HS PF

Interdisziplinäre Projektarbeit

Prof. Dr. Alfred Schätter

SS 2018

Delta5 Drohnen Race-Timer

Thema:

Lap-Timer mittels 5.8GHz Videosignalmessung

Rotzler, Marius

Studiengang „Wirtschaftsingenieurwesen“

Matr. Nr. 311116

rotzlerm@hs-pforzheim.de

Gegenheimer, Nico

Studiengang „Wirtschaftsingenieurwesen“

Matr. Nr. 311706

gegenhen@hs-pforzheim.de

Abgabetermin: 06.04.2018

Inhaltsverzeichnis

1 Einleitung und Zielsetzung

Im Rahmen einer interdisziplinären Projektarbeit, welche im Studiengang des Wirtschaftsingenieurwesens vorgesehen ist, wurde dieses Projekt unter Betreuung von Herrn Triebenstein durchgeführt.

Die Grundlage hierfür bildet ein Internetprojekt von dem *GitHub*-User Scott Chin, welches als offenes Projekt zum Nachbau frei zugänglich ist. Das Projekt lebt von Anregungen, Verbesserungen und Ideen von weiteren Usern, die sich über das soziale Netzwerk *Facebook* organisieren und austauschen.

Das Ziel dieser interdisziplinären Projektarbeit war es ein Race-Timer nach der Vorlage von Scott Chin zu entwickeln, in Betrieb zu nehmen und zu dokumentieren.

In aufeinander aufbauenden Projektschritten entstand sukzessive eine Race-Timer-Box, welche in der Lage ist, bei einem Drohnen- oder Hovercraftrennen die exakte Rundenzeit und Rundenzahl zu ermitteln. Die erfassten Daten werden automatisch in einer Datenbank gespeichert und auf einer Webseite visualisiert dargestellt, sodass jeder Rennteilnehmer das Ergebnis einsehen kann. Der Rennadministrator ist in der Lage in einem geschützten Bereich Einstellungen der RaceTimer-Box vorzunehmen.

Die Race-Timer-Box misst mittels Videosignalempfängern die Signalstärke der von den Drohnen gesendeten Videosignale. Da bei einem Vorbeiflug an der Race-Timer-Box die Signale am stärksten sind, lässt sich hierdurch auf eine abgeschlossene Rennrunde schließen. Durch einen Validierungsalgorithmus wird die Runde auf Richtigkeit geprüft und anschließend in die Datenbank gespeichert.

Diese Dokumentation ist eine Anleitung zur Reproduktion des hier umgesetzten Projekts und stellt somit keinen Anspruch an eine vollständige Projektdokumentation. Kenntnisse in Programmierung, Konstruktion und Elektronik sollten für eine Nachbildung vorhanden sein.

Das Projekt umfasst folgende Themenfelder:

- Software
  - Programmierung
    - C++
    - Python
    - Linux Shell
    - HTML
- Hardware
  - Elektronik
    - Bestimmung nötiger Bauteile z.B. Spannungswandler
    - Bestückung einer Platine
    - Löten
  - Gehäuse
    - Konstruktion mit SolidWorks
    - Umsetzung mit Creality CR10 3D-Drucker und Speedy 300 Lasermaschine
    - Montage

Für die Umsetzung werden sämtliche Dateien und Ordner, welche auf GitHub bereitgestellt sind, benötigt. Diese sind vor Beginn herunterzuladen, zu sichten und die ReadMe-Dokumente zu lesen auf: https://github.com/scottgchin/delta5_race_timer (Abgerufen am: 01.12.2017)

2 Hardware

2.1 Elektrische Bauteile

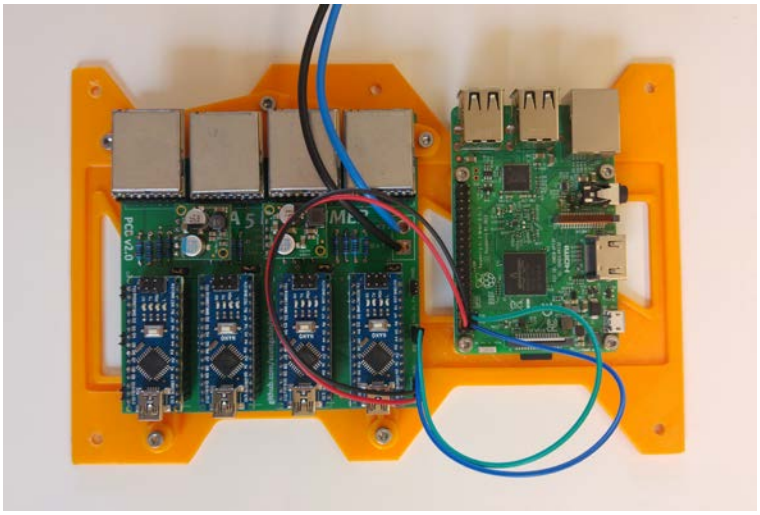

Abbildung 1 Übersicht Hardware

Zur Umsetzung des Projektes wird folgenden Hardware benötigt:

Anzahl	Bezeichnung
1	Raspberry Pi 3
1	8GB Micro SD Card
4	Arduino Nano
4	rx5808 mit SPI mod
12	1k Widerstand
4	100k Widerstand
1	Platine, wie https://www.seeedstudio.com/Delta-5-Race-Timer-v2-0-g-1048578
1	Spannungsquelle (min. 3A) ggf. als Akku ausgeführt
1	3,3V Spannungsregler, wie: https://eckstein-shop.de/Pololu-33V-25A-Step-Down-Spannungsregler-D24V25F3
1	5V Spannungsregler, wie: https://eckstein-shop.de/Pololu-5V-5A-Step-Down-Spannungsregler-D24V50F5
2	40mmx40mm 12V Lüfter
1	Drucktaster (Schließer)
4	Jumper für Steckerleisten
1	Gehäuse
div.	Leitungen, Schauben und Steckerleisten

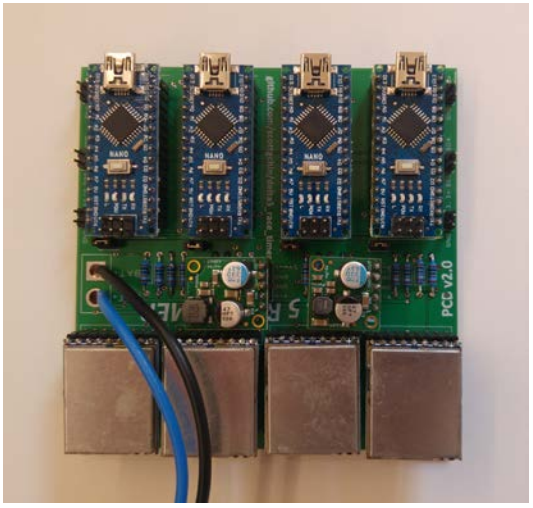

Abbildung 2 bestückte Delta5 Platine

Die Platine muss mit den entsprechenden Bauteilen, wie im Internetprojekt beschrieben, bestückt und verlötet werden. Alle Widerstände werden direkt auf der Platine verlöten wohingegen die restlichen Bauteile über Steckerleisten aufgesteckt werden. Somit werden alle weiblichen Steckerleisten auf der Platine festgelötet und alle männlichen an die jeweiligen Bauteile. Auf die Steckplätze, welche für Kabelverbindungen oder Jumper vorgesehen sind, werden männliche Steckerleisten aufgelötet.

Beim Löten ist die korrekte Vorgehensweise und alle Sicherheitsbestimmungen zum Schutze der Gesundheit und der Bauteile zu beachten (ggf. Schulung notwendig).

2.2 Gehäuse

Abbildung 3 Draufsicht Gehäuse

Im Abgabeordner sind sämtliche Solidworks CAD Daten, sowie die daraus generierten STL Daten zu finden. Aus diesen Daten kann ein G-Code für den 3D-Druck erstellt werden.

Das Gehäuse besteht aus sieben einzelnen Bauteilen und kann mit den CAD Daten reproduziert werden. Der Deckel und Boden lässt sich auf einer Laserschneidemaschine ausschneiden, alle anderen Bauteile können auf einem 3D-Drucker ausgedruckt werden.

3 Software

3.1 Delta5

3.1.1 Installation der Arduinos

Um die Arduino Nano´s mittels virtueller serieller Schnittstelle von einem PC oder MAC bespielen zu können wird die aktuellste Software von „Arduino IDE“ benötigt. Zu beziehen unter: https://www.arduino.cc/en/main/software
(Abgerufen am: 01.12.2017)

Ebenfalls wird ein USB zu Mini USB Kabel benötigt.

1. Arduino IDE starten.
2. Die Datei „delta5node.ino“ aus dem Projektordner öffnen.
3. Den PC/MAC mit einem Arduino Nano per Kabel verbinden.
4. In der Menüleiste werden unter dem Punkt „Werkzeuge“ folgende Einstellungen vorgenommen:
   Board: „Arduino Nano“
   Processor: „ATmega328P“
   Port: der verwendete serielle Port (herauszufinden über den Gerätemanager bei einem PC)
5. Nun muss für jeden Arduino Nano eine die I2C-Busadresse angepasst werden, da jeder Arduino Nano eine separate Busadresse benötigt:

   Node 1 = 8
   Node 2 = 10
   Node 3 = 12
   Node 4 = 14
   Node 5 = 16
   Node 6 = 18
   Node 7 = 20
   Node 8 = 22

   Hierzu wird der gewünschte Wert (rote Zahlen) in die Variable „i2cSlaveAddress“ eingetragen (Abbildung 4).

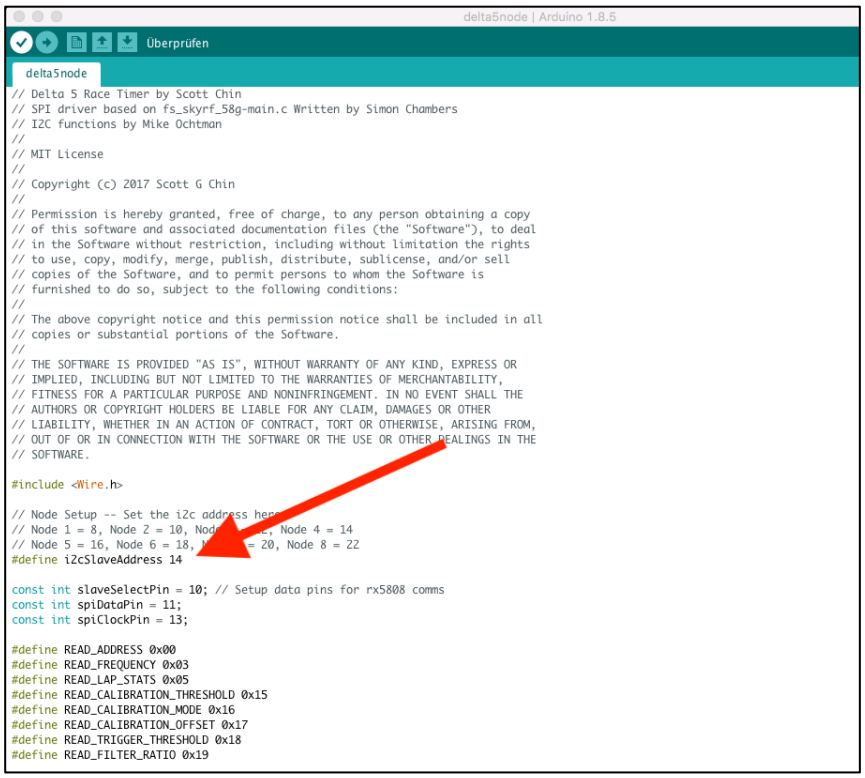

Abbildung 4: Screenshot Arduino IDE

6. Den Butten „Hochladen“ klicken und warten bis die Übertragung an den angeschlossenen Arduino Nano abgeschlossen ist. Die LEDs des Arduino Nano zeigen an, ob Daten empfangen werden.
7. Schritt 3 bis 6 mit jedem Arduino Nano wiederholten.

3.1.2 Installation des Raspberry Pi

1. SD-Karte auf das Format FAT32 formatieren
2. Downloaden des Raspbian-Installers „Noobs“ als ZIP-Datei von der offiziellen Raspberry Pi-Seite
3. Entpacken der ZIP-Datei und den Inhalt des Ordners auf die formatierte SD-Karte ziehen
4. Die SD-Karte sicher entfernen und in den Raspberry Pi einstecken
5. Jetzt wo sich das Operation-System auf der SD-Karte befindet, kann der Strom angeschlossen werden: Ab hier ist es hilfreich, Maus, Tastatur und einen Bildschirm am Raspberry Pi angeschlossen zu haben

Falls NOOBS genutzt wird, muss zu Beginn der Installation „Raspbian“ ausgewählt werden. Nun sollten den Installationsanweisungen gefolgt werden.

Eine detailliertere Beschreibung kann hier gefunden werden:
https://www.raspberrypi.org/learning/software-guide/quickstart/
(Abgerufen am: 01.12.2017)

3.1.3 Vorbereitung der Software

Quelle: https://github.com/scottgchin/delta5_race_timer
(Abgerufen am: 01.12.2017)

Zur Eingabe der Linux Shell Befehle ist auf der Benutzeroberfläche des Raspberry Pi der Terminal zu öffnen.

Um die SD-Karte upzudaten folgenden Befehl eingeben:

```
Sudo apt-get update && sudo apt-get upgrade
```

Mit Y und Enter bestätigen und warten, bis das Upgrade abgeschlossen ist. Hierfür muss eine Internetverbindung bestehen.

Mit folgendem Befehl die Konfigurations-Einstellungen des PI's öffnen und auf ‚Advanced Options' I2C erlauben:

```
Sudo raspi-config
```

Python und Python Treiber für die GPIO installieren:

```
Sudo apt-get install python-dev

Sudo apt-get install python-rpi.gpio
```

Delta5-Race-Timer-Software von Github in das ‚/home/pi/'-Verzeichnis des Raspberry Pi klonen:

```
Git clone https://github.com/scottgchin/delta5_race_timer
```

Installieren der Web-Server-Pakete, dafür im Terminal ‚/home/pi/delta5_race_timer/src/delta5server' öffnen:

```
cd /home/pi/delta5_race_timer/src/delta5server

sudo pip install –r requirements.txt
```

Raspberry neustarten mit:

```
Sudo reboot now
```

3.1.4 Systemstart

Manuell:

Im Terminal ‚/home/pi/delta5_race_timer/src/delta5server' öffnen und Befehl ausführen:

```
Cd /home/pi/delta5_race_timer/src/delta5server

Python server.py
```

Beim Booten starten:

Service erstellen:

```
Sudo nano /lib/systemd/system/delta5.service
```

Folgenden Inhalt einfügen:

```
[Unit]
Description=Delta5 Server
After=multi-user.target

[Service]
WorkingDirectory=/home/pi/delta5_race_timer/src/delta5server
ExecStart=/usr/bin/python server.py

[Install]
WantedBy=multi-user.target
```

Mit Strg-X, Y, Enter speichern und beenden

Rechte vergeben:

```
Sudo chmod 644 /lib/systemd/system/delta5.service
```

Zum Starten beim Neustarten folgende Befehle vergeben:

```
Sudo systemctl daemon-reload
Sudo systemctl enable delta5.service
Sudo reboot
```

3.2 An und aus Button

Quelle: https://howchoo.com/g/mwnlytk3zmm/how-to-add-a-power-button-to-your-raspberry-pi
(Zuletzt aufgerufen: 29.03.2018)

Zuerst muss ein Skript erstellt werden, welches „liste-for-shutdown.py“ genannt wird:

```
sudo nano listen-for-shutdown.py
```

Folgendes Script einfügen. Hier wird PIN21 als Power-Button definiert:

```
#!/usr/bin/env python


import RPi.GPIO as GPIO
import subprocess


GPIO.setmode(GPIO.BCM)
GPIO.setup(21, GPIO.IN, pull_up_down=GPIO.PUD_UP)
GPIO.wait_for_edge(21, GPIO.FALLING)

subprocess.call(['shutdown', '-h', 'now'], shell=False)
```

Nun muss das erstellte Skript ausführbar gemacht werden:

```
sudo mv listen-for-shutdown.py /usr/local/bin/
sudo chmod +x /usr/local/bin/listen-for-shutdown.py
```

Nun wird ein weiteres Skript „listen-for-shutdown.sh“ erstellt, welches den Service startet/stoppt:

```
sudo nano liste-for-shutdown.sh
```

In dieses Skript wird folgendes eingefügt:

```
#! /bin/sh

### BEGIN INIT INFO
# Provides:          listen-for-shutdown.py
# Required-Start:    $remote_fs $syslog
# Required-Stop:     $remote_fs $syslog
# Default-Start:     2 3 4 5
# Default-Stop:      0 1 6
### END INIT INFO

# If you want a command to always run, put it here

# Carry out specific functions when asked to by the system
case "$1" in
  start)
    echo "Starting listen-for-shutdown.py"
    /usr/local/bin/listen-for-shutdown.py &
    ;;
  stop)
    echo "Stopping listen-for-shutdown.py"
    pkill -f /usr/local/bin/listen-for-shutdown.py
    ;;
  *)
    echo "Usage: /etc/init.d/listen-for-shutdown.sh {start|stop}"
    exit 1
    ;;
esac

exit 0
```

Die Datei wird in /etc/init.d hinzufügt. Anschließend werden Rechte vergeben, um die Datei ausführbar zu machen.

```
sudo mv listen-for-shutdown.sh /etc/init.d/
sudo chmod +x /etc/init.d/listen-for-shutdown.sh
```

Das Skript wird nun registriert, um beim Booten des Raspberrys gestartet zu werden.

```
sudo update-rc.d listen-for-shutdown.sh defaults
```

Zum Testen, kann das Skript manuell gestartet werden:

```
sudo /etc/init.d/listen-for-shutdown.sh start
```

3.3 Raspberry Pi als Access Point:

Quelle: https://www.raspberrypi.org/documentation/configuration/wireless/access-point.md
(Zuletzt aufgerufen: 29.03.2018)

3.3.1 Benötigte Software installieren

```
Sudo apt-get install dnsmasq hostapd
```

Während die Konfigurationsdateien noch nicht bereit sind, die neue Software vorerst deaktivieren:

```
Sudo systemctl stop dnsmasq
Sudo systemctl stop hostapd
```

3.3.2 Eine Static-Ip konfigurieren

Es wird ein Standalone-Netzwerk konfiguriert, um als Server zu agieren. Dafür benötigt der Raspberry Pi eine statische IP-Adresse.
Diese Dokumentation geht davon aus, dass die Standard 192.168.X.X-IP Adresse für das WiFi-Netzwerk genutzt wird. So wird dem Server die IP-Adresse 192.168.4.1 zugewiesen. Es wird ebenfalls angekommen, dass das wlan0 genutzt wird.

Um die statische IP-Adresse zu konfigurieren, muss die dhcpcd-Konfigurations-Datei verändert werden:

```
Sudo nano /etc/dhcpcd.conf
```

Am Ende der Datei, muss diese so bearbeitet werden, dass es wie folgt aussieht:

```
interface wlan0
  static ip_address=192.168.4.1/24
```

Nun folgt ein Neustart des dhcpcd daemon und Setup der neuen wlan0-Konfiguration:

```
Sudo service dhcpcd restart
```

3.3.3 Konfiguration des DHCP-Server (dnsmasq)

Der DHCP-Service wird von dnsmasq bereitgestellt. Standardmäßig enthält die Konfigurationsdatei viele Informationen, die nicht benötigt werden. Daher ist es einfacher bei null anzufangen. Hierfür muss die Konfigurationsdatei umbenannt und eine neue erstellt werden.

```
Sudo mv /etc/dnsmasq.conf /etc/dnsmasq.conf.orig
Sudo nano /etc/dnsmasq.conf
```

Folgenden Inhalt in die dnsmasq Konfigurationsdatei hinzufügen und speichern:

```
interface=wlan0 # Use the require wireless interface - usually wlan0
dhcp-range=192.168.4.2,192.168.4.20,255.255.255.0,24h
```

Für wlan0, werden IP-Adressen zwischen 192.168.4.2 und 192.168.168.4.20, mit einer Nutzungsdauer von 24 Stunden zur Verfügung gestellt. Wenn DHCP-Dienste für andere Netzwerkgeräte bereitgestellt werden (z.B. eth0), können weitere Abschnitte mit dem entsprechenden Interface-Header hinzugefügt werden.

Es gibt noch viele weitere Optionen für dnsmasq; weitere Details finden Sie in der dnsmasq-Dokumentation (http://www.thekelleys.org.uk/dnsmasq/doc.html)

3.3.4 Konfigurieren der Access-Point Host Software (hostapd)

Die hostapd-Konfigurationsdatei, die sich unter /etc/hostapd/hostapd.conf befindet, muss bearbeitet werden, um die verschiedenen Parameter für das drahtlose Netzwerk hinzuzufügen. Nach der Erstinstallation handelt es sich um eine neue leere Datei.

```
Sudo nano /etc/hostapd/hostapd.conf
```

Die folgenden Informationen zur Konfigurationsdatei müssen anschließend hinzugefügt werden. Diese Konfiguration setzt voraus, dass Kanal 7 mit dem Netzwerknamen „Delta5Network“ und dem Passwort „Dronerace“ verwendet werden kann. Es muss beachtet werden, dass der Name und das Passwort keine Anführungszeichen enthalten sollten. Die Passphrase sollte zwischen 8 und 64 Zeichen lang sein.

```
interface=wlan0
driver=nl80211
ssid=Delta5Network
hw_mode=g
channel=10
wmm_enabled=0
macaddr_acl=0
auth_algs=1
ignore_broadcast_ssid=0
wpa=2
wpa_passphrase=DroneRacer
wpa_key_mgmt=WPA-PSK
wpa_pairwise=TKIP
rsn_pairwise=CCMP
```

Dem System muss nun mitgeteilt werden, wo die Konfigurationsdatei zu finden ist:

```
Sudo nano /etc/default/hostapd
```

Nun muss nach der Zeile **#DAEMON_CONF** gesucht werden, um diese mit folgendem zu ersetzen:

```
DAEMON_CONF="/etc/hostapd/hostapd.conf"
```

Starten der Services:

```
Sudo systemctl start hostapd
Sudo systemctl start dnsmasq
```

Routing und Masquerade hinzufügen:

```
Sudo nano /etc/sysctl.conf
```

Folgende Line mit einem „#“ auskommentieren:

```
#Net.ipv4.ip_forward=1
```

Wird zu:

```
Net.ipv4.ip_forward=1
```

Masquerade hinzufügen für outbound traffic auf eth0:

```
Sudo iptables –t nat –A POSTROUTING –o eth0 –j MASQUERADE
```

Speichern der iptables-Regeln:

```
Sudo sh –c „iptables-save > /etc/iptables.ipv4.nat"
```

Editieren der /etc/rc.local:

```
Sudo nano /etc/rc.local
```

Und direkt über **exit 0** folgendes einfügen:

```
Iptables-restore < /etc/iptables.ipv4.nat
```

Den Raspberry Pi neustarteten und nun sollte das neue WLAN mit andere Geräten gefunden werden können. Um mit anderen Linux-Boxen auf den Raspberry Pi Access Point zuzugreifen, muss SSH erlaubt werden.

4 Delta 5 Race Timer User Guide

4.1 Verbindung zum Server

Zum Verbinden des Raspberry Pis wurden folgende Zugangsdaten festgelegt:

Benutzername: pi
Passwort: raspberrypi

Zum Verbinden des drahtlosen Netzwerkes wurden folgende Zugangsdaten festgelegt:

Netzwerkname: Delta5Network
Passwort: DroneRacer

Den Internetbrowser öffnen und die IP-Adresse des Zeitmesssystems im Netzwerk über den Port 5000 oder wie in "server.py" konfiguriert eingeben.

192.168.4.1:5000/

Die für den Rennleiter reservierten Seiten sind passwortgeschützt mit

Standardbenutzer: admin

Passwort: delta5

4.2 Systemeinstellungen und -konfiguration (Seite „Einstellungen“)

Zu Beginn eines jeden Rennens sollte mit dem Zurücksetzen der Datenbank begonnen werden. Hierfür steht ein kompletter Reset „Reset Database/Database zurücksetzten“ sowohl „Reset Keep Pilots/Zurücksetzten und Piloten behalten“ zur Auswahl. Bei diesem wird die Datenbank zurückgesetzt, die vorherig eingestellte Piloten allerdings beibehalten.

Frequenzen werden unter der Überschrift „Heats“ konfiguriert. Die Standardeinstellung ist für bis zu 6 Knoten verfügbar.

Die Dropdown-Liste kann verwendet werden, um die Frequenz nach Bedarf zu ändern.

Mit einem Klick auf „Add Pilot/Pilot hinzufügen“ kann jeder Pilot eingetragen werden. Hier können Rufzeichen und Namen der Piloten aktualisiert werden.

Mit einem Klick auf „Add Heat/Heat hinzufügen“ kann jedem Piloten ein Platz zugewiesen werden. Mit Hilfe des Dropdown-Menüs, kann jedem Piloten ein Heat zugewiesen werden.

Wenn beim Passieren des Tores verpasste oder mehrfache Runden bemerkt werden, können die Sensorabstimmungswerte anhand der Standardeinstellungen mit einer detaillierten Beschreibung angepasst werden.

Diese können im Delta5-Ordner auf dem Raspberry Pi unter /doc/Tuning Parameters.md gefunden werden.

4.3 Laufende Rennen („Rennen“-Seite)

Der Rennleiter wird die meiste Zeit auf dieser Seite verbringen.

Wählen Sie zunächst die Schaltfläche „Heat“, um festzulegen, welcher Lauf gestartet werden soll.

Klicken Sie auf „Start Race/Rennen starten“ für einen Count-Up-Timer, der bei null beginnt und kein definiertes Ende hat. Dies wird für Heads-Up-Rennen verwendet, zuerst um X-Runden zu beenden.
Der Rennleiter klickt auf die Schaltfläche „Stop Race/Rennen stoppen“, nachdem alle Piloten ihrer Runden beendet haben.

Alternativ können Sie auch auf die Schaltfläche „Start Race 2min/Starte 2min Rennen“ klicken, um einen Countdown-Timer von zwei Minuten zu erhalten. Jeder Pilot hat zwei Minuten Zeit, um so viele Runden wie möglich zu absolvieren. Nachdem der letzte Summer ertönt und alle Piloten ihre letzte Runde beendet haben, klicken Sie auf die Schaltfläche „Stop Race/Rennen stoppen“.

Für jeden Knoten in einer Reihe unter den Pilotrufzeichen werden die RSSI-Werte, Current RSSI / Trigger / Peak angezeigt. Dies gibt dem Rennleiter ein sofortiges Sensor-Feedback für eventuelle Anpassungen.

Während eines Rennens gibt es neben jeder Runde einen X-Knopf. Dadurch wird diese Runde verworfen und die Zeit in die nächste Runde verschoben, wenn es nicht die letzte Runde ist.
Generell ist es sinnvoll, das System so einzustellen, dass es mehr Runden aufnimmt, anstatt fehlende Runden zu fahren, und so werden die Extras gelöscht.
Am Ende des Rennens kann es passieren, dass Piloten am Starttor vorbeifliegen, wenn sie zur Landung ansetzen und auf diese Weise werden auch die Runden entfernt, die eventuell abgeholt werden.

Klicken Sie nach jedem Rennen auf „Save Laps/Runden speichern“, um die Ergebnisse eines guten Rennens in der Datenbank zu speichern, oder auf „Clear Laps/Zurücksetzten“, um einen Fehlstart zu vermeiden oder die aktuellen Runden zu verwerfen.

4.4 Gespeicherte Rennen („Runden“ Seite)

Dies ist eine öffentliche Seite, frühere Rennergebnisse werden auf dieser Seite nach Läufen und Runden sortiert angezeigt.

4.5 Piloten und Heats („Heats“ Seite)

Auch eine öffentliche Seite, zeigt eine Zusammenfassung der Piloten und ihrer Läufe mit Kanalzuordnung.

5 Fazit und Ausblick

Das Internetprojekt von Scott Chin bietet eine gute Grundlage zur Entwicklung eines Race-Timers, jedoch waren nicht alle Umfänge ausreichend ausformuliert. Die fehlenden Punkte sind in dieser Anleitung berücksichtigt. Der Race-Timer ist denkbar intuitiv zu bedienen und bietet umfangreiche Funktionen für Rennteilnehmer und -administrator. Die momentan verbauten vier Kanäle lassen sich mit geringem Aufwand auf acht Kanäle erweitern, sodass mehr Drohen am Rennen teilnehmen können.

Das Projekt umfasst mehrere Themenbereiche, sodass unsere Kenntnisse in diesen Bereichen weiter vertieft wurden. Speziell das Zusammenwirken aus Programmierung, Elektronik und Konstruktion stellte uns vor besondere Herausforderungen.

Da sich das Onlineprojekt kontinuierlich weiterentwickelt, stellt diese interdisziplinäre Projektarbeit den Stand vom 01.12.2017 dar. Das Gehäuse des Race-Timer ist für die Verwendung einer Ampelanlage, angebunden mittels eines 4-poligen Steckers, vorbereitet. Die Ampelanlage visualisiert in Kombination mit einem aktuelleren Softwarestand den Rennverlauf. Somit kann das Projekt mit einfachen Mitteln erweitert und weiter aufgewertet werden.

Darüber hinaus ist der Race-Timer flexibel für jegliche Rennen einsetzbar, welche ein Videosignal im Bereich von 5,8 GHz verwenden. Infolge dessen ist der Race-Timer nicht nur für Drohnenrennen geeignet, sondern ebenfalls für Rennen mit Modellautos, Modellflugzeuge oder Modellhovercrafts.