses Projekt. Es ist keine Abkürzung bzw. Synonym. Im folgendem wird das Projekt immer als „*Betty*“ bezeichnet.

Der Aufbau und die Funktionsweise von *„Betty“* soll dem einer Fernbedienung mit grafischem Display ähneln, wie z.B. die programmierbarer Videorecorder. Die Verwendung von *„Betty“* ist mit der Programmierung eines Videorecorders vergleichbar. Als Gegenstelle wird bei diesem Projekt ein PC mit Windows NT 4.0 dienen, welche am Institut zur Softwareentwicklung zur Verfügung stehen.

Für die Kommunikationsstrecke wollen wir ein Protokoll entwerfen, welches den Anforderungen an Modularität zur Softwaresteuerung gewährleistet.

Die Entwicklung und Implementierung einer API (Application Programmer Interface) wäre ein krönender Abschluss für unser Projekt, wird jedoch voraussichtlich aus Zeitgründen nicht erreichbar sein.

„Betty“ ist ein in drei Module unterteilter Aufbau, welcher ein graphisches Display inklusive Eingabesteuerung, einen Mikrocontroller und eine Kommunikationsschnittstelle beinhaltet.

1.3.1 Mikrocontroller

Der Mikrocontroller dient zur vereinfachten Kommunikation zwischen den einzelnen Komponenten. Insbesondere soll hierbei die Ansteuerung des

Displays vereinfacht werden. So sollen beispielsweise Buttons und Menustrukturen als Funktionen vorgegeben sein.

Weiterhin hat der Mikrocontroller den Vorteil der leichter änderbaren Programmabläufe, gegenüber fest verdrahteter Ansteuerung oder EPROMS.

Ein zentrales Programm im Mikrocontroller steuert den dynamischen Programmablauf.

Der Mirkocontroller stellt die zentrale Komponente des gesamten Projekts dar. Er ist mit allen Teilkomponenten direkt verbunden und stellt somit ein führendes Bindeglied dar.

1.3.2 Schnittstelle

Da *„Betty“* eine Art universeller Fernbedienung sein soll, brauchen wir Schnittstellen nach außen, um beispielsweise mit einem PC kommunizieren zu können. Um ‚Betty‘ tatsächlich universell einsetzen zu können, brauchen wir die Möglichkeit, die Funktionalität in einer Initialisierungsphase zu konfigurieren, und dies impliziert einen Kanal vom PC zu ‚Betty‘.

Unsere Wunschvorstellung bestand zunächst aus einem Funkmodul, das auf Seite des PCs über den USB-Port verbunden werden sollte, um Plug and Play-Funktionalität bieten zu können.

Im Rahmen unserer Vorbereitungen kristallisierte sich heraus, dass das USB-Modul zwar durchaus realisierbar ist, in seinem Umfang allerdings schon alleine ein eigenes Projekt rechtfertigen würde.

Deshalb haben wir uns auf folgenden Kompromiss verständigt: Die Verbindung auf der PC-Seite erfolgt über die serielle Schnittstelle (RS232). Um möglichst schnell mit der Verbindung arbeiten zu können, wird zunächst ein kleines Modul entwickelt, dass die direkte Verbindung des Mikrocontrollers mit dem PC ermöglicht. Zum Zeitpunkt der Formulierung dieses Protokolls haben wir bereits erfolgreich über die serielle Schnittstelle und ein Terminal-Programm mit dem Mikrocontroller kommuniziert und ein Platinenlayout ist in Arbeit. Die Kommunikation über dieses Modul emuliert das geplante Funkmodul, d. h. insbesondere, es wird lediglich als halbduplex-Verbindung verwendet, da wir mit dem geplanten Funkmodul in unserem Rahmen nur einen Kanal verwirklichen können.

Die USB-Verbindung halten wir uns als Option für die Zukunft offen.

1.3.3 Display/Eingabesteuerung

Bei den ersten Treffen unserer Projektgruppe wurden über verschieden Vorschläge nachgedacht, wobei als Gedanke eine Wetterstation ins Auge gefasst wurde. Da bei einer solchen Wetterstation mehrere Daten angezeigt werden müssen, wurde die Entscheidung zur Verwendung eines graphischen Displays gefasst. Das gesamte Projekt wurde später noch näher spezifiziert und auch der Gedanke an eine Wetterstation verworfen, aber es blieb bei der Entscheidung für ein Display, und zwar ein Grafikdisplay.

Weiterhin sollte eine Eingabe an dem "Terminal", sprich dem Display, möglich sein. Aus diesem Grund haben wir uns für ein Touchscreen bzw. eine Touchfolie entschieden. Da es sich aber als fast unmöglich erwies, ein

Touchscreen zu organisieren, wurde die Eingabe auf ein Eingabegerät wie Tastatur bzw. Schalter reduziert.

Zur schnellen Realisierung unseres Projekts, werden wir den Input über eine herkömmlichen Tastatur (max. 6 Tasten) bzw. eine Art Drehgeber durchführen, da wir nur einen begrenzten Bereich von Eingabemöglichkeiten zur Verfügung stellen wollen.

Im Bausatz nach einem Projekt in der c’t haben wir die Möglichkeit eines Drehgebers gefunden. Diesen werden wir zur Eingabesteuerung verwenden.[Heise], [Projekt]

2 Erläuterung der Module

2.1 Projekt Betty im Ganzen

Folgende Abbildung gibt einen Überblick zu Betty. Dieses ist ein grober Entwurf des Projekts und gibt nicht die endgültige Zielsetzung wieder. Hier sind alle Komponenten, welche verwendet werden sollen, aufgeführt.

Die aufgeführten „additional sensors“ sollen die Erweiterungsfähigkeit des gesamten Projekts zeigen, dieses ist jedoch nicht in der Zielsetzung beinhaltet. Sollten wir zum Ende des Projekts noch Zeit haben, ist evtl. noch der Einbau von kleinern Datenerfassungen geplant. [Betty]

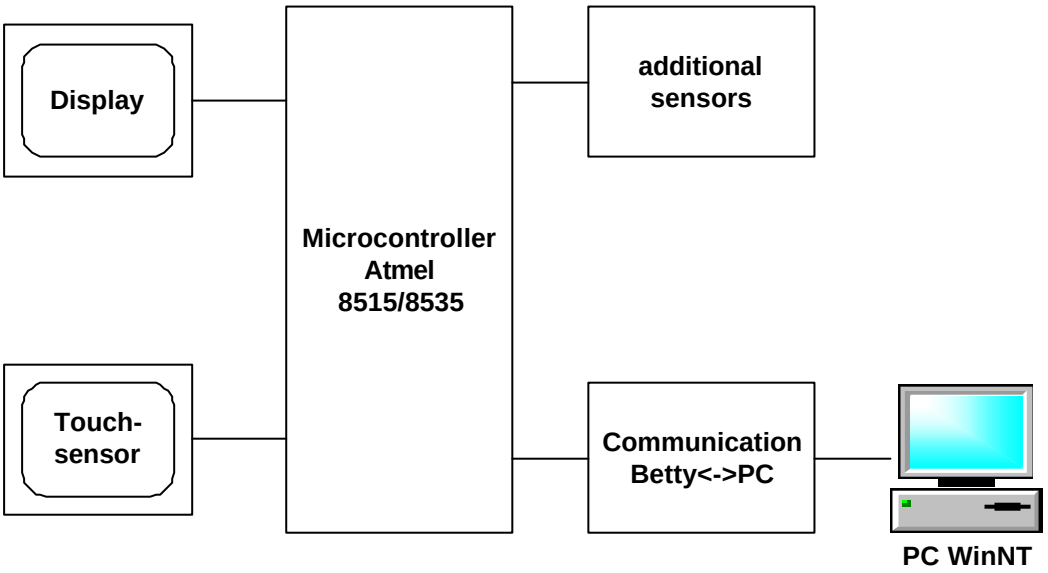

Abbildung 1 – Projektskizze

Um eine effiziente Projektplanung und –durchführung zu gewährleisten, wurde das Projekt in drei Teilbereiche aufgegliedert:

- Mikrocontroller (René und Jens)
- Kommunikationsschnittstelle (El Fadih, Mohamed, Jan)
- Display/Eingabe (Christian, Wolfram)

Um alle Aufgaben und Zielsetzungen zu erreichen, treffen wir uns neben dem allgemeinen Projekttermin zusätzlich einmal pro Woche, um Entwicklungsstände zu erläutern bzw. Probleme abzugleichen.

Die Webseite ist momentan http://user.cs.tu-berlin.de/~jenszech. Diese wird in den nächsten Wochen auch über die allgemeinen Projektseiten des Instituts abrufbar sein.

2.1.1 Mikrocontroller Atmel 8515

Der Atmel 8515 ist ein frei programmierbarer Mikrocontroller mit vier 8-Bit Ports, welche als Ein- oder Ausgang benutzt werden können.

Der Controller wird in Assembler programmiert und stellt dafür 8kB Programmspeicher zur Verfügung. Unter Assembler können die High/Low Pegel der Ports direkt abgefragt/gesetzt werden. Auch der Zugriff auf andere Komponenten, wie z.B. Timer, Watchdog usw., können direkt in Assembler realisiert werden.

Beachtet werden muß dabei, wie die Ports möglicherweise bereits für andere Funktionen mitbelegt sind, so ist beispielsweise der Port D mit der RS232 Schnittstelle direkt verbunden. Von den 8 Bit des Ports dienen 2 Bit für die Datenübertragung der UART/RS232 Schnittstelle.

Als Versorgungsspannung benötigt der 8515 eine Gleichspannung von 5 Volt, wobei wir zur Vereinfachung einen Spannungswandler auf dem Board verwenden der eine Eingangsspannung von 7 bis ca. 12 Volt erlaubt.

2.1.2 Kommunikationsschnittstelle

Die Kommunikation des Atmel 8515 mit einem PC über die serielle Schnittstelle erweist sich als vergleichsweise einfach.

Es muss lediglich eine Pegelanpassung vorgenommen werden, die durch Standardbausteine, wie z. B. von Maxim realisiert werden kann. Wir entscheiden uns für den Baustein MAX232ACPE, der nur noch mit wenigen Kondensatoren beschaltet werden muss. Das Schaltbild und das Platinenlayout finden sich im Kapitel zum Entwurf von *„Betty“*.

Auf Seite des Mikrocontrollers kommuniziert der UART über zwei Leitungen (TxD und RxD) des Ports D. Nachdem der UART über das zuständige Register konfiguriert ist, verläuft die Kommunikation quasi automatisch.

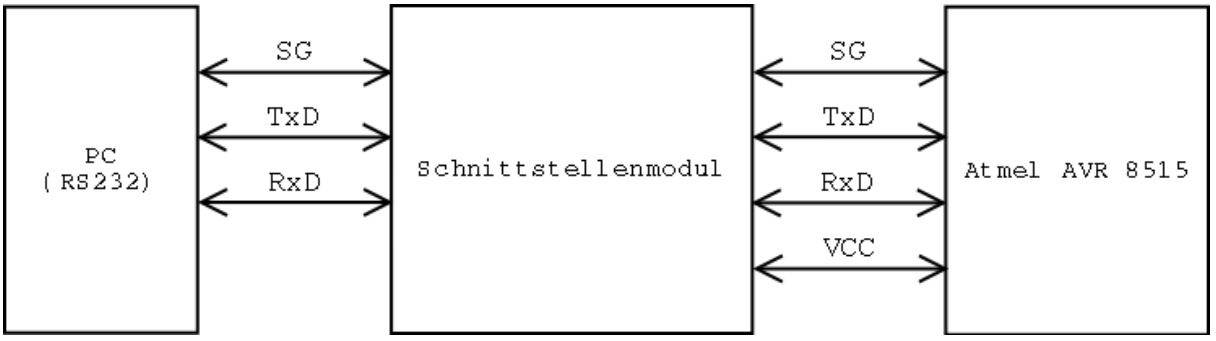

Abbildung 2 – Schnittstelle PC <=> Mikrocontroller

2.1.2.1 Funk-Schnittstelle

Da wir weder illegal funken, noch unser Projekt erst offiziell anmelden wollen, nutzen wir für die Funk-Schnittstelle die sogenannten ISM-Frequenzen. ISM ist eine Abkürzung für „Industrial Scientific and Medical“ und bezeichnet eine Gruppe von Frequenzbereichen, die von Hochfrequenzgeräten und Hochfrequenzanlagen genutzt werden dürfen. Generell ist eine Anwendung von

ISM-Frequenzen möglich, wenn es nicht so sehr auf eine störungsfreie Funkverbindung ankommt, bzw. Störungen nur vorübergehend auftreten und hingenommen werden können und nur kurze Entfernungen zu überbrücken sind. Entscheidend für uns war wie gesagt, daß wir unser Gerät nicht erst anmelden müssen. Um Störungen weitestgehend vermeiden zu können, ist ein angemessenes Kommunikationsprotokoll vorgesehen (s. u.). [RegTP]

Es gibt auf dem Markt mittlerweile eine Menge vorgefertigter HF-Sende- und Empfangsmodule, mit denen, da sie aufeinander abgestimmt sind, eine Funkstrecke relativ leicht realisiert werden kann. Die meisten von ihnen senden im 433 MHz-Band.

Viele dieser Module arbeiten intern mit SAW-Filtern, so dass sie sehr stabil funktionieren ohne nachjustiert werden zu müssen. SAW-Filter (SAW steht für Surface Accoustic Wave) nutzen gut zu integrierende Oberflächenstrukturen, indem elektrische Signale in akustische umgeformt werden. Der Vorteil der Stablilität wird in der Regel durch deutliche Leistungsverluste erkauft.

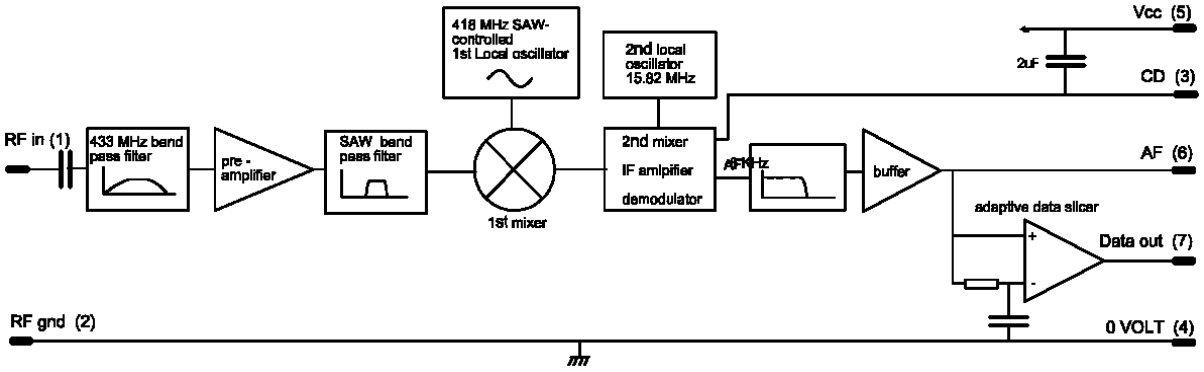

Abbildung 3 – Funk-Modul I

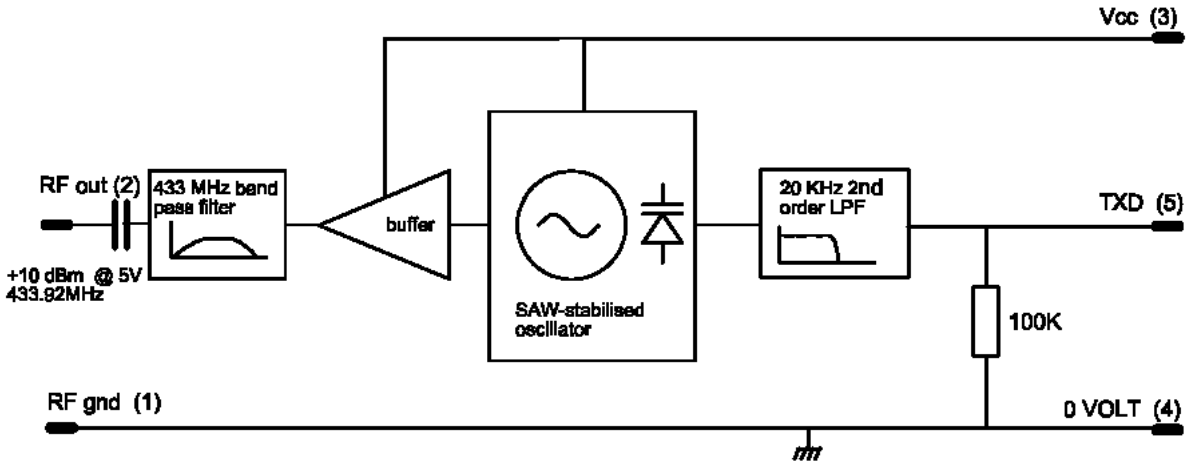

Abbildung 4 – Funk Modul IIr

2.1.2.2 USB-Schnittstelle (optional)

Universal Serial Bus, kurz USB, ist eine I/O intelligente Schnittstellentechnologie, die für ein reibungsloses Zusammenspiel von Endgeräten sorgt. Sie erlaubt den Anschluss von bis zu 127 Peripheriegeräten wie Tastatur, Maus, Drucker, Monitor, Lautsprecher, Scanner oder TK-Endgeräten, wie beispielsweise einem Modem. [USB]

Das Ziel ist, für alle zukünftigen Ein- und Ausgabegeräte am PC sowie für alle Anwendungen, eine gemeinsame Schnittstelle zu entwickeln. Der Anwender soll mit einfachen Mitteln alle Komponenten miteinander koppeln können.

Bei USB ist es immer noch möglich, die Geräte direkt an den PC anzuschließen. Jedoch hat der Anwender im weiteren die Möglichkeit, sogenannte „Hubs“ (eine Art Verteiler) mit dem PC zu verbinden, welche ihm erlauben, mehrere USB-Geräte an eine PC USB Buchse anzuhängen. Der USB bietet aber noch weitere Vorteile. So entfällt beispielsweise die lästige Aufgabe, jede neu an den PC angeschlossene Einheit extra zu konfigurieren.

Um langsame Geräte wie Tastatur und Maus als auch schnelle Geräte wie Modems oder Videokameras über ein und denselben Bus zu führen, wurde die Übertragung über den USB in Kanäle unterteilt.

Es gibt einen Low-Speed-Kanal mit bis 1,5 MBit/s(Maus, Tastatur) und einen Medium-Speed-Kanal mit 12 MBit/s(ISDN, Audio) die über dieselbe Schnittstelle geführt werden. Ein High-Speed-Kanal mit 500 MBit/s(Video, Speichermedien) ist auch vorgesehen. Unabhängig welcher Geschwindigkeitskategorie ein Gerät angehört, wird immer der gleiche vierpolige Stecker verwendet.

Bus Topologie

Der USB hat eine über vier Stufen kaskadierte Sterntopologie. Über einen Hub (Verteiler) ist die Peripherie sternförmig angeschlossen. Jede Verbindung Host-Hub, Hub-Hub, Hub-Node stellt eine Punkt-zu-Punkt-Verbindung dar.

Es gibt nach Vereinbarung nur einen Host im System, der den USB-Host-Controller aufnimmt.

USB Architektur

Ähnlich zu Konzepten wie z.B. dem OSI-Referenzmodell findet auch bei USB die Kommunikation auf mehreren Ebenen statt. Die Abbildung zeigt einen Überblick über das Kommunikationsmodell von USB.

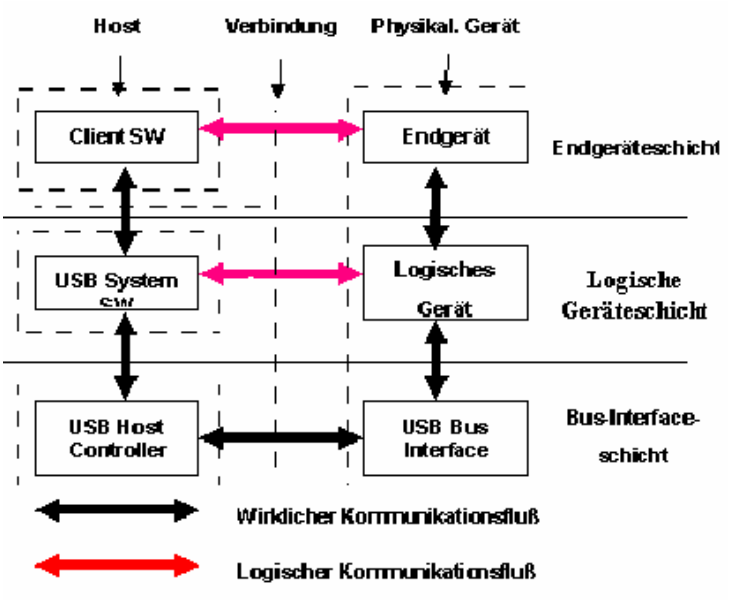

Abbildung 5 – USB Architektur

Aus der Abbildung wird ersichtlich, dass auf dem Host drei wichtige Komponenten zu beachten sind:

1. Client Software. Hierbei handelt es sich um Gerätetreiber, welche mit Funktionen auf USB Geräten End-zu-End Kommunikation betreiben. Beispiele dafür sind z. B. ein USB Maustreiber.
2. USB System Software. Diese Softwarekomponente ist unabhängig von speziellen USB-Gerätetypen und -eigenschaften, auf ihrer Ebene wird Kommunikation mit den logischen Geräten, also den Geräten als Ganzes, betrieben. Zu ihren Aufgaben gehört das Managen der Geräte, so z. B.:
    - das Erkennen neu angeschlossener Geräte an einem Bus,
    - Grundsetup eines Geräts (u.a. Zuweisen einer eindeutigen Nummer auf dem Bus),
    - Erkennen, dass Geräte vom Bus getrennt wurden
3. USB Host Controller. Der Host Controller ist einerseits eine Hardware-komponente, welche Pakete auf dem Bus übermittelt, andererseits gehört zu ihm auch eine Softwarekomponente (Host Controller Treiber), die seine Steuerung übernimmt und die Schnittstelle zur USB Systemsoftware herstellt. Eine Zusatzbemerkung: Ein Host kann durchaus mehrere Host Controller (auch verschiedenen Typs) besitzen, mit der Folge, dass die USB Systemsoftware mehrere Busse zu verwalten hat.

Ziel dieser Arbeit war es, für das Projekt Betty die Schnittstelle USB zu realisieren.

Es muss gesagt werden, dass das Thema USB ziemlich umfangreich ist, und wir denken, dass wir mit dem Zeitrahmen, den wir haben, nicht hinkommen. Ein paar Mitglieder haben vorgeschlagen, eine andere Schnittstelle (RS232) zu realisieren und USB als Option hinzunehmen.

2.1.3 Display/Eingabesteuerung

2.1.3.1 Funktionsweise von Displays

LCDs (Liquid Crystal Displays) bestehen aus einer dünnen Flüssigkeitsschicht zwischen zwei Glasplatten, die auf ihren Innenseiten durchsichtige leitende Beläge aus Zinnoxid haben. An diese Beläge wird die Spannung gelegt, die das benötigte elektrische Feld erzeugt, um die Kristalle der Flüssigkeit auszurichten. Ein Belag hat die Strukturen der anzuzeigenden Zeichen bzw. Pixel. Der Plattenabstand beträgt 5-10 µm. Um eine Trübung zu erzielen ist die Feldstärke von ca. 0,1 V/µm erfolderlich. Vergrößerte Feldstärke bedeutet intesivere Trübung (max. Trübung bei 3V/µm). Der übliche Leistungbedarf liegt bei ca. 0,1 µW/cm² Trübungsfläche. Die Trübung wird durch geeignete Beleuchtung sichtbar, besonders durch Verwendung eines Polarisationsfilters, der auf die Glasplatten aufgeklebt wird.

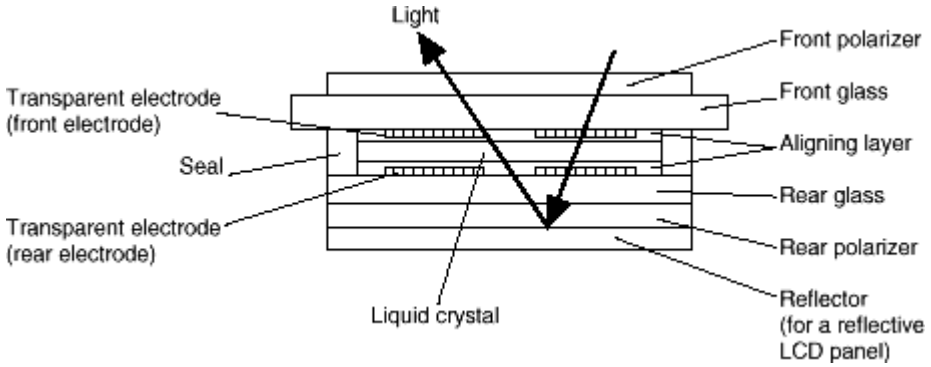

Abbildung 6 – Aufbau eine Displays

2.1.3.2 Eingabesteuerung

Aufgrund der Schwierigkeiten bei der Beschaffung eines Touchsensors als Eingabemodul, haben wir uns beim Treffen am 14.11.01 entschieden, die Steuerung des Hauptmoduls durch eine einfache Tastatur bzw. Joystick-Steuerung zu implementieren.

Der Grund hierfür ist die geringe Anzahl von Tastenfunktionen. Diese werden durch 3 verfügbare Ports am Atmel µController codiert. Hierfür verwenden wir die restlichen Datenleitungen des 2. Ports (PORT B), an dem vom Display bis jetzt nur die Steuersignale belegt sind (5 Datenleitungen). Mittels 3 Datenleitungen sind 7 Tastenfunktionen frei wählbar.

Nach einem nun vorliegendem konkreten Blockschaltbild für unsere Schaltung (siehe Kapitel 3) ist die Verwendung eines einzelnen Ports für die Eingabe sehr wahrscheinlich sinnvoll, da wir für den 4. Port des Mikrocontrollers bis jetzt noch keine andere Anwendung haben.

Kurz vor Fertigstellung dieses Zwischenberichts haben wir uns für die Ansteuerung durch einen Drehgeber entschieden. Aus diesem Grund können wir momentan leider nur auf den MP3-Player verweisen. Jedoch sollte die Schaltung und die Ansteuerung kein Problem darstellen, wenn man es mit dem Aufwand für die anderen Komponenten vergleicht. [Projekt]

3 Entwurf von Betty

3.1 Blockschaltbild

In folgendem Blockschaltbild ist der Aufbau des Moduls *„Betty“* näher erläutert, hierin sind die für die einzelnen Teilmodule erforderlichen Datenleitungen aufgelistet.

Nach Absprache ist die genaue Spezifikation der Anzahl von Zusatztasten mit fester Definition noch nicht festgelegt. Es wird jedoch darauf hinauslaufen, dass der Port C des Mikrocontrollers vollständig als Eingabeeinheit gesehen wird.

Die freien Port am Port B (Display) bzw. Port D (Serielle Schnittstelle) haben keine weitere Verwendung. Es ist angedacht, sofern noch die Zeit besteht, Status-LEDs oder zusätzliche Sensoren darüber zu steuern.

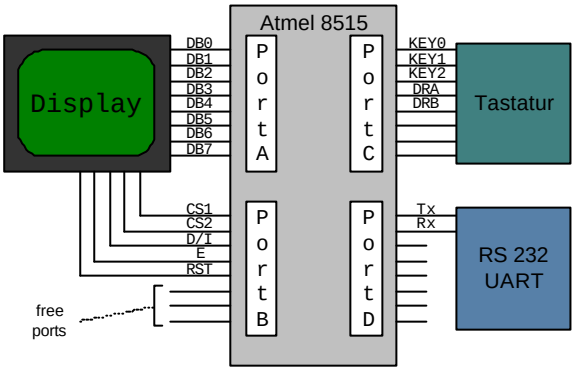

Abbildung 7 – Blockschaltbild mit Datenleitungen

Im den folgenden Kapiteln werden die drei Teilgebiete genauer erläutert.

3.2 Modulspezifischer Aufbau

3.2.1 Mikrocontroller

Gesucht wurde ein Controllerchip, der mehrere Ports als Ein- bzw. Ausgang zur Verfügung hat. Dieses Kriterium war insofern von Bedeutung, da wir relativ unabhängig von Display und/oder Sendeeinheit bleiben wollten.

Der Controller sollte möglichst eine schon vorhandene serielle oder besser USB Schnittstelle integriert haben. Dadurch wollten wir den Aufwand zur Übertragung verringern.

Die USB Schnittstelle kam für uns letztlich nicht in Frage, da schon alleine der Kostenfaktor für unser Projekt zu hoch erschien, immerhin hätten wir neben dem Controller auch entsprechendes Software Developer Kit benötigt.

Wir entschieden uns für einen schon beinahe *klassischen* Chip, dem Atmel 8515. Entwicklungsboard und Software sind uns zugänglich und die langjährige Existens dieses Chips vereinfachte unsere Wahl, da bereits Tutorials bzw. Beispiele ausreichend vorhanden sind und er auch an der TU schon mehrfach erfolgreich verwendet wurde.

3.2.1.1 Schaltung im Testmodus und erste Programmiertests

Der Atmel 8515 verfügt über insgesamt 4 Ports, welche als Ein- oder Ausgangports verwendet werden können. Die Programmierung erfolgt über ein selbstgebautes STK200 Board, angeschlossen über den Parallelport. [ATMEL]

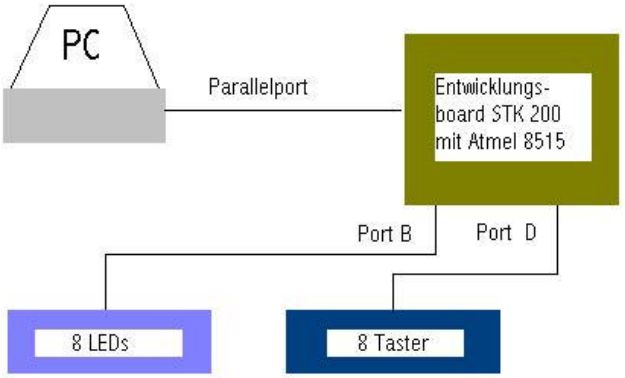

Abbildung 8 – erste Testschaltung

Unsere ersten Testschaltungen und Programmierungen beruhten auf der Ansteuerung von 8 LED´s an einem Port im Zusammenspiel mit 8 Tastern an einem anderen Port.

Unser Ziel sollte es sein die Assemblersprache der AVR Familie kennen zu lernen und essentiell wichtige Grundlagen für die Ansteuerung des Displays zu proben. Da wir noch keine Entscheidung hinsichtlich der Verwendung einer Touch Sensor Folie für den Prototyp treffen konnten (Problematik ist in erster Linie die Anschaffung und vor allem auch deren Kosten), bereiteten wir uns natürlich auch in erster Linie darauf vor, einfache Taster zu verwenden. Deren Abfrage stand im Vordergrund. Dabei versuchten wir verschieden Methoden der Abfrage (z.B. Schleifen und Interupts).

So programmierten wir unter anderem einen Zeitzähler, der uns de verstrichenen Sekunden von 0 bis 59 (also einer Minute) als Binärzahl auf den LED´s anzeigte. Zeitgesteuerte Abfragen werden wir später im Zusammenhang mit der Displaysteuerung benötigen.

Die Programmierübungen sind für uns auch zur späteren Einweisung der anderen Gruppenteilnehmer interessant. Aufgabe ist es später zur Programmierung des Displays grundlegende Programmiertechniken und Know-how weiterzugeben.

Über die inhaltliche Datenkommunikation zwischen den Komponenten haben wir bisher keine Entwürfe. Diese soll erst später folgen. Unser Vorhaben ist es in den ersten Schritten die allgemeine Ansteuerung zu erlernen und zu testen. Was und wie wir die Daten auf dem Display vom PC darstellen und welche Art von Daten auf den Datenleitungen gesendet werden steht für uns noch nicht zur Diskussion. Dies wird Aufgabe der nächsten Entwicklungsstufe sein.

Unser nächster Schritt war die Interaktion zwischen unserem Atmel Controller und dem PC. Später sollen schließlich Daten vom PC an den Controller übergeben werden, die dann verarbeitet werden und ggf. auf das Display

ausgegeben werden. Außerdem müssen Daten vom Controller an den PC weitergereicht werden.

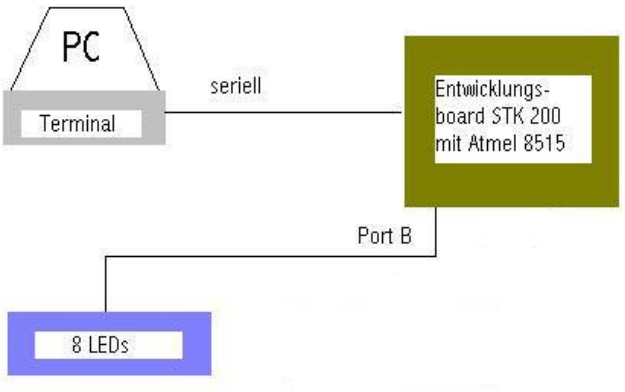

Abbildung 9 – Testschaltung zur seriellen Kommunikation mit dem PC

Hierzu verwendeten wir ein Terminalprogramm am PC und ließen darüber die empfangenden Daten anzeigen bzw. Daten an den Controller senden.

Unser Ziel war es ein Interupt - gesteuertes auslesen des UART Ports und dessen Benutzung kennen zu lernen. Mit Hilfe von unseren 8 LED´s am Port B überprüften wir empfangen Daten und führten auch gleichzeitig ein Ereignis aus (individuelles leuchten der LED´s).

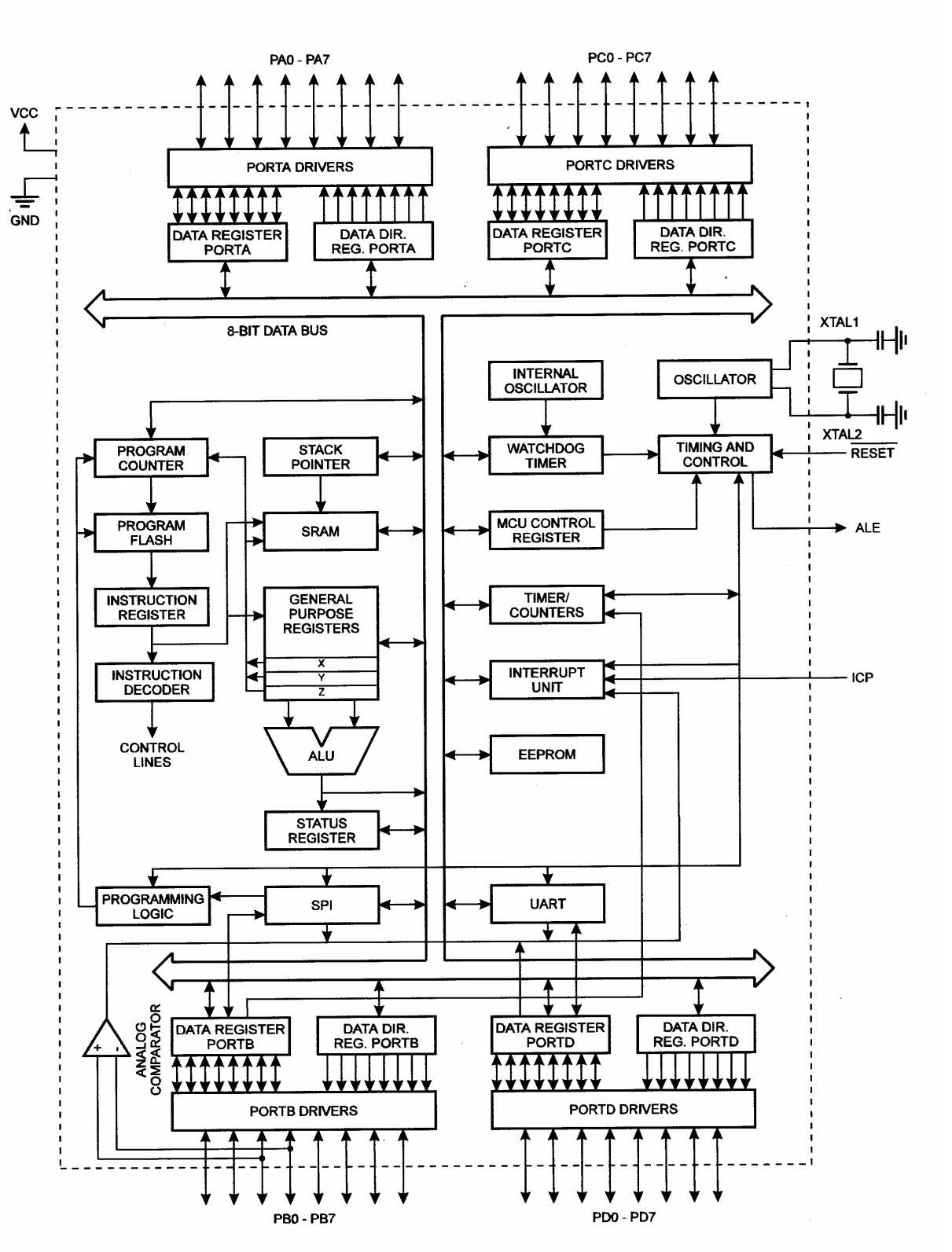

Abbildung 10 – Blockschaltbild des Atmel 8515

3.2.2 Display/Tastatur

Für die Darstellung von Datensätzen haben wir das graphische Display von SEIKO Instruments Inc. G1216B1N000. Dieses Display hat eine Auflösung von

128 x 64 Pixel, welches einem Graphikspeicher von 8192 bit entspricht. [SEIKO]

Über die Ansteuerung durch den Atmel sind wir uns noch nicht einige, da hierbei eine wichtige Entscheidungsschwelle der zur Verfügung stehende Speicher des Mikrocontrollers ist. Es muss also genau abgewogen werden, welche Verfahrensweise verwendet wird oder ob auch die Möglichkeit des Outsourcens durch eine Software-Applikation auf dem PC gegeben wird. Dieses wird in einem der nächsten Projekttermine geklärt.

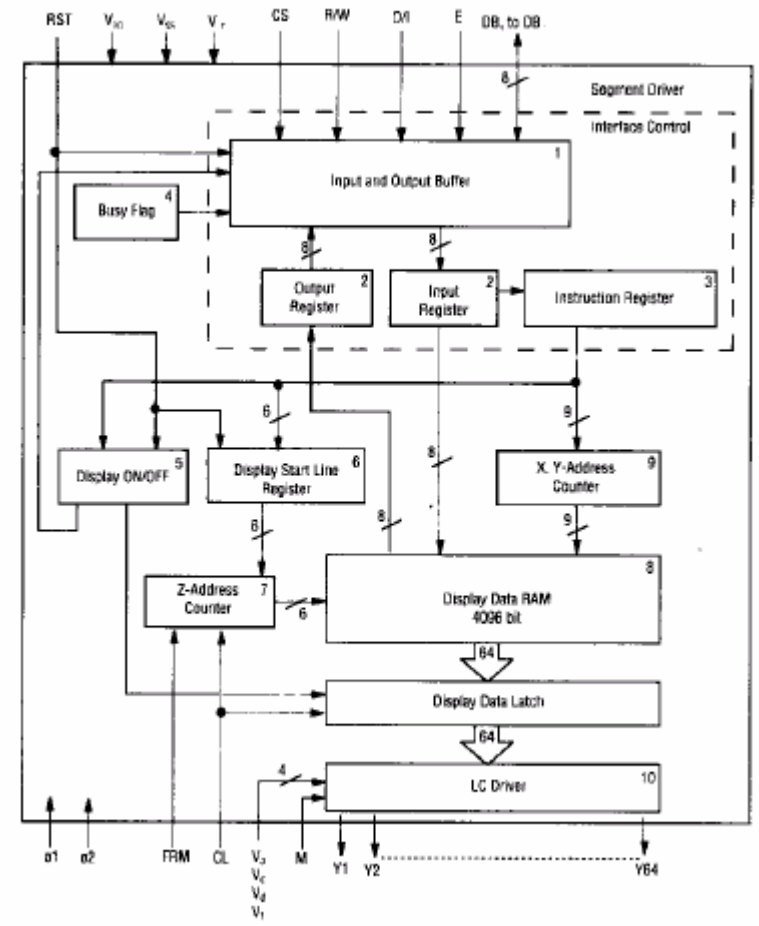

Abbildung 11 – Blockschaltbild Graphikmodul G1216B1N000

Der Vorteil dieses Modules ist die Integration von Steuereinheit und Data-RAM. Die Datenleitung DB0..DB7 sind für die Übertragung von Displaydaten, die restlichen (CS1, CS2, D/I, RST) sind zur Übertragung von Daten- bzw. Steuersequenzen.

Das Enable-Signal (E) ist zu Taktung des Adressbusses. Bei fallender Taktflanke liest der Controller die angelegten Daten ein. Die max. Frequenz beträgt 1MHzt, welche für die Nutzung im Zusammenhang mit dem 8515 ausreicht, da dieser max. mit 8MHz betrieben wird.

3.2.3 Kommunikation

Angeregt durch eine Artikelserie in der ‚Elektor' beschäftigten wir uns insbesondere mit den Modulen RX2 (Empfänger) bzw. TX2 (Sender) von Radiometrix und im Rahmen unserer Vorbereitungen wurde uns auch von mehreren Seiten bestätigt, das sich diese Module am besten für unser Projekt eignen.

Sie zeichnen sich aus durch außergewöhnlich hohe Unempfindlichkeit gegenüber Streustrahlung (wie sie z. B. von unserem Display erzeugt wird), sowie einer ungewöhnlich hohen Übertragungsrate. Typische Bitraten liegen bei anderen Modulen bei 2400 bit/s bis maximal 9600 bit/s. Das RX2 Modul von Radiometrix ist in der Lage bis zu 40000 bit/s zu empfangen. Nachteilig ist natürlich der erwartungsgemäß höhere Preis.

3.2.4 Kommunikationsprotokoll

Da wir auf der Mikrocontrollerseite die Applikation nicht in Abhängigkeit von der verwendetet Schnittstelle konfigurieren wollen, brauchen wir ein Protokoll, das dem schwächsten der zu verwendenden Module gerecht wird. Das ist in unserem Fall die halbduplex-Verbindung über den Funkkanal. Die Daten werden über die Funkstrecke vergleichbar mit der TxD-Datenleitung der RS232-Schnittstelle seriell übertragen. Es spricht also nichts dagegen, das RS232-Protokoll mit Start-, Stop- und Paritätsbit entsprechend zu erweitern. Da uns über die Funkverbindung nur ein Kanal gleichzeitig zur Verfügung steht, brauchen wir ein paketorientiertes Protokoll, das sicherstellt, dass gegebenenfalls auf beiden Seiten zwischen Sende- und Empfangsbetrieb umgeschaltet werden kann.

4 Zeitplan

30.10.01

- Planung der Verwendung eines Displays

06.11.01

- Entscheidung für das Display von SEIKO Instruments, Mikrocontroller Atmel 8515, Kommunikation RS232

13.11.01

- Vorplanung für Entwurf der Platine des Displays,
- erster Betrieb des Atmel mit Ansteuerung von LEDs

20.11.01

- Aufbau der Platine und erster Testversuch mit dem µController
- erste Kommunikation über RS232 mit Atmel (Terminalprogramm PC-seitig)

27.11.01

- Ansteuerung des Displays durch ATMEL-Controller (8515 oder 8535)
- MAX232-Platinen-Layout, Funkmodelstruktur

04.12.01

- Schriftzeichendarstellung auf dem Display, von Mikrocontroller zum Display
- Festlegung des Kommunikationsprotokolls

11.12.01

- Weitere Programmierung und Ansteuerung durch den Atmel µController, Display und serielle Schnittstelle

18.12.01

- Weihnachtsfeier,
- Timing der Steuerung von Display und Kommunikationsschnittstelle messen und verbessern

08.01.02

- Endentwurf und Layout

15.01.02

- Layout und Kontaktierung

22.01.02

- Layout und Kontaktierung

31.01.02

- Fertigstellung, weitere Programmierung

07.02.02

- Fertigstellung

15.02.02

- Abschlußberichterstellung

18.02.02

- ABGABE!

5 Referenzen

[ATMEL] http://www.atmel.com

[Betty] http://user.cs.tu-berlin.de/~jenszech

[Heise] http://www.heise.de

[Projekt] http://www.mp3pump.de/german/index.html

[RegTP] http://www.regtp.de

[SEIKO] http://www.seiko-instruments.com

[USB] http://www.usb.org