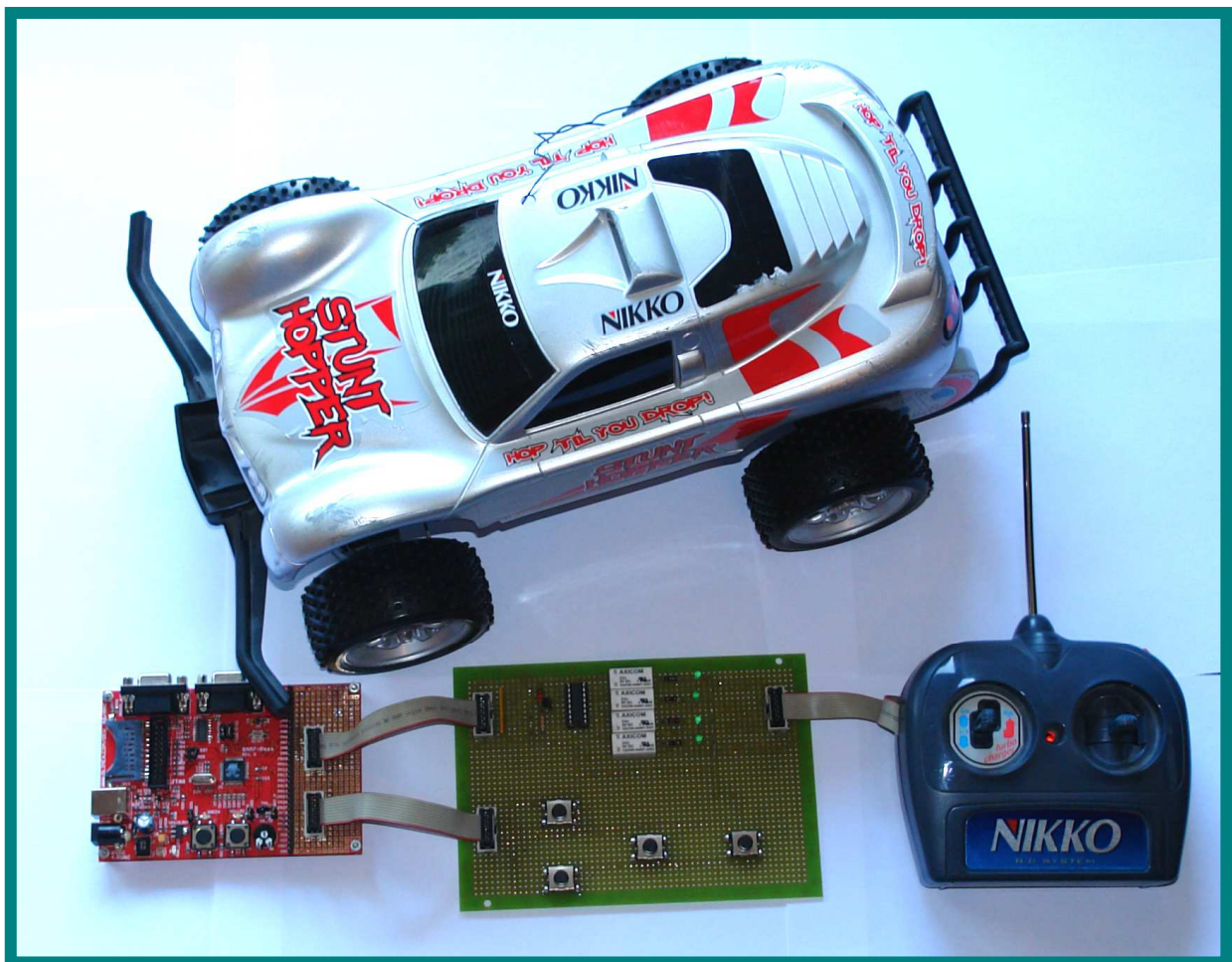


Projektarbeit 2008

Micro-Controller-Steuerung



Betreuer:
Herr Behrens

Datum: 13.05.08
Realschule am Lehmwohld

Inhaltsverzeichnis

Abbildungsverzeichnis.....	3
1. Wie es dazu kam:	4
2. Einleitung	5
2.1 Aufgabenstellung	5
2.2 Problemstellung.....	5
2.3 Ablauf.....	6
2.4 Bericht	6
2.5 Mittel	6
3. Systembeschreibung.....	7
3.1 Gesamtschaltung	7
3.1.1 Materialliste der Schaltung.....	11
3.2 Programmierung.....	12
3.3 Inbetriebnahme.....	12
3.3 Struktogramm.....	13
3.4 Quellcode	14
4. Die Bauteile.....	15
4.1 Der Micro-Controller SAM7S256 von Atmel:	15
4.1.1 Allgemeines über Micro-Controller	15
4.1.2 Aufbau eines Micro-Controllers.....	16
4.1.3 CPU - Central Processing Unit.....	17
4.1.4 MCU Funktionen.....	18
4.1.5 AT91SAM7S256.....	21
4.2 Der Compiler.....	23
4.3 Der Debug-Anschluss	24
4.4 Das Relais.....	24
Funktionsweise des Relais	24
4.5 Das IC.....	25
4.6 Der Widerstand	25
4.7 Der Transistor.....	25
4.8 Die Leuchtdiode	26
4.9 Der Taster	26
4.9.1 Das „Prellen“	26
5. Unsere Arbeitsweise.....	27
5.1 Danksagung	27
6. Fazit.....	28
7. Quellenverzeichnis	29
8. Bestätigung.....	30

Abbildungverzeichnis

Abb.1	Foto von der Schaltung vorne	7
Abb.2	Foto von der Schaltung hinten	7
Abb. 3	Die Schaltung	8
Abb. 4	Schaltplan	9
Abb. 5	Layout	10
Abb. 6	Struktogramm	13
Abb.7	Prinzipieller Aufbau eines Micro-Controllers	16
Abb.8	Schaltplan des Micro-Controllers	21
Abb.9	Der Micro-Controller SAM7S256	22
Abb.10	Code Optimierung des Compilers	23
Abb.11	Schaltung des Relais	24
Abb.12	Die Leitungen sind instabil.	25
Abb.13	Der PNP-Transistor	25
Abb.14	LED's	26

1. **Wie es dazu kam:**

Als wir zum ersten Mal gehört haben, dass wir dieses Jahr ein Projekt fertigstellen müssen, wussten wir, dass es irgendetwas Anspruchsvolles sein muss. Zuerst haben wir an einen motorisierten City-Roller gedacht. Aufgrund mangelnder Kenntnisse und der Tatsache, dass man Metall hätte schweißen müssen, verwarfen wir diese Idee. Durch meinen Vater, der in der Computerbranche tätig ist, kamen wir auf ein Projekt mit Micro-Controllern. Da wir aber nur wenige Kenntnisse in Elektronik hatten, hat uns mein Vater geholfen die richtigen Relais etc. auszusuchen. Auf der Basis des SAM7S256 Micro-Controllers der Firma Atmel wollten wir eine selbstlernende Steuerung bauen, eine Steuerung, die den Fahrweg aufzeichnet und danach von alleine wieder abfährt. Dazu gehörte auch das Programmieren in der Programmiersprache C, die wir mithilfe meines Vaters kennen lernen konnten.

2. Einleitung

2.1 Aufgabenstellung

Für die, im Rahmen der erwähnten Projektarbeiten entwickelte Steuerung soll ein Programm geschrieben werden, welches erlaubt, einen Fahrweg aufzuzeichnen und danach noch einmal exakt abzufahren.

Neben den eigentlichen Kontroll- und Steuerfunktionen ist ein spezielles Augenmerk auf die plausible und einfache Erklärung zu richten. Die selbstlernende Steuerung soll in der Lage sein, die anfallenden Daten ganzer Fahrwege zu speichern und bei Bedarf wieder abzufahren.

Die Projektarbeit umfasst unter anderem die folgenden Schwerpunkte:

- Erarbeiten des Gesamtkonzeptes
- Verfassen von Struktogrammen
- Festlegen des Funktionsumfanges
- Aufbau und Inbetriebnahme der eigentlichen Steuerung sowie der benötigten Hard- und Software

2.2 Problemstellung

Ursprünglich hatten wir geplant, eine Schaltung zu konstruieren, die einen Fahrweg speichert und wieder abfährt. Das erste Problem stellte sich, als wir dachten wir könnten den Speicher in das Auto hinein bauen.

Das hätte aber deutlich unseren Zeit- und Kenntnisrahmen gesprengt. Wir dachten uns, dass das Projekt einigermaßen leicht zu erarbeiten ist. Am Ende hatte uns die Zeit eingeholt und uns wieder auf den Boden der Tatsachen zurückgebracht, da dieses Projekt alles andere als einfach ist.

Das größte Problem und den damit verbundenen Zeitverlust stellte für uns die Programmierung da. Welche wir aber mit viel Zeit, Geduld, Ausdauer und hier und da mit einem Tipp gemeistert haben.

2.3 Ablauf

Zu Beginn der Arbeit ist ein Projektplan zu erarbeiten, um die Arbeit zeitlich zu strukturieren.

Die Arbeiten sollen innerhalb des Projektteams geeignet aufgeteilt werden; die Aufteilung ist im Bericht entsprechend festzuhalten.

Weitere Einzelheiten werden an den regelmäßigen Besprechungen festgelegt.

2.4 Bericht

Über die Projektarbeit ist ein Bericht zu verfassen, dessen Textteil (ohne Anhang) 2000 Wörter nicht übersteigen soll. Im Bericht sollen alle wesentlichen gemachten Überlegungen, Abklärungen, Berechnungen und Untersuchungen detailliert (in Text und Bild) dokumentiert werden. Der Bericht muss gut leserlich geschrieben und übersichtlich gegliedert sein. Er soll mindestens die folgenden Kapitel umfassen: Inhaltsverzeichnis, allgemeine (für Nichtfachleute) verständliche Einleitung, Aufgabenstellung (Original) sowie eine Zusammenfassung. Zum Bericht gehört auch eine Systembeschreibung.

2.5 Mittel

Zur Verfügung stehen folgende Mittel:

1. SAM7S256 Micro-Controller von Olimex
2. C-Compiler von IAR Systems
3. IAR embedded Workbench
5. Eagle.exe
6. Microsoft Office für die Dokumentation

3. Systembeschreibung

3.1 Gesamtschaltung

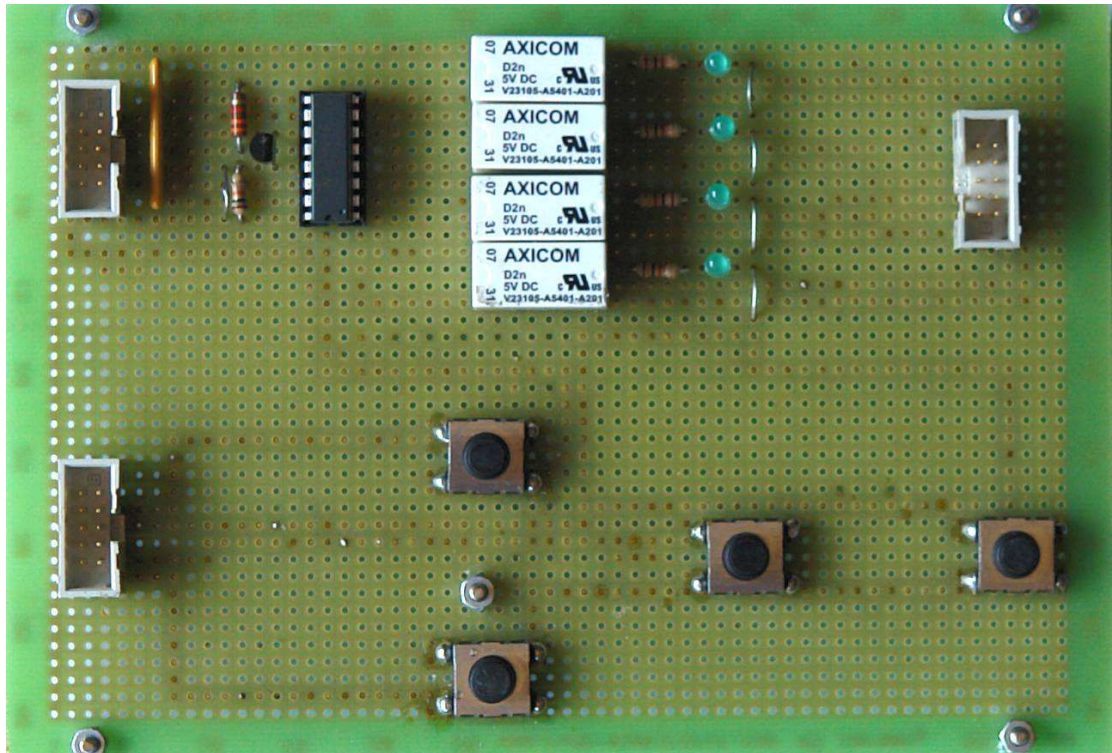


Abb.1 Foto von der Schaltung vorne

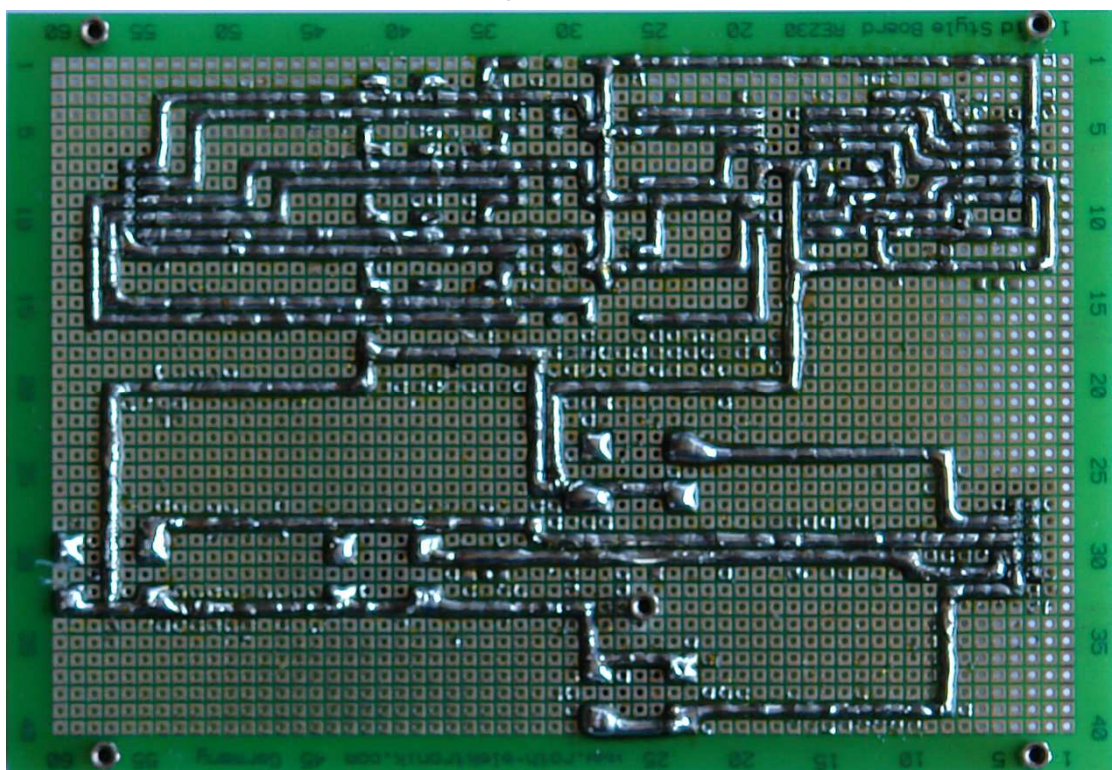


Abb.2 Foto von der Schaltung hinten

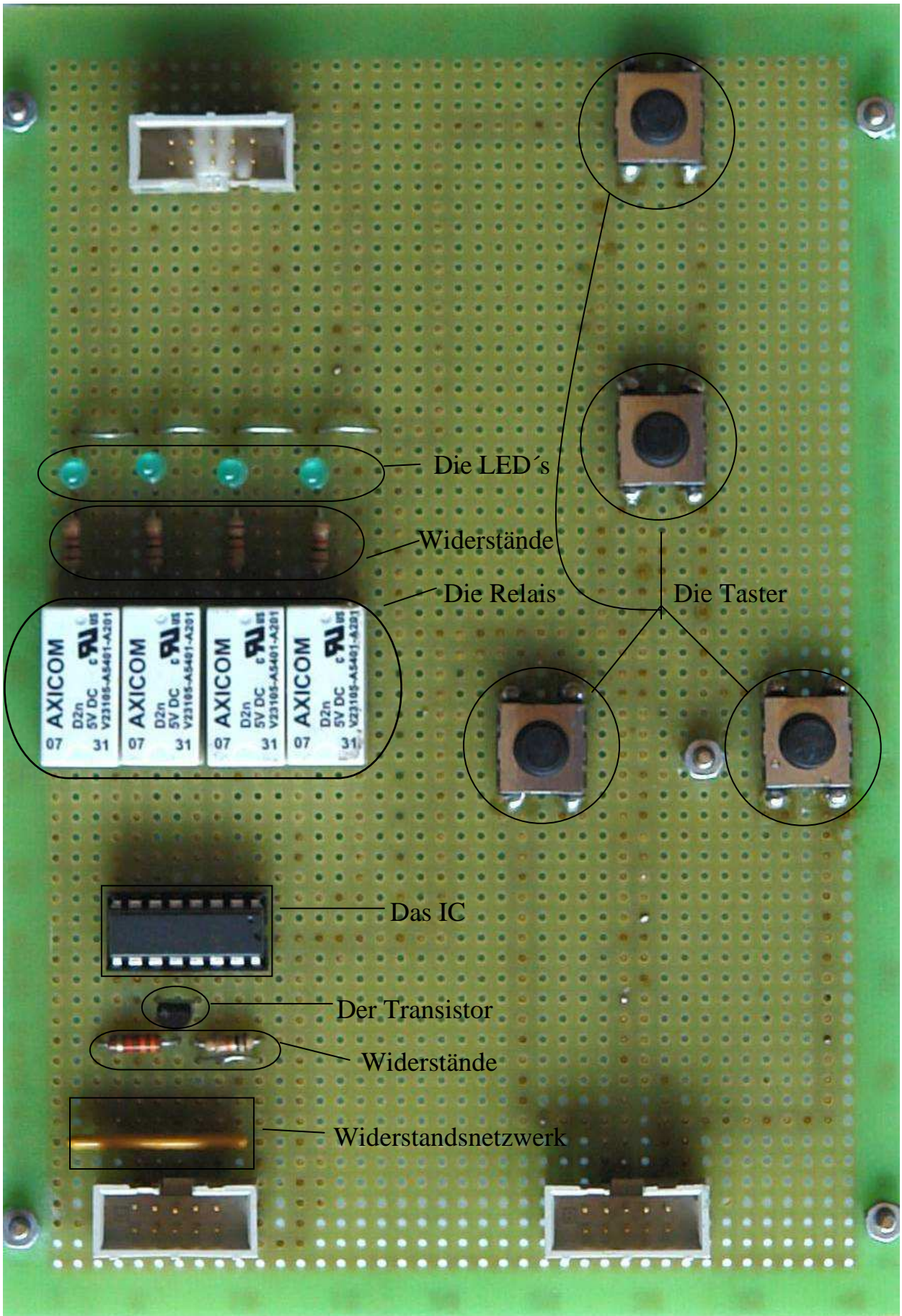


Abb. 3 Die Schaltung

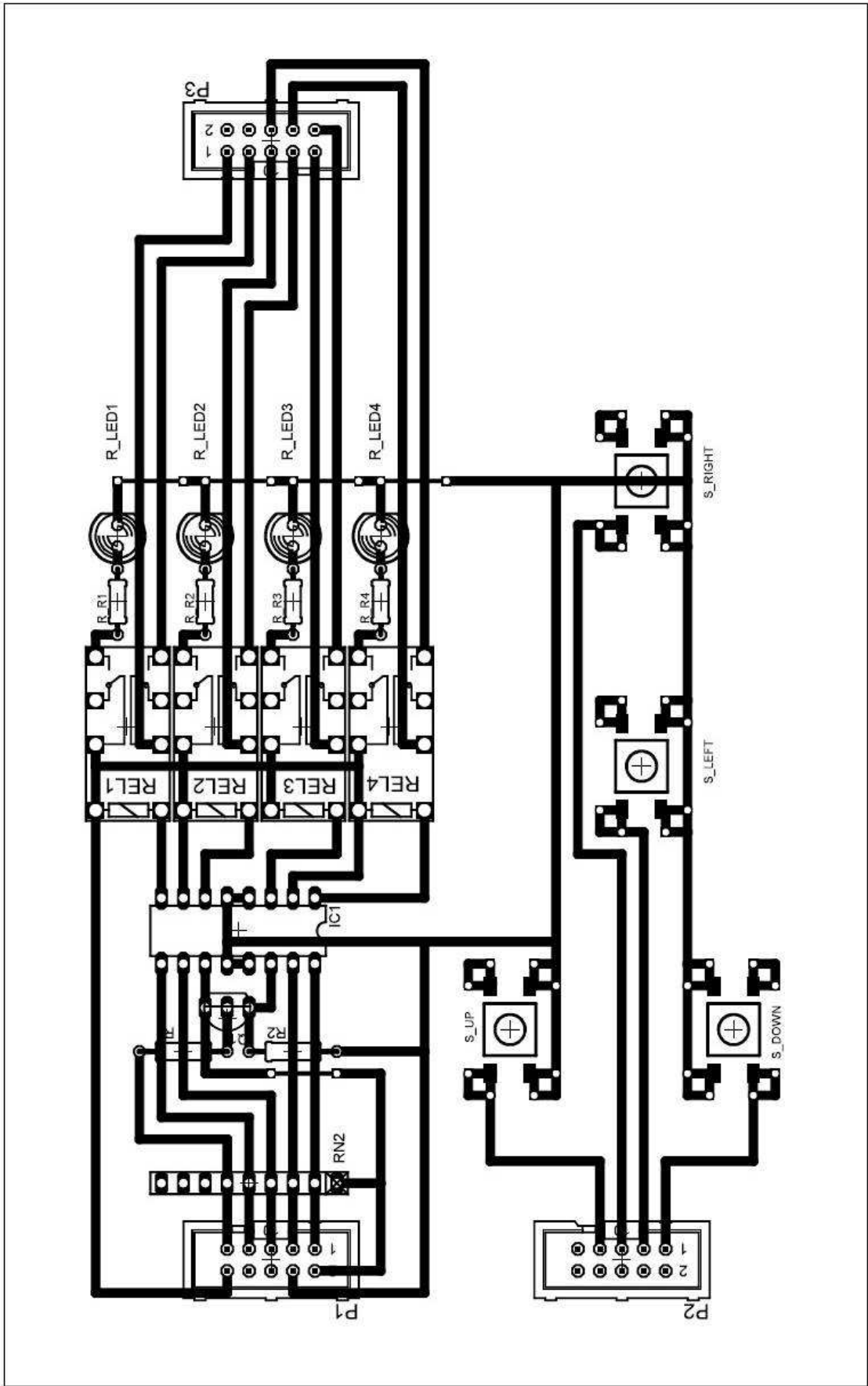


Abb. 5 Layout

3.1.1 Materialliste der Schaltung

Part	Value	Device	Package	Description
IC1	DS3668	DS3668	DIL16	DRIVER ARRAY
P1		ML10E	ML10	Connector
P2		ML10E	ML10	Connector
P3		ML10E	ML10	Connector
Q1	BC556	BC556	TO92-EBC	PNP Transistor
R1	3K3	R-EU_0207/10	0207/10	RESISTOR, European symbol
R2	10K	R-EU_0207/10	0207/10	RESISTOR, European symbol
REL1		REL-D2N	REL-D2N	RELAY
REL2		REL-D2N	REL-D2N	RELAY
REL3		REL-D2N	REL-D2N	RELAY
REL4		REL-D2N	REL-D2N	RELAY
RN2		RNX8	RN-9	RESISTOR NETWORK
R_LED1		LED5MM	LED5MM	LED
R_LED2		LED5MM	LED5MM	LED
R_LED3		LED5MM	LED5MM	LED
R_LED4		LED5MM	LED5MM	LED
R_R1	1K	R-EU_0204/7	0204/7	RESISTOR
R_R2	1K	R-EU_0204/7	0204/7	RESISTOR
R_R3	1K	R-EU_0204/7	0204/7	RESISTOR
R_R4	1K	R-EU_0204/7	0204/7	RESISTOR
S_DOWN	LSH	LSH	LSH	SMD Schurter Switch
S_LEFT	LSH	LSH	LSH	SMD Schurter Switch
S_RIGHT	LSH	LSH	LSH	SMD Schurter Switch
S_UP	LSH	LSH	LSH	SMD Schurter Switch

3.2 Programmierung

Zum Programmieren des Micro-Controller Boards verwendeten wir das Programm IAR embedded Workbench in der ARM Version, da diese mit unserem Prozessor kompatibel ist.

3.3 Inbetriebnahme

Als erstes verbindet man die Schaltung mit dem Micro-Controller Board und der Fernbedienung. Die Spannung von 5V schließt man an USB Port des Micro-Controllers an. Nachdem der Taster SW1(Aufzeichnen) betätigt wurde, kann der Prozessor mit der Aufzeichnung der Tastenabläufe an der Schaltung beginnen. Die Fernbedienung übermittelt dann die von uns gedrückten Tasten bzw. die daraus entstandenen Befehle an das RC Auto, welches dann vorwärts, rückwärts sowie nach links und rechts fahren kann. Nach einer vorher festgelegten Wegstrecke wird der Taster SW1 noch einmal gedrückt. Nun wird das Fahrzeug auf die Ausgangsposition gestellt und der Taster SW2(Abspielen) gedrückt.

Hinweis:

SW1 befindet sich unter der grünen LED auf dem Micro-Controller Board
SW2 befindet sich unter der orangen LED auf dem Micro-Controller Board

3.3

Struktogramm

Um die Programmierung einfacher zu gestalten, zeichnen Programmierer so genannte „Struktogramme“.

Hier ein Beispiel für „Vorwärtsfahrt“

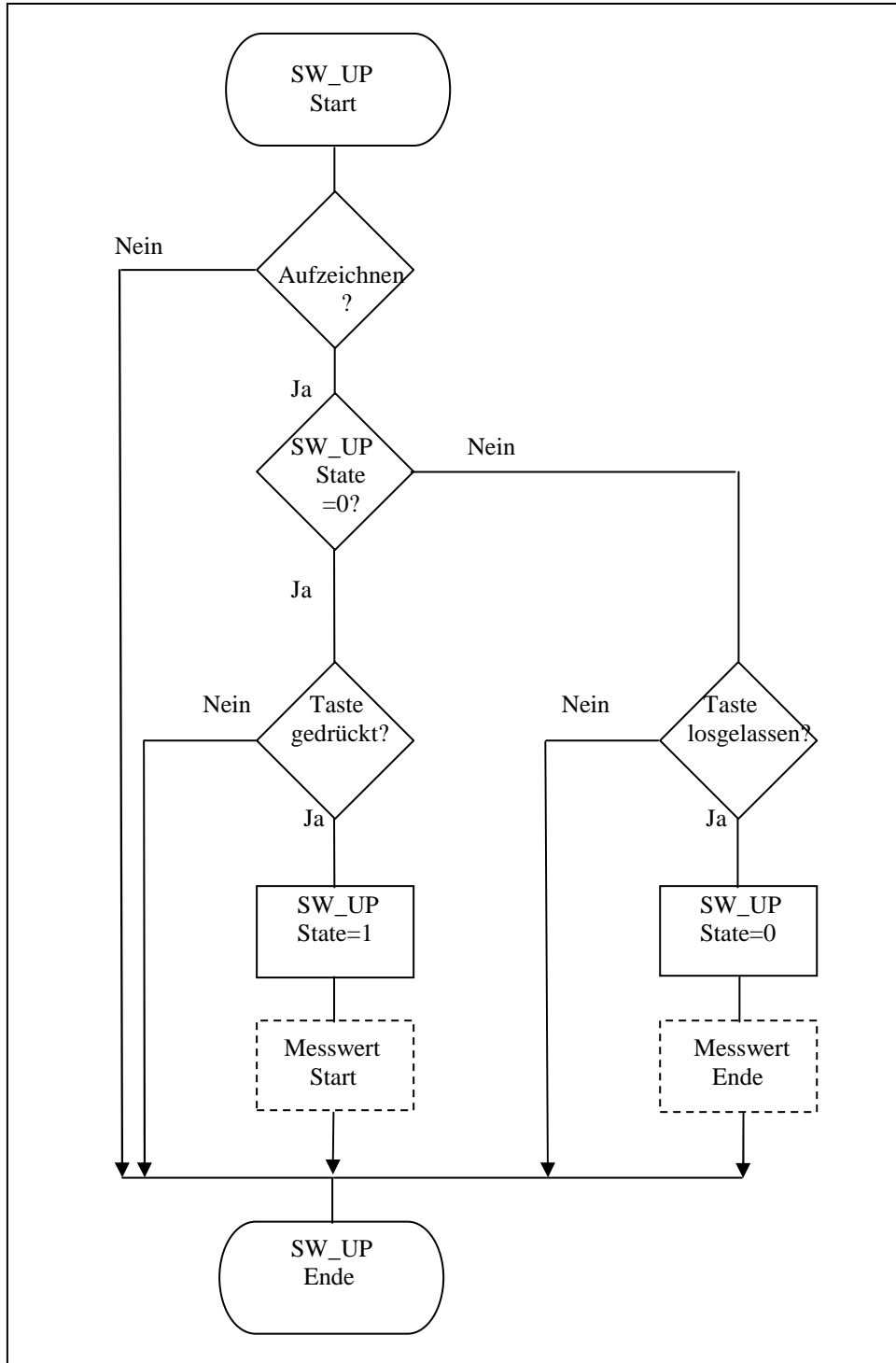


Abb. 6 Struktogramm

3.4

Quellcode

Hier der Quellcode für die oben, im Struktogramm, beschriebene Funktion:

```
-----  
/**  
 * Function Name      : CheckSW_UP  
 * Object            : Prüft den Zustand von SW_UP und signalisiert  
 *                   dessen Zustand. Dieser befindet sich auf der Schaltung.  
 * Input Parameters  : none  
 * Output Parameters : none  
 */  
-----  
void CheckSW_UP(void)  
{  
    if(StartAufzeichnen==1)  
    {  
        if(StateSW_UP==0)  
        {  
            //Taste gedrückt?  
            if(IsPinSet(SW_UP))  
            {  
                //Taste gedrückt!  
                StateSW_UP=1;  
            }  
        }  
        else  
        {  
            //Taste gedrückt?  
            if(IsPinSet(SW_UP))  
            {  
                //Taste gedrückt!  
                //nichts tun  
            }  
            else  
            {  
                //Taste  
                StateSW_UP=0;  
            }  
        }  
    }  
}
```

4. Die Bauteile

4.1 *Der Micro-Controller SAM7S256 von Atmel:*

4.1.1 Allgemeines über Micro-Controller

Micro-Controller (MCU) sind programmierbare Schaltungen (Prozessoren), welche für die unterschiedlichsten Anwendungen eingesetzt werden können. Ein Programm (Software) steuert den Funktionsablauf der Schaltung. Micro-Controller findet man in allen elektronisch kontrollierten Geräten vom ABS-System im Auto über das Videogerät, den Fotoapparat, die Harddisk-Einheit im PC, den Telefonapparat bis hin zu Haushaltgeräten. In all diesen Anwendungen kann der gleiche Micro-Controller eingesetzt werden. Die unterschiedlichen Geräteanforderungen werden durch anwendungsspezifische Software und durch spezielle Interfaceschaltungen (Sensoren, Aktoren) abgedeckt.

Bei allen modernen Micro-Controller findet man Timersysteme, Ereigniszähler, Interfaces für verschiedene Schnittstellen, Analog-Digitalwandler, Pulsweitenmodulatoren, Pulsakkumulatoren, Watchdog-Systeme sowie digitale Ports.

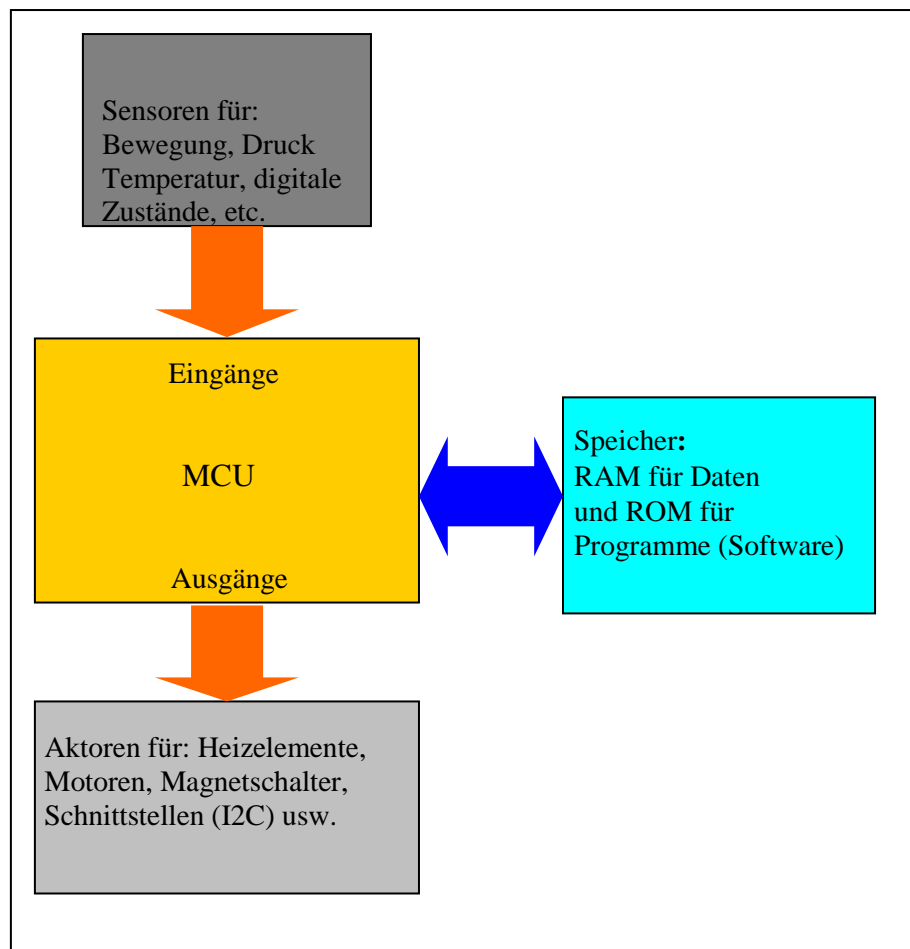


Abb.7 Prinzipieller Aufbau eines Micro-Controllers

4.1.2 Aufbau eines Micro-Controllers

Bei fast allen Micro-Controllern findet man eine ähnliche Aufbaustruktur. Der zentrale Teil ist die CPU (Central Processing Unit). Die CPU interpretiert den Programmcode, steuert den Programmablauf und führt mit ihrem Rechenwerk, der ALU (Arithmetic-Logic-Unit), arithmetische und logische Operationen durch. Nicht alle CPU's können mit der gleichen Programmiersprache programmiert werden. Je nach Fabrikat und Typ weichen die Programmiersprachen stark voneinander ab.

4.1.3 CPU - Central Processing Unit

Die CPU besteht aus dem Rechenwerk (ALU Arithmetic-Logic-Unit und Akkumulator), dem Steuerwerk (Control Unit) und verschiedenen Registern. Die CPU führt schrittweise das im Speicher vorgegebene Programm aus. Dabei übernimmt das Rechenwerk die Durchführung der arithmetischen und logischen Verknüpfungen. Der Programmzähler zeigt immer auf die Speicherstelle mit dem nächsten auszuführenden Programmcode. Die CPU hat auch Zugriff auf die System-Busse, welche außerhalb des Mikro-Controllers zusätzliche Bauteile, wie Speicher oder spezielle I/O Bausteine, bedienen.

Adress-Bus:

Die CPU bestimmt über den Adress-Bus die Speicherstelle, den Ein-Ausgabe-Port oder das Register einer anderen Peripherieschaltung des Systems. Nur die CPU kann Adressen ausgeben. Dieser Bus ist unidirektional, d.h., die Signale laufen nur in einer Richtung.

Daten-Bus:

Auf dem Datenbus werden die Daten zwischen externen Bauteilen und der CPU verschoben. Als Daten versteht man sowohl Programmcode als auch Ein- und Ausgabedaten der Peripheriebausteine. Der externe Teilnehmer wird über die, auf dem Adressbus ausgegebene Adresse bestimmt. Die Daten müssen in beiden Richtungen transportiert werden. Der Datenbus ist daher bidirektional ausgelegt.

Control-Bus:

Mit verschiedenen Signalen steuert die CPU über diesen Bus die übrigen System-Bausteine. So wird z.B. mit dem Read/Write-Signal festgelegt, in welcher Richtung der Datentransfer auf dem Daten-Bus erfolgt.

4.1.4 MCU Funktionen

I/O-Ports:

Ports sind digitale Schnittstellen, die als Ein- oder als Ausgang den Micro-Controller mit, wie in unserem Fall, mit der Schaltung verbinden. In den meisten Fällen haben Ports acht Leitungen die mit meistens P0..P7 bezeichnet werden. Falls zum Port ein Datenrichtungsregister (DDR) gehört, sind die Leitungen einzeln, wahlweise als Ein- oder als Ausgang programmierbar. Portleitungen können mehrere verschiedene Funktionen haben. So ist es möglich, Timer Ein- und Ausgänge über Portleitungen zu schalten.

Serielle Schnittstellen:

Für die Kommunikation mit Datenendgeräten sind zwei serielle Schnittstellen (SCI Serial Communications Interface) vorgesehen. Diese Schnittstellen können nur über einen externen Treiber an ein Datenendgerät angeschlossen werden. Zur Kommunikation mit anderen Prozessoren und Peripherieschaltungen verfügt der Controller zudem über ein schnelles serielles Interface (SPI Serial Peripheral Interface).

Analog-Digital-Wandler:

Analoge Signale können mit dem Analog-Digitalwandler ADC über Kanäle digitalisiert werden. Mit den beiden Referenzeingängen V_{rh} und V_{rl} kann der Spannungsbereich gewählt werden, in welchem die Digitalisierung erfolgen soll.

Timer-System:

Das universelle Timersystem des Micro-Controllers basiert auf einem Timer-Counter mit einem programmierbaren Vorteiler.

Jeder Timer-Kanal kann den Zeitpunkt eines externen Ereignisses speichern (IC Input-Capture), an einem Ausgang programmierbare Zeitintervalle generieren (OC Output-Compare) oder als Pulsweitenmodulator (PWM) arbeiten.

Die Input-Capture-Funktion (IC) wird ausgelöst, wenn sich am entsprechenden Timer-Kanal der Eingangszustand ändert. Die IC-Funktion kann so programmiert werden, dass sie auf eine positive, eine negative oder auf beide Flanken des Eingangssignals reagiert. Tritt das programmierte Ereignis auf, wird der Inhalt des Systemzählers in ein spezielles Register kopiert. So kann der genaue Zeitpunkt dieses Ereignisses festgehalten werden.

Zu jeder Output-Compare-Funktion (OC) gehört ein Register, welches durch den Anwender programmiert werden kann. Sobald der Systemzähler den Wert erreicht, der im OC-Register gespeichert ist, können Ereignisse an den Timer-Ausgängen ausgelöst werden. Die OC-Funktion eines bestimmten Timer-Kanals kann auch einen Reset des Timer-Counters bewirken.

Jeder Timer-Kanal lässt sich auch einzeln als Pulsweitenmodulator (PWM) programmieren. Bei einem PWM kann die Einschaltdauer eines Rechtecksignals mit konstanter Frequenz softwaremäßig verändert werden. So lässt sich die Leistung an einem Verbraucher mit einem hohen Wirkungsgrad über das Tastverhältnis steuern (Lampe dimmen, Motor steuern, usw.).

Watch Dog:

Der COP-Watchdog (COP Computer Operating Properly) überprüft, ob der Programmablauf des Controllers richtig funktioniert. Dazu muss die vom Anwender geschriebene Software den Watchdog innerhalb einer vorgegebenen Zeit zurücksetzen. Wird der Watchdog nicht zurückgesetzt, löst er einen System-Reset aus und bewirkt damit, dass die Software neu gestartet wird. Ein 'Systemabsturz' kann so automatisch aufgehoben werden.

Interrupts:

Bestimmte Ereignisse (Timer, externe Signale, an der seriellen Schnittstelle eingetroffene Daten usw.) können Interrupts auslösen. Ein Interrupt unterbricht das Hauptprogramm und bewirkt, dass ein spezielles Unterprogramm ausgeführt wird. Sobald dieses Unterprogramm beendet ist, wird das Hauptprogramm an der unterbrochenen Stelle wieder weitergeführt. Der Anwender kann für alle Interrupts eigene Programme verfassen.

RAM:

Das interne RAM (Random Access Memory Schreib- und Lesespeicher) des Controllers steht dem Anwender zur Verfügung. Daten im RAM gehen bei einem Unterbruch der Speisespannung verloren.

EEPROM:

Das interne EEPROM (elektrisch löschbarer Lesespeicher) kann Daten oder Programmcode speichern, die auch bei einem Unterbruch der Speisespannung erhalten bleiben. EEPROM's können durch die Software bitweise gelöscht und neu programmiert werden. Die Programmierung nimmt allerdings sehr viel Zeit in Anspruch und kann nur etwa 10'000 mal erfolgen.

4.1.5 AT91SAM7S256

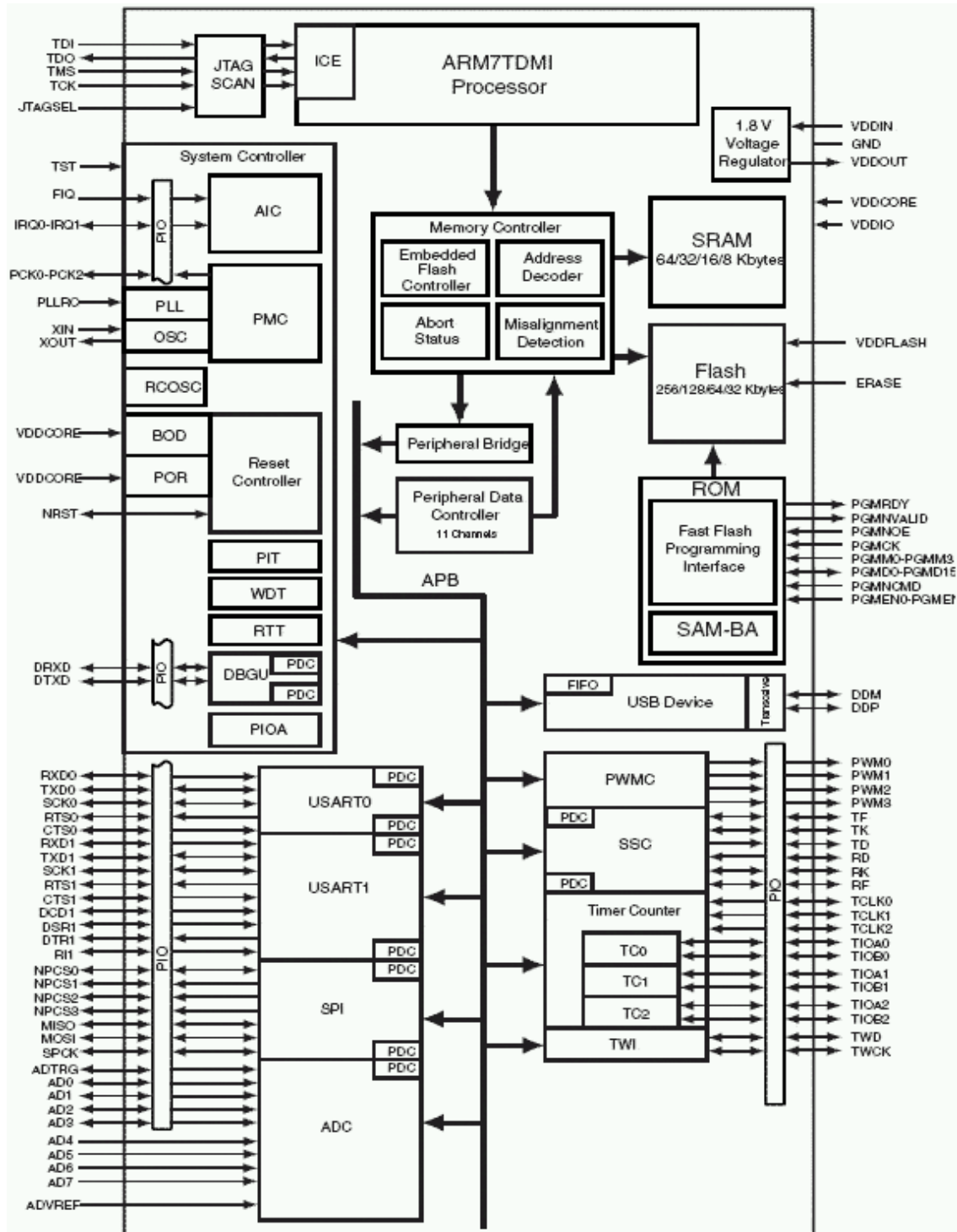


Abb.8 Schaltplan des Micro-Controllers

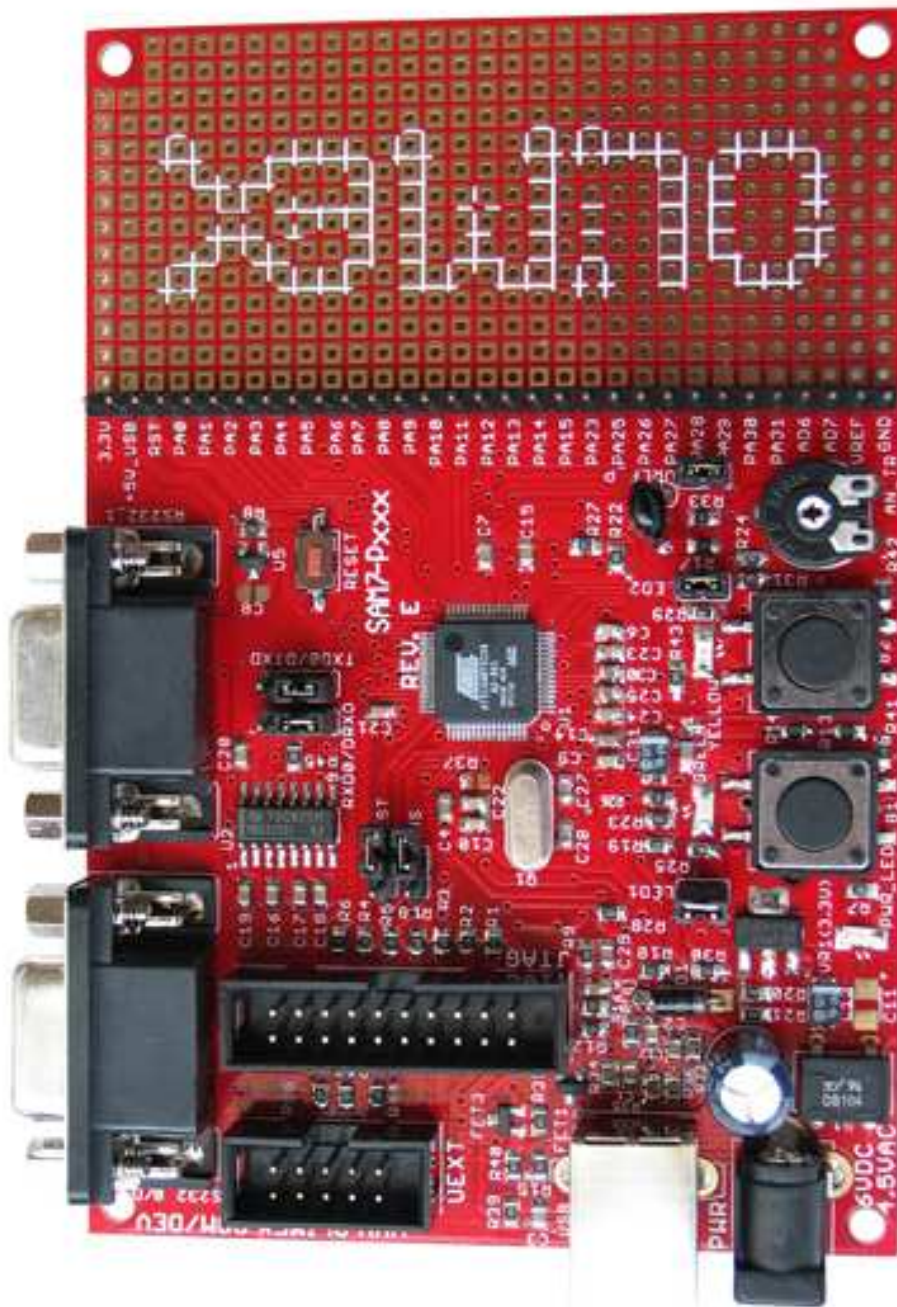


Abb.9 Der Micro-Controller SAM7S256

4.2 Der Compiler

Ein Compiler ist ein Programm, das den Quelltext eines anderen Programms, das in einer bestimmten Programmiersprache vorliegt, für den Computer in verständliche Zeichenfolgen übersetzt.

Diese Übersetzung erfolgt in 3 Phasen:

1. Überprüfung der korrekten Verwendung von Schlüsselwörtern und Formulierung, Entfernung überflüssiger Kommentare.
2. Überprüfung auf korrekte Verwendung der Datentypen sowie der korrekten Syntax.
3. Bei der Codegenerierung werden dann die den Befehlen der Programmiersprache sinngleichen Maschinencodebefehle eingesetzt.

Zwischendurch werden immer wieder Optimierungsphasen eingeschoben, wobei der Compiler dann versucht, den bisher erzeugten Code nach bestimmten Kriterien wie z.B. Dateigröße oder Geschwindigkeit zu modifizieren. Nach dem der Compiler seine Arbeit getan, hat fügt der Linker die einzelnen Dateien zu einem Programm zusammen.

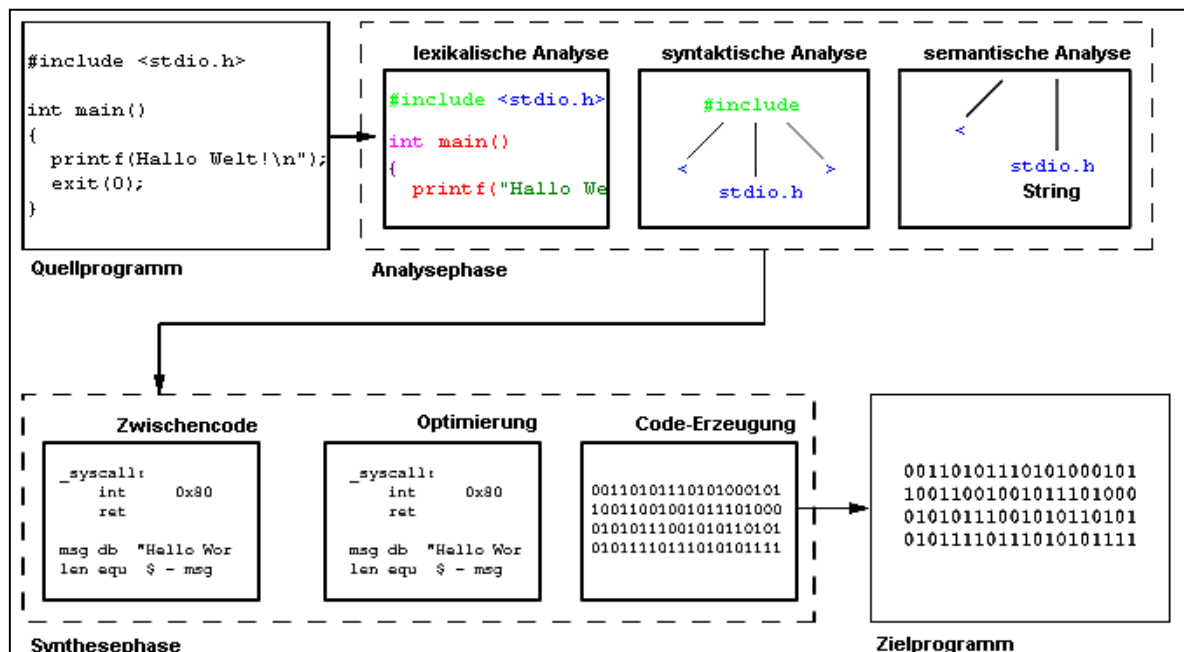


Abb.10

Code Optimierung des Compilers

4.3 *Der Debug-Anschluss*

Für das Beladen des Micro-Controllers haben wir den Debug-Anschluss J-Link-ARM-KS von IAR Systems benutzt.

4.4 *Das Relais*

Funktionsweise des Relais

Ein Relais ist ein Schalter, der nicht von Hand, sondern mit Hilfe eines Elektromagneten betätigt wird. Es besteht aus zwei getrennten Stromkreisen. Der erste Stromkreis wird als Steuerstromkreis und der zweite als Arbeitsstromkreis bezeichnet. Wird der Steuerstromkreis über einen Schalter (S1) geschlossen, dann zieht der Elektromagnet (bestehend aus einer Spule mit Eisenkern) den Schalter (S2) im Arbeitskreis an und der zweite Stromkreis ist ebenfalls geschlossen. Wird der erste Schalter S1 geöffnet, dann lässt der Magnet den Schalter S2 los, und der Arbeitsstromkreis ist unterbrochen.

Das Relais bietet somit die Möglichkeit, mit kleinen Spannungen, z.B. Batteriespannungen, Stromkreise mit hohen Spannungen und Strömen zu steuern. Auch in vielen elektrischen Geräten und Maschinen findet man Relais. Wir verwenden es, um die einzelnen Fahrrichtungen zu steuern

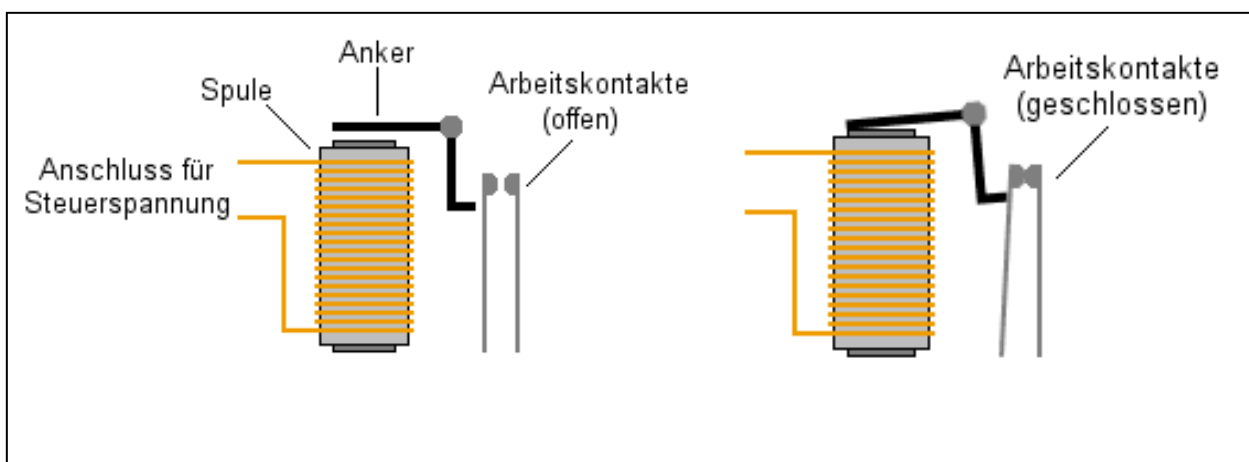


Abb.11 Schaltung des Relais

4.5 **Das IC**

Das IC, ein integrierter Schaltkreis, ist eine Zusammenfassung von mehreren Widerständen und Transistoren. Für unsere Schaltung haben wir ein DS3668 Quad Fault Protected Peripheral Driver IC genommen, weil es in der Lage ist, die Relais anzusteuern.

4.6 **Der Widerstand**

Ein Widerstand ist in der Regel da, um den elektrischen Strom zu verringern. Wir jedoch, haben den elektrischen Widerstand benutzt, damit die Leitungen relativ stabil bleiben.

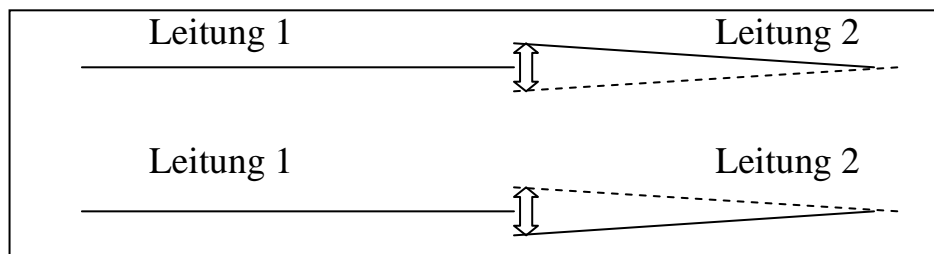


Abb.12 Die Leitungen sind instabil.

4.7 **Der Transistor**

Für unser Projekt haben wir einen PNP-Transistor genommen. Der Oberbegriff dafür ist der Bipolare Transistor.

Der PNP-Transistor besteht aus zwei positiv leitenden Schichten. Dazwischen liegt eine dünne negativ leitende Schicht.

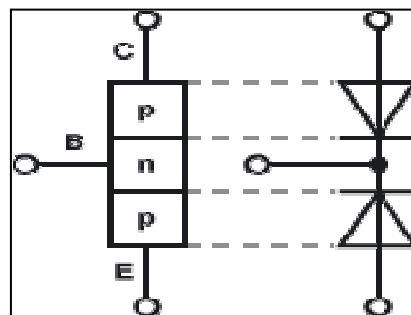


Abb.13 Der PNP-Transistor

4.8 *Die Leuchtdiode*

Die Leuchtdiode, kurz LED, ist eine kleine Lampe. Fließt durch die LED Strom in Durchlassrichtung, so leuchtet diese auf. Wir benutzen LED's, damit wir überprüfen können, ob unsere Schaltung funktioniert.

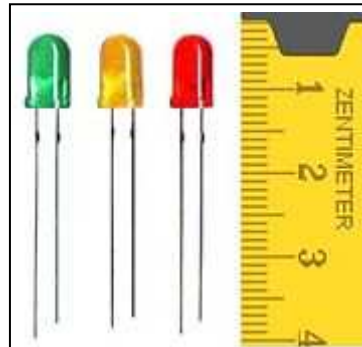


Abb.14 LED's

4.9 *Der Taster*

Der Taster ist ein Bedienungselement, welches durch Drücken betätigt wird und danach in die Ausgangslage zurückkehrt. Hierfür wird meist eine mechanische Feder eingesetzt. Anders als beim Schalter (der in der jeweiligen Position verbleibt) kann man nicht optisch erkennen, ob eine Taste betätigt worden ist. Erst an der nachfolgenden Wirkung ist das erkennbar.

4.9.1 *Das „Prellen“*

Das Prellen und die damit verbundenen Schwierigkeiten sind nicht besonders förderlich für unser Projekt. Jedesmal, wenn man eine Taste drückt, gibt sie einen Wert ab.

Wenn jedoch ein Taster prellt, gibt dieser mehrere Werte ab.

Dies ist soweit nicht schlimm, aber wenn der Taster zu oft prellt, sind dies natürlich höhere Werte und somit auch mehr Daten die der Micro-Controller speichern muss.

5. Unsere Arbeitsweise



5.1 *Danksagung*

Wir bedanken uns für die selbstlose Unterstützung von Manfred Tollning, ohne dessen Hilfe wir niemals soweit gekommen wären:

Wir sagen Danke!!!

6. Fazit

Wir haben erreicht, was wir uns vorgenommen haben. Sicher gibt es noch viele Punkte an denen man Verbesserungen vornehmen könnte, aber es fehlte uns schlicht die Zeit.

Am Ende entstand eine funktionierende Schaltung inkl. Software für das Aufzeichnen und Abspielen des Fahrweges.

Die Arbeit gestaltete sich sehr spannend. Wir hatten viele Probleme und viele Rückschläge einstecken müssen. Aber es hat sich gelohnt. Wir haben Einblicke in Themen erhalten, die uns nur wenig vertraut waren. Die Arbeit hat viel Spaß gemacht, war aber leider in der Zeit zu kurz bemessen.

7. Quellenverzeichnis

1. <http://www.atmel.com/>
2. <http://www.sn.schule.de/~gyfloeha/pixel/lex01/code.html>
3. <http://de.wikipedia.org>
4. <http://www.arndt-bruenner.de>
5. <http://www.lemps.ch/>
6. <http://www.uni-koblenz-landau.de/cms/>

8. Bestätigung

Wir versichern, dass wir die vorliegende Projektarbeit selbständig verfasst und keine anderen als die angegebenen Hilfsmittel verwendet haben.

Oliver Tollning: _____

Jannes Schlüter: _____