Robotername : MadMax


Inhalt:

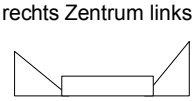

## 1. Aufgabenstellung

In diesem Projekt soll ein fussballspielender Roboter auf Basis des 8-Bit Aksen-Boards gebaut und programmiert werden. Er darf maximal so groß wie ein DIN-A4-Blatt sein. Am Ende des Semesters soll er gegen die anderen Roboter antreten.

## 2. Startentscheidung

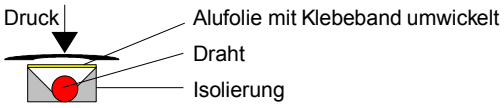
Zu Beginn war eine Designentscheidung zwischen dem onmidirektionalen und dem klassischen Antrieb zu treffen. Aufgrund der Vorteile des omnidirektionalen Antriebs (höhere Wendigkeit und „cool“) haben wir uns für diese Antriebsform entschieden.

## 3. Sensoren




diese Anordnung erlaubt ein dichtes Heranfahren an die Wand

- 3 Sharpsensoren
- 5 IR-Sensoren (vorne links,rechts, hinten links,rechts, hinten)
- 1 Sensorenphalanx (Ball vorne)
- 1 Sensor (hat Ball)
- Bumper (Rohrisolierung mit „Draht“ und Alufolie) geschützt durch Klebeband




## 4. Anzeige und Steuerung

- Statusanzeige erfolgt über 4 LED (zur Zeit als Lauflicht missbraucht)
- Die Steuerung erfolgt über das Lego-Männchen (nicht sehr zuverlässig ;-) )

## 5. Positives

- Der Bot funktioniert vollständig :-)
- der Bumper regiert erst bei hohem Druck, daher eher träges Ausweichen (siehe 8.)
- Torschussvorrichtung per Direktantrieb über Servo/Zahnrad (die Befestigung des Servos erfolgt mit Kabelbindern, die bei Überlast etwas nachgeben, so dass eine Rutschkupplung entsteht).

## 6. Negatives

- das Getriebe des Omi-Antriebes ist unzuverlässig
- der Bumper regiert erst bei hohem Druck
- Bumper ist nicht durchlaufend
- ein Verstärkerausgang der Sharpsensoren ist defekt, daher nur 3 Sensoren möglich
- der Bot neigt dazu mit dem Ball zu spielen

## 7. Software

- Subsumption
- Erkennung der optimalen Drehrichtung zum gegnerischen Tor durch Erkennung des eigenen Tors

## 8. Das Turnier

8.1 Das Verhalten der Gegner

Die gefährlichsten Gegner besaßen folgende Strategie : die Bots griffen mit extrem hohen Tempo den Ball an. Hatten sie diesen, bestand ihre Strategie daraus, auf das Tor zuzufahren. Diese Bots besaßen außerdem noch einen Ballkäfig, der die IR-Strahlung abschirmte. Er wurden nur Ballsensoren benutzt. Sharps und Bumper kamen nicht zum Einsatz (weitere wirkliche Gegner gab es nicht für unsere Bots).

8.2 Probleme unserer Bots

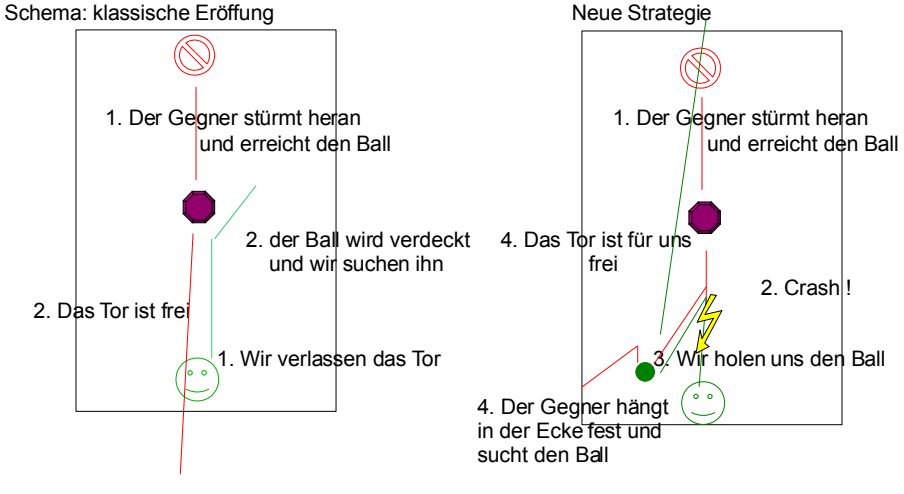
Aufgrund der schwachen Motorisierung hatten unsere Bots keine Chance diese Bots einzuholen. Auch machten sich die Bumper negativ bemerkbar (unsere Bots wichen sehr schnell aus).

8.3 Stärken unserer Bots

Wir sind beweglicher, haben eine genauere Sensorik, können Wände erkennen und unsere Software ist besser (gilt für fast alle HAW-Bots).

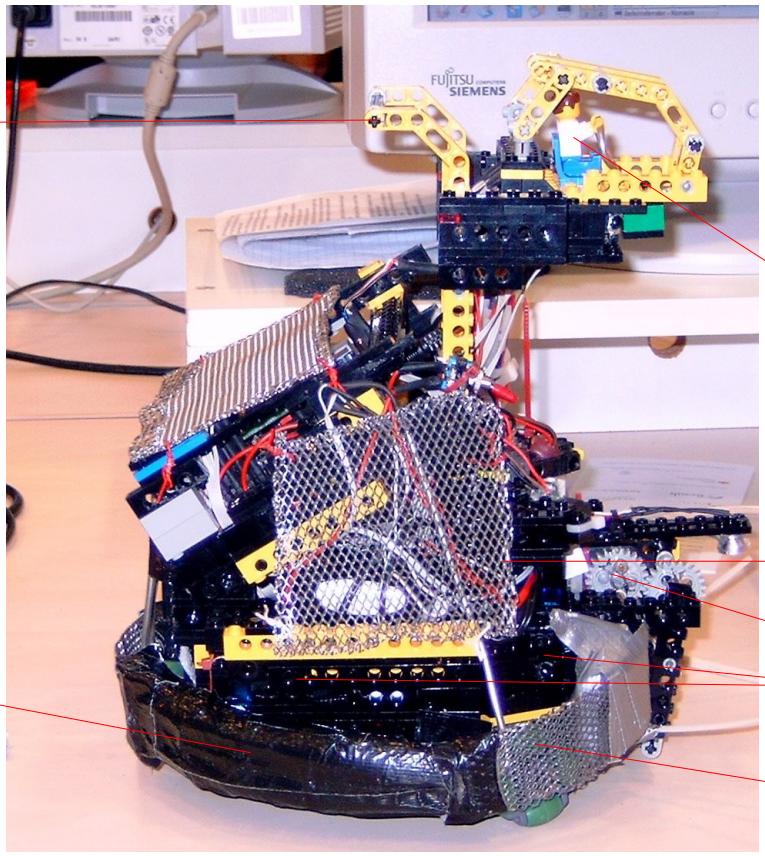
8.4 Strategien um diese „Killer auszuschalten“

1. In der Startphase dürfen die Bots den Torraum nicht allzu weit verlassen (das Tempo kann bei MadMax in der Ballsuchphase reduziert werden). Hat der Gegner nämlich den Ball, so wird er versuchen auf das Tor zu stürmen. Das klappt nicht, weil hier unser Bot steht. Im Idealfall wird nun der Gegner zur Seite abgedrängt (sofern die Bumper nicht allzu empfindlich sind). Jetzt kommt unsere Chance: wir sind wendiger. Der andere Bot gibt nach 3 Sekunden den Ball frei bzw. wird durch die Wand (die er nicht erkennt) aufgehalten. Dank unserer Wendigkeit klauen wir nun den Ball. Während der andere Bot noch mit der Wand und mit unserem Bot als Hindernis kämpft, nutzen wir die Gunst der Stunde, um langsam auf das Tor zu zielen, uns in Position zu bringen und zu einem Schuss anzusetzen. Treffen wir nicht genau, können wir uns in aller Ruhe um den Ball kümmern, während der andere Bot immer noch versucht mit uns und den Wänden zu kämpfen (die er ja nicht sieht :-) ). Mit diesem Verfahren konnten wir im „kleinen Finale“ gegen den anderen Bot ein 3:0 erreichen (es war so eine typische Situation David gegen Goliath - und der Listigere hat gesiegt :-) ). In der Vorrunde versuchten wir dem anderen Bot Paroli zu bieten und wurden mit 0:5 besiegt.
2. Wir könnten selber schneller werden (setzt neue Motoren voraus), damit wir nicht mehr über den Haufen gefahren werden

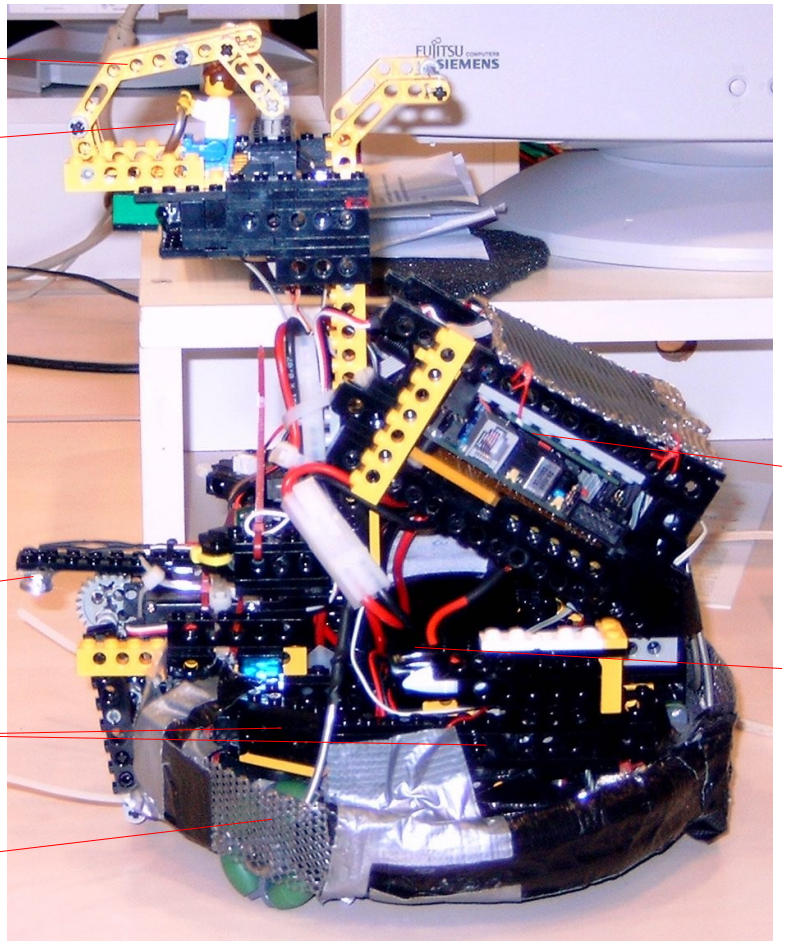


## 9. Bilder

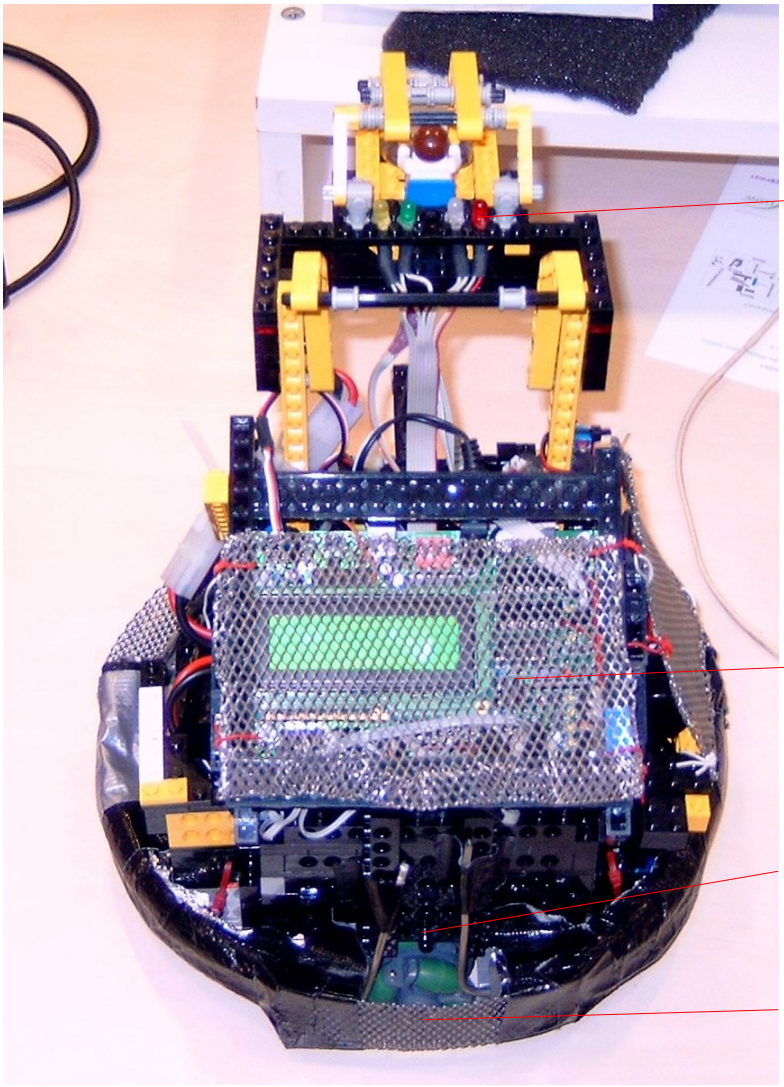
Seite:



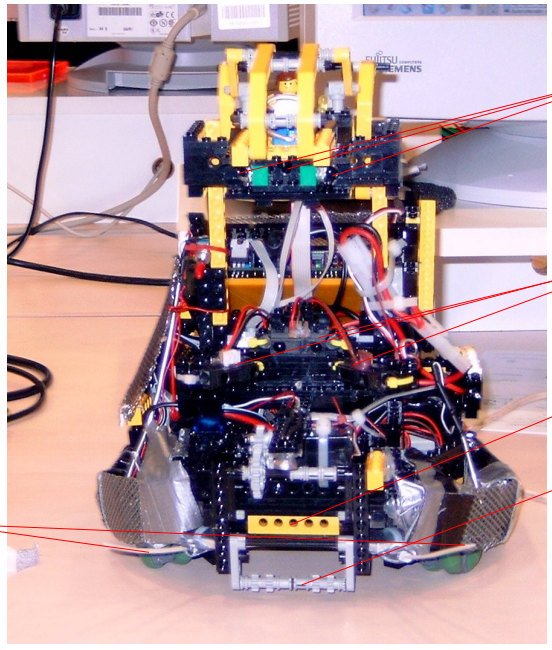

Andere Seite :



Ansicht von oben:




Ansicht von vorne:



## 10 Code

Der Code dient nicht dem Kopieren sondern soll nur eine Hilfe darstellen, wie ein Programm aussehen könnte (nebenbei: er glänzt nicht durch Schönheit ).

Zu beachten ist hierbei, dass die Verwendung der globalen Variable die Aufgabe hat den Stack zu entlasten, da es gelegentlich zu einen Stack-Overflow kommt, wenn Unterprogramme mit Parametern aufgerufen werden.

PS: ein Ausführen des Codes in einen anderen Bot wird seltsames Verhalten zur Folge haben, da alle Einstellungen für MadMax optimiert sind.

```
#include <stdio.h>
#include <regc515c.h>
#include <stub.h>
//LEDS
#define rot 3
#define weiss 2
#define gruen 1
#define gelb 0
//motorpower
#define power 10
#define halfpower 5
//sharpwerte
//je hoeher desto dichter
#define sharpFront 180
#define sharpSide 190
//schwellenwerte
#define bdc_count 70
#define searchborder 250
#define runborder 100
#define schussborder 40
#define deadlockborder 100

//niche to have
    unsigned char firstcontact=0;
    unsigned char lasttorsight=0;
    unsigned char ledlight=0;
    unsigned char leddelay=0;
    unsigned char runswap=0;
    unsigned char runspin=0;
//
    unsigned char takt=0;       // takt des ziel Tores
    unsigned char owntakt=0;    // takt des eigenen Tores
    unsigned char searchswap = 0;
    unsigned char drehspin =0;
    unsigned char test_out=0; // LED Nachrichten an(1)/aus(0)
//countvariablen
    unsigned char bdc =0;  //Ball Detection Counter
    unsigned char deadlock=0;
//Bumper
    unsigned char bumper_l = 0;
    unsigned char bumper_r = 0;
    unsigned char bumper_m = 0;
//Sharpsensoren
    unsigned char frontwert_m=0;
    unsigned char frontwert_r=0;
    unsigned char frontwert_l=0;
//Ballsensoren
    unsigned char heck=0;
    unsigned char ballfront=0;
    unsigned char linksvorne=0;
    unsigned char rechtsvorne=0;
    unsigned char linksheck=0;
    unsigned char rechtsheck=0;
    unsigned char ball=0;
//Torsensoren
    unsigned char tor_l=0;
    unsigned char tor_m=0;
    unsigned char tor_r=0;
//TorCountvariablen
    char torswap=0;
    unsigned char torswap_l=0;
    unsigned char torswap_r=0;
    unsigned char stoss_c=0;

//aviod stackoverflow
    unsigned int t=0;

/*-------------------------------------*
 *      Fahrroutinen                   *
 *-------------------------------------*/
void dribble(){
        runswap++;
        if (runswap<=runborder && runspin == 0){
```

```
                motor_richtung(0,1);
                motor_richtung(1,0);
                motor_richtung(2,1);
                motor_pwm(0,power);
                motor_pwm(1,power);
                motor_pwm(2,halfpower);
                runswap = 0;
                runspin = 1;
        }else {
                motor_richtung(0,1);
                motor_richtung(1,0);
                motor_richtung(2,0);
                motor_pwm(0,power);
                motor_pwm(1,power);
                motor_pwm(2,halfpower);
                runswap = 0;
                runspin = 0;
        }
        sleep(t);
}

void vorwarts(){
        motor_richtung(0,1);
        motor_richtung(1,0);
        motor_richtung(2,0);
        motor_pwm(0,power);
        motor_pwm(1,power);
        motor_pwm(2,0);
        sleep(t);
}

void slowvorwarts(){
        motor_richtung(0,1);
        motor_richtung(1,0);
        motor_richtung(2,0);
        motor_pwm(0,7);
        motor_pwm(1,7);
        motor_pwm(2,0);
        sleep(t);
}

void schuss() {
        servo_arc(0,50);
        sleep(200);
        servo_arc(0,115);
        sleep(200);
}

void ruckwarts(){
        motor_richtung(0,0);
        motor_richtung(1,1);
        motor_richtung(2,0);
        motor_pwm(0,power);
        motor_pwm(1,power);
        motor_pwm(2,0);
        sleep(t);
}

void slowrechtsdreh(){
        motor_richtung(0,1);
        motor_richtung(1,1);
        motor_richtung(2,1);
        motor_pwm(0,7);
        motor_pwm(1,7);
        motor_pwm(2,7);
        sleep(t);
}

void slowlinksdreh(){
        motor_richtung(0,0);
        motor_richtung(1,0);
        motor_richtung(2,0);
        motor_pwm(0,7);
        motor_pwm(1,7);
        motor_pwm(2,7);
        sleep(t);
}

void rechtsdreh(){
        motor_richtung(0,1);
        motor_richtung(1,1);
        motor_richtung(2,1);
        motor_pwm(0,power);
        motor_pwm(1,power);
        motor_pwm(2,power);
        sleep(t);
}

void linksdreh(){
        motor_richtung(0,0);
```

```
        motor_richtung(1,0);
        motor_richtung(2,0);
        motor_pwm(0,power);
        motor_pwm(1,power);
        motor_pwm(2,power);
        sleep(t);
}

void balltanz_l(){   //Links
        motor_richtung(0,1);
        motor_richtung(1,0);
        motor_richtung(2,0);
        motor_pwm(0,0);
        motor_pwm(1,0);
        motor_pwm(2,power);
        sleep(t);
}

void balltanz_r(){   //Rechts
        motor_richtung(0,1);
        motor_richtung(1,0);
        motor_richtung(2,1);
        motor_pwm(0,0);
        motor_pwm(1,0);
        motor_pwm(2,power);
        sleep(t);
}

void rechts(){
        motor_richtung(0,1);
        motor_richtung(1,1);
        motor_richtung(2,0);
        motor_pwm(0,halfpower);
        motor_pwm(1,halfpower);
        motor_pwm(2,power);
        sleep(t);
}

void links(){
        motor_richtung(0,0);
        motor_richtung(1,0);
        motor_richtung(2,1);
        motor_pwm(0,halfpower);
        motor_pwm(1,halfpower);
        motor_pwm(2,power);
        sleep(t);
}

void lauflicht(){
        unsigned char ledpos=ledlight;
        if (leddelay==30) {
                leddelay=0;
                if (ledlight==4) {
                        led(0,0);
                        led(1,0);
                        led(2,0);
                        led(3,0);
                }
                if (ledlight>3) {
                        ledpos=8-ledlight;
                }
                led(ledpos,1);
                ledlight++;
                if (ledlight>8) {
                        ledlight=0;
                        //led(0,0);
                        led(1,0);
                        led(2,0);
                        led(3,0);
                }
        }
        leddelay++;
}

void suche(){
        searchswap++;
        if (searchswap>=searchborder && drehspin == 0){
                searchswap = 0;
                drehspin = 1;
                t=120;
                rechtsdreh();
        }else if (searchswap>=searchborder && drehspin == 1){
                searchswap = 0;
                drehspin = 2;
                t=120;
                linksdreh();
        }else if (searchswap>=searchborder && drehspin == 2){
                searchswap = 0;
                drehspin = 1;
                t=120;
```

```
                rechtsdreh();
        }else {
                dribble();
//              vorwarts(t);
        }
}

/*--------------------------------------*
 *                                      *
 *              KI-part                 *
 *                                      *
 *--------------------------------------*/
void sensorenEinlesen(){

//      Sensoren einlesen
        //Sharpsensoren
        frontwert_m=analog(6);
        frontwert_l=analog(7);
        frontwert_r=analog(5);
        //bumper
        bumper_l = digital_in(0);
        bumper_r = digital_in(1);
        //Ballsensoren
        linksvorne=analog(9);
        linksheck=analog(8);
        heck=analog(10);
        rechtsheck=analog(11);
        rechtsvorne=analog(12);
        ball=analog(1);
        //Frontphalanx;
        ballfront=analog(0);
        //Torsensoren
}

//      TorSensoren mit takt einlesen
void torsensoren(unsigned char takt) {
        mod_ir0_takt(takt);
        mod_ir1_takt(takt);
        mod_ir2_takt(takt);
        tor_l=mod_ir0_status();
        tor_m=mod_ir1_status();
        tor_r=mod_ir2_status();
}

/*--------------------------------------*
 *      Verhalten wenn Robo Ball hat    *
 *--------------------------------------*/
void hatBall(){
        firstcontact=1;
        torsensoren(takt);
        sleep(5);
        if (test_out) led(gruen,1);
        searchswap=0;
        drehspin=0;
        torswap=torswap_l-torswap_r;
        //hat Ball -> Counter auf max
        if (ball<100) {bdc=bdc_count;}
        //bumper active and ball
        //deadlock=0;
        if (frontwert_m>250 || frontwert_r>250 || frontwert_l>250 ) {
                if (test_out) led(weiss,1);
                if (frontwert_l<frontwert_r) {
                        t=200;
                        balltanz_l();
                        drehspin=2;
                }
                else {
                        t=200;
                        balltanz_r();
                        drehspin=1;
                }
        }
        else if (!bumper_l ){
                if (test_out) led(rot,1);
                deadlock=0;
                t=100;
                vorwarts();
                t=200;
                rechts();
        }
        else if (!bumper_r ){
                if (test_out) led(rot,1);
                deadlock=0;
                t=100;
                vorwarts();
                t=200;
                links();
        }
        //Torsuche
        else if (tor_m >3 || torswap>5) {
```

```
                if (test_out) led(gelb,1);
                deadlock=0;
                t=1;
                vorwarts();
                //dribble(1);
                torswap_l=0;
                torswap_r=0;
                torswap=0;
                stoss_c++;
                if (stoss_c > schussborder) {
                        schuss();
                        stoss_c=0;
                }
        }
        //sieht Tor auf der linken Seite
        else if (tor_l >5) {
                t=1;
                balltanz_l();
                //if (test_out) led(weiss,1);
                torswap_l++;
                deadlock++;
                if (torswap_l==torswap_r) torswap++;
        }
        //sieht Tor auf der rechten Seite
        else if (tor_r >5) {
                t=1;
                balltanz_r();
                torswap_r++;
                deadlock++;
                if (torswap_l==torswap_r) torswap++;
                //if (test_out) led(rot,1);
        }
        //hat Tor noch nicht gesehen
        else if (tor_l<5&& tor_m<5 && tor_r<5) {
                //if (test_out) led(gruen,1);
                t=5;
                if (lasttorsight && frontwert_l>=frontwert_r) balltanz_r();
                else balltanz_l();
        }
}

/*----------------------------------------*
 *      Verhalte bei der Suche nach Ball   *
 *----------------------------------------*/
void suchtBall(){
        stoss_c=0;
        torsensoren(owntakt);
        if (tor_l>tor_r) lasttorsight=1;
        else lasttorsight=0;
        if (frontwert_m>sharpFront || frontwert_r>sharpSide || frontwert_l>sharpSide ) {
                if (test_out) led(weiss,1);
                if (frontwert_l<frontwert_r) {
                        t=600;
                        linksdreh();
                        searchswap=0;
                        drehspin=1;
                }else {
                        t=600;
                        rechtsdreh();
                        searchswap=0;
                        drehspin=2;
                }
                deadlock++;
        }
        //hat Ball noch nicht
        else if (!bumper_l ){
                deadlock=0;
                searchswap = searchborder-1;
                drehspin = 2;
                t=50;
                rechts();
        }
        else if (!bumper_r ){
                deadlock=0;
                searchswap = searchborder-1;
                drehspin =1;
                t=50;
                links();
        }
        // Ball vorne
        else if (ballfront<200){
                if (test_out) led(weiss,1);
                t=1;
                vorwarts();
                //dribble(1);
        }
        // Ballsuche
        else if (linksvorne<230) {
                t=100;
                linksdreh();//50
```

```
                deadlock++;
        }
        else if (rechtsvorne<230){
                t=100;
                rechtsdreh();//50
                deadlock++;
        }
        else if (linksheck<210) {
                t=60;
                linksdreh();//60
                deadlock++;
        }
        else if (rechtsheck<210) {
                t=60;
                rechtsdreh();//60
                deadlock++;
        }
        else if (heck<210) {
                t=300;
                linksdreh();
                deadlock++;
        }
        // normal fahren
        else {
                deadlock=0;
                if (!firstcontact) {
                        t=10;
                        vorwarts();
                }
                else {
                        t=1;
                        suche();
                }
        }
}

/*--------------------------------------*
 *      Hauptroutine                    *
 *--------------------------------------*/
void roboMain(){
//      Programmschleife
        while(1) {
                lauflicht();
                if (bdc>0) bdc--;
                sensorenEinlesen();
                // Leds zuruecksetzten
                //if (test_out) {
                //      led(gelb,0);
                //      led(gruen,0);
                //      led(weiss,0);
                //      led(rot,0);
                //}
        /*--------------*
         *   KI Beginn  *
         *--------------*/
                if (deadlock>deadlockborder) {
                        if (test_out) led(rot,1);
                        deadlock=0;
                        t=150;
                        ruckwarts();
                        if (frontwert_l > frontwert_r){
                                t=200;
                                rechts();
                        }else{
                                t=200;
                                links();
                        }
                }
                // hat Ball
                else if (ball<100 || bdc>0) {
                        hatBall();
                }
                //sucht Ball
                else suchtBall();
        }
}

void roboDebugSuche(){
        int i;
        for(i=0;i<100;i++){
                torsensoren(owntakt);
                lcd_cls();
                lcd_ubyte(tor_l);
        }
        torsensoren(takt);
        sleep(10);
        if (tor_l>5) lasttorsight=1;
        else lasttorsight=0;
        lcd_ubyte(lasttorsight);
        lcd_puts(" ");
```

```
        while(1) {
                // Leds zuruecksetzten
                if (test_out) {
                        led(gelb,0);
                        led(gruen,0);
                        led(weiss,0);
                        led(rot,0);
                }
        /*--------------*
         *   KI Beginn  *
         *--------------*/
                sensorenEinlesen();
                lcd_cls();
                lcd_ubyte(ball);
                sleep(500);
                if (deadlock>deadlockborder) {
                        if (test_out) led(rot,1);
                        deadlock=0;
                        t=150;
                        ruckwarts();
                        if (frontwert_l > frontwert_r){
                                t=200;
                                rechts();
                        }else{
                                t=200;
                                links();
                        }
                }
                hatBall();
        }
}

void elfmeter(){
        stoss_c=0;
        torsensoren(owntakt);
        if (tor_l>tor_r) lasttorsight=1;
        else lasttorsight=0;
        if (frontwert_m>sharpFront || frontwert_r>sharpSide || frontwert_l>sharpSide ) {
                if (test_out) led(weiss,1);
                if (frontwert_l<frontwert_r) {
                        t=600;
                        slowlinksdreh();
                        searchswap=0;
                        drehspin=1;
                }else {
                        t=600;
                        slowrechtsdreh();
                        searchswap=0;
                        drehspin=2;
                }
                deadlock++;
        }
        //hat Ball noch nicht
        else if (!bumper_l ){
                deadlock=0;
                searchswap = searchborder-1;
                drehspin = 2;
                t=50;
                rechts();
        }
        else if (!bumper_r ){
                deadlock=0;
                searchswap = searchborder-1;
                drehspin =1;
                t=50;
                links();
        }
        // Ball vorne
        else if (ballfront<200){
                if (test_out) led(weiss,1);
                t=1;
                slowvorwarts();
                //dribble(1);
        }
        // Ballsuche
        else if (linksvorne<230) {
                t=100;
                slowlinksdreh();//50
                deadlock++;
        }
        else if (rechtsvorne<230){
                t=100;
                slowrechtsdreh();//50
                deadlock++;
        }
        else if (linksheck<210) {
                t=60;
                slowlinksdreh();//60
                deadlock++;
        }
```

```
        else if (rechtsheck<210) {
                t=60;
                slowrechtsdreh();//60
                deadlock++;
        }
        else if (heck<210) {
                t=300;
                slowlinksdreh();
                deadlock++;
        }
        // normal fahren
        else {
                deadlock=0;
                if (!firstcontact) {
                        t=10;
                        slowvorwarts();
                }
                else {
                        t=1;
                        suche();
                }
        }
}

void elfmeterMain(){
//      Programmschleife
        while(1) {
                lauflicht();
                if (bdc>0) bdc--;
                sensorenEinlesen();
                // Leds zuruecksetzten
                //if (test_out) {
                //      led(gelb,0);
                //      led(gruen,0);
                //      led(weiss,0);
                //      led(rot,0);
                //}
        /*--------------*
         *   KI Beginn  *
         *--------------*/
                if (deadlock>deadlockborder) {
                        if (test_out) led(rot,1);
                        deadlock=0;
                        t=150;
                        ruckwarts();
                        if (frontwert_l > frontwert_r){
                                t=200;
                                rechts();
                        }else{
                                t=200;
                                links();
                        }
                }
                // hat Ball
                else if (ball<100 || bdc>0) {
                        hatBall();
                }
                //sucht Ball
                else elfmeter();
        }
}


void AksenMain(void){
        takt = dip()+4;// 5=125Hz  4=100Hz
        owntakt=5;
        if (takt==5)owntakt=4;
        //lcd_ubyte(takt);
        if (takt<6) roboMain();
        if (takt==6 ||takt==7){
                //led(gelb,1);
                takt=takt-2;
                if (takt==5)owntakt=4;
                roboDebugSuche();
        }
        if (takt==8 ||takt==9){
                //led(gelb,1);
                takt=takt-4;
                if (takt==5)owntakt=4;
                elfmeterMain();
        }
        else {  led(gelb,1);
                led(gruen,1);
                led(weiss,1);
                led(rot,1);
        }
}
```