**Lastenheft Triple Plattform**

**Inhalt**

[1 Zweck des Dokuments 3](#_Toc424906389)

[2 System 3](#_Toc424906390)

[3 Hardware 3](#_Toc424906391)

[3.1 Allgemein 3](#_Toc424906392)

[3.2 Analog Eingänge 3](#_Toc424906393)

[3.3 Digital Ausgang 4](#_Toc424906394)

[3.4 PWM 4](#_Toc424906395)

[4 Software 4](#_Toc424906396)

[4.1 GUI 4](#_Toc424906397)

[4.2 Switch Management 5](#_Toc424906398)

[4.3 Position Management 5](#_Toc424906399)

[4.4 Offset 5](#_Toc424906400)

[4.5 Adjust Plattform 5](#_Toc424906401)

[4.6 Berechnung des Performance-Index 6](#_Toc424906402)

[4.7 Arduino Board 6](#_Toc424906403)

[5 Anhang 7](#_Toc424906404)

[5.1 Lichtschranke-OP Schaltung 7](#_Toc424906405)

[5.2 Schaltplan LED Ansteuerung 8](#_Toc424906406)

[6 Änderungshistorie 8](#_Toc424906407)

# Zweck des Dokuments

Dieses Dokument dient zur Dokumentation des Versuchs Triple Plattform. Die Software zu diesem Versuch stammt aus einem ähnlichen Projekt und wurde auf das Experiment angepasst.

# System

Das System soll aus drei Plattformen bestehen an denen jeweils eine Fliege befestigt ist. Die Fliege kann sind auf der Plattform in die Richtung links und rechts bewegen. Die Bewegung der Fliege wird mit Hilfe von einer Lichtschranke erfasst und an einen analogen Port des Labjack U12 weitergegeben. Der Versuch unterteilt sich in 10 Zeiteinheiten, innerhalb dieser Einheiten kann definiert werden ob die jeweilige Plattform Master, Yoked oder Disable ist. Über das LED Management kann definiert werden welche LED Farbe geschalten werden soll sobald sich die Fliege im rechten oder linken Bereich befindet. Jede Plattform kann mit zwei unterschiedlichen LED Farben beschaltet werden.

# Hardware

## Allgemein

Als Hardware wird ein USB-Messmodul von Labjack verwendet. Der LabJack U12 USB hat folgende Ausstattung.

* 8 Single-Ended Analog Eingänge
* 2 Analog Ausgänge
* 20 Digital Ein-/ Ausgänge

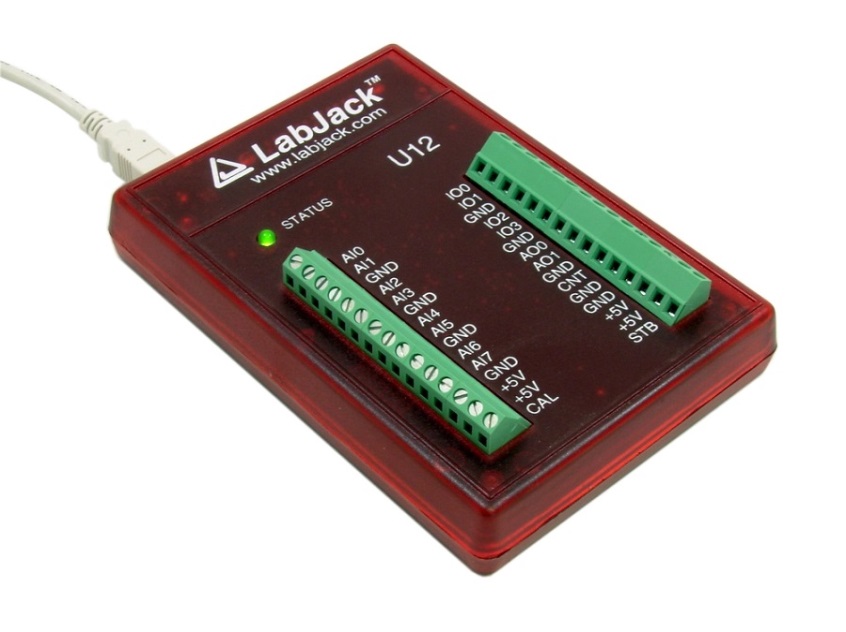


Abbildung 1

## Analog Eingänge

Die Lichtschranke wird mit Hilfe einer OP-Schaltung ausgewertet. Die Unterlagen befinden sich im Anhang. Die Ausgangsspannung des OPs geht an den jeweiligen Single-Ended Eingang des Messmoduls.

|  |  |
| --- | --- |
| Plattform 1 | AI0 |
| Plattform 2 | AI2 |
| Plattform 3 | AI4 |

## Digital Ausgang

Die LEDs für die drei Plattformen werden mit Hilfe der digitalen Ausgänge des Labjack U12 genutzt.

|  |  |
| --- | --- |
| Plattform 1-LED 1 | D0 |
| Plattform 1-LED 2 | D1 |
| Plattform 2-LED 1 | D2 |
| Plattform 2-LED 2 | D3 |
| Plattform 3-LED 1 | D4 |
| Plattform 3-LED 2 | D5 |

## PWM

Die Ansteuerung der LEDs erfolgt mit Hilfe eines PWM Signals. Die PWM wird mit Hilfe von einem Arduino UNO Boards erzeugt.

|  |  |
| --- | --- |
| Input | Output |
| PIN 2 | PIN 3 |
| PIN 4 | PIN 5 |
| PIN 7 | PIN 6 |
| PIN 8 | PIN 9 |
| PIN 12 | PIN 10 |
| PIN 13 | PIN 11 |

# Software

## GUI

Die Software soll über eine GUI gesteuert werden. Abbildung 2 zeigt ein Beispiel für das Layout der Software.

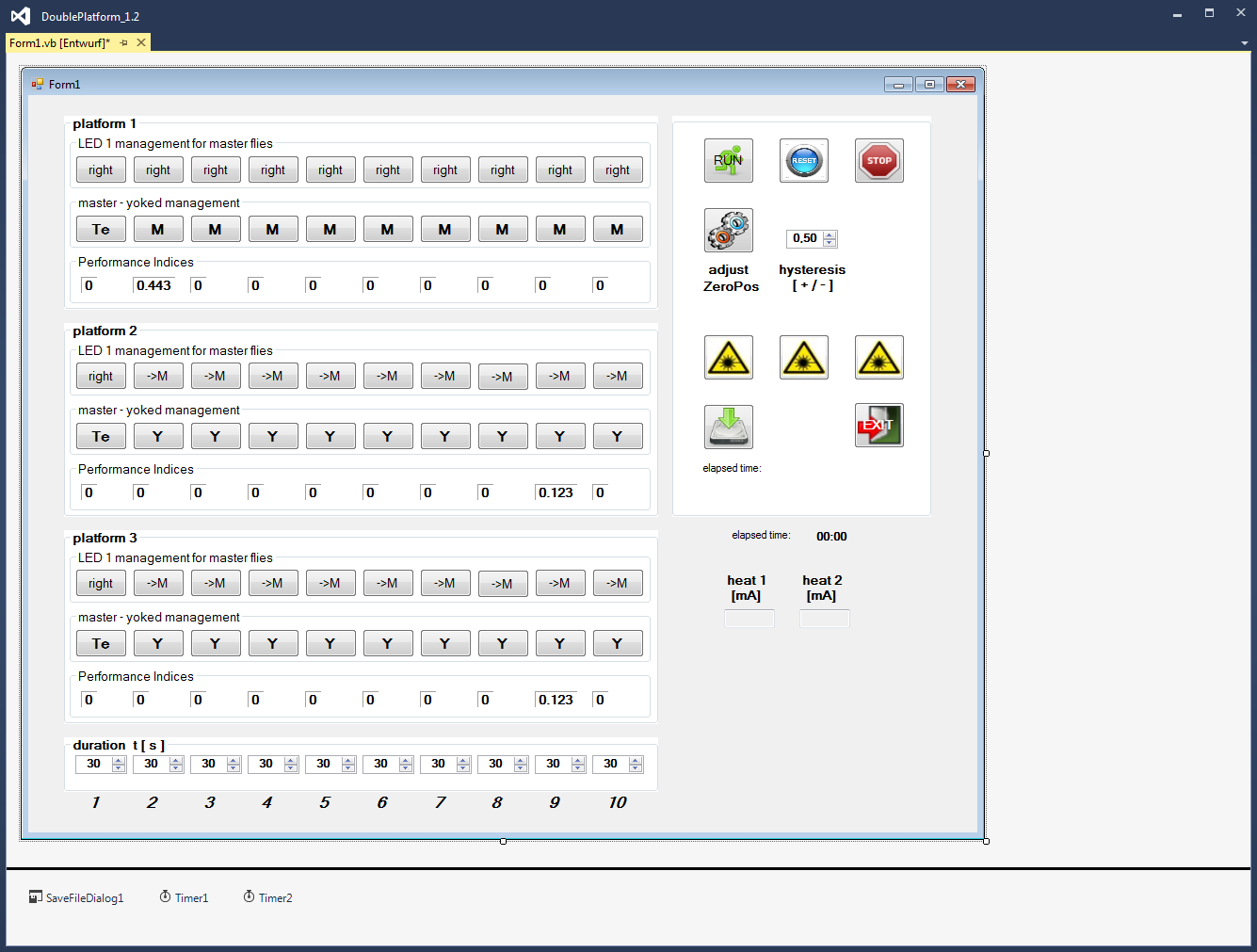


Abbildung 2

## Switch Management

Alle drei Plattformen können als Master, Yoked oder Disable definiert werden. Mit der Software ist es möglich mit mehr als einen Master zu arbeiten. Der erste Master wird mit „M“ für First Level Master gekennzeichnet, weitere Master im System werden mit „M\_S“ für Second Level Master gekennzeichnet. Plattformen die das Attribut „Y“ besitzen sind vom Master „M“ abhängig. Der Nutzer ist für die korrekte Konfiguration des „Y“ zuständig. Der „Y“ Zustand wird nicht durch Logik überprüft, bei der Konfiguration der drei Plattformen mit dem Zustand „Y“ liefert die Software ungültige Werte.

## Position Management

Besitzt eine Plattform das Attribut „M“, „M\_S“ oder „Te“ kann definiert werden ob die LED1 aktiviert wird wenn die Fliege im rechten oder linken Bereich befindet. Die LED2 wird komplementär geschaltet.

## Offset

Die Software soll einen Schieberegler besitzen, damit der Offset jeder Plattform angepasst werden kann. Die Werte die um den Offset korrigiert wurden und die Rohwerte sollen in einem Diagramm angezeigt werden.

## Adjust Plattform

Mit Hilfe einer Schaltfläche soll es möglich sein den Offset zu justieren. Während der Justage dürfen die Ausgangstreiber der LEDs nicht angesteuert werden.

## Berechnung des Performance-Index

Die Berechnung des Performance-Index erfolgt nach folgender Formel.

tc und th werden nach folgendem Algorithmus erhöht.

|  |  |  |
| --- | --- | --- |
| Platfrom1\_led1\_onoff | True | th1 = th1 + 1 |
| Platfrom1\_led2\_onoff | True | tc1 = tc1 + 1 |
| Platfrom2\_led1\_onoff | True | th2 = th2 + 1 |
| Platfrom2\_led2\_onoff | True | tc2 = tc2 + 1 |
| Platfrom3\_led1\_onoff | True | th3 = th3 + 1 |
| Platfrom3\_led2\_onoff | True | tc3 = tc3 + 1 |

## Arduino Board

Die PWM Frequenz der Ausgänge des Arduino Boards kann über die serielle Schnittstelle des Arduino Boards festgelegt werden. Über ein Terminalprogramm wird der Rechner mit dem Arduino Board verbunden. Nach einem Reset des Arduino Boards können die Werte für t\_on und t\_off geschrieben werden. Das Board wird mit einer Baudrate von 9600/ 8N1 mit dem Rechner verbunden. Die Einheit des Wertes für t\_on und t\_off ist Millisekunden (ms).

# Anhang

## Lichtschranke-OP Schaltung



## Schaltplan LED Ansteuerung



## Quellcode PWM Arduino

#include <EEPROM.h>

byte port\_b, port\_d;

byte status\_b = 0x00;

byte status\_d = 0x00;

byte store\_eeprom = 0x00;

boolean bo\_led = true;

int int\_true = 25;

int int\_false = 25;

int int\_time\_on = 25;

int int\_time\_off = 25;

void setup()

{

pinMode(2, INPUT\_PULLUP); // sets the pin as input

pinMode(4, INPUT\_PULLUP);

pinMode(7, INPUT\_PULLUP);

pinMode(8, INPUT\_PULLUP);

pinMode(12, INPUT\_PULLUP);

pinMode(13, INPUT\_PULLUP);

pinMode(3, OUTPUT); // sets the pin as output

pinMode(5, OUTPUT);

pinMode(6, OUTPUT);

pinMode(9, OUTPUT);

pinMode(10, OUTPUT);

pinMode(11, OUTPUT);

Serial.begin(9600);

Serial.setTimeout(10000);

Serial.println("Triple Platform/ SW 1.0/ Elektronik Werkstatt Biologie/ J. Schmid");

Serial.println("Value for time\_on");

int\_time\_on = Serial.parseInt();

Serial.print("Value for time\_on is ");

Serial.println(int\_time\_on);

if (int\_time\_on > 0){

store\_eeprom = int\_time\_on & 0x00FF;

EEPROM.write(0,store\_eeprom);

int\_time\_on = int\_time\_on >> 8;

store\_eeprom = int\_time\_on & 0x00FF;

EEPROM.write(1,store\_eeprom);

Serial.println("Value saved in EEPROM");

}

else{

Serial.println("No value saved");

}

Serial.println("Value for time\_off");

int\_time\_off = Serial.parseInt();

Serial.print("Value for time\_off is ");

Serial.println(int\_time\_off);

if (int\_time\_off > 0){

store\_eeprom = 0x00;

store\_eeprom = int\_time\_off & 0x00FF;

EEPROM.write(2,store\_eeprom);

int\_time\_off = int\_time\_off >> 8;

store\_eeprom = int\_time\_off & 0x00FF;

EEPROM.write(3,store\_eeprom);

Serial.println("Value saved in EEPROM");

}

else{

Serial.println("No value saved");

}

store\_eeprom = 0x00;

int\_true = EEPROM.read(1);

int\_true = int\_true << 8;

store\_eeprom = EEPROM.read(0);

int\_true = int\_true | store\_eeprom;

store\_eeprom = 0x00;

int\_false = EEPROM.read(3);

int\_false = int\_false << 8;

store\_eeprom = EEPROM.read(2);

int\_false = int\_false | store\_eeprom;

Serial.print("System Value for time\_on is ");

Serial.println(int\_true);

Serial.print("System Value for time\_off is ");

Serial.println(int\_false);

Serial.println("Running");

Serial.end();

}

void loop()

{

port\_b = PINB;

port\_d = PIND;

if (bo\_led == true){

bo\_led = false;

delay(int\_true);

status\_d = status\_d & 0x97;

status\_b = status\_b & 0xF1;

}

else{

bo\_led = true;

delay(int\_false);

status\_d = status\_d | 0x68;

status\_b = status\_b | 0x0E;

}

if ((port\_d & 0x04) == false){

status\_d = status\_d & 0xF7;

}

else{

}

if ((port\_d & 0x10) == false){

status\_d = status\_d & 0xDF;

}

else{

}

if ((port\_d & 0x80) == false){

status\_d = status\_d & 0xBF;

}

else{

}

if ((port\_b & 0x01) == false){

status\_b = status\_b & 0xFD;

}

else{

}

if ((port\_b & 0x10) == false){

status\_b = status\_b & 0xFB;

}

else{

}

if ((port\_b & 0x20) == false){

status\_b = status\_b & 0xF7;

}

else{

}

PORTD = status\_d;

PORTB = status\_b;

}

# Änderungshistorie

|  |  |  |  |
| --- | --- | --- | --- |
| Version | Änderung | Name | Datum |
| 1.0 | Erste Fassung | J. Schmid | 01.06.2015 |
|  |  |  |  |
|  |  |  |  |