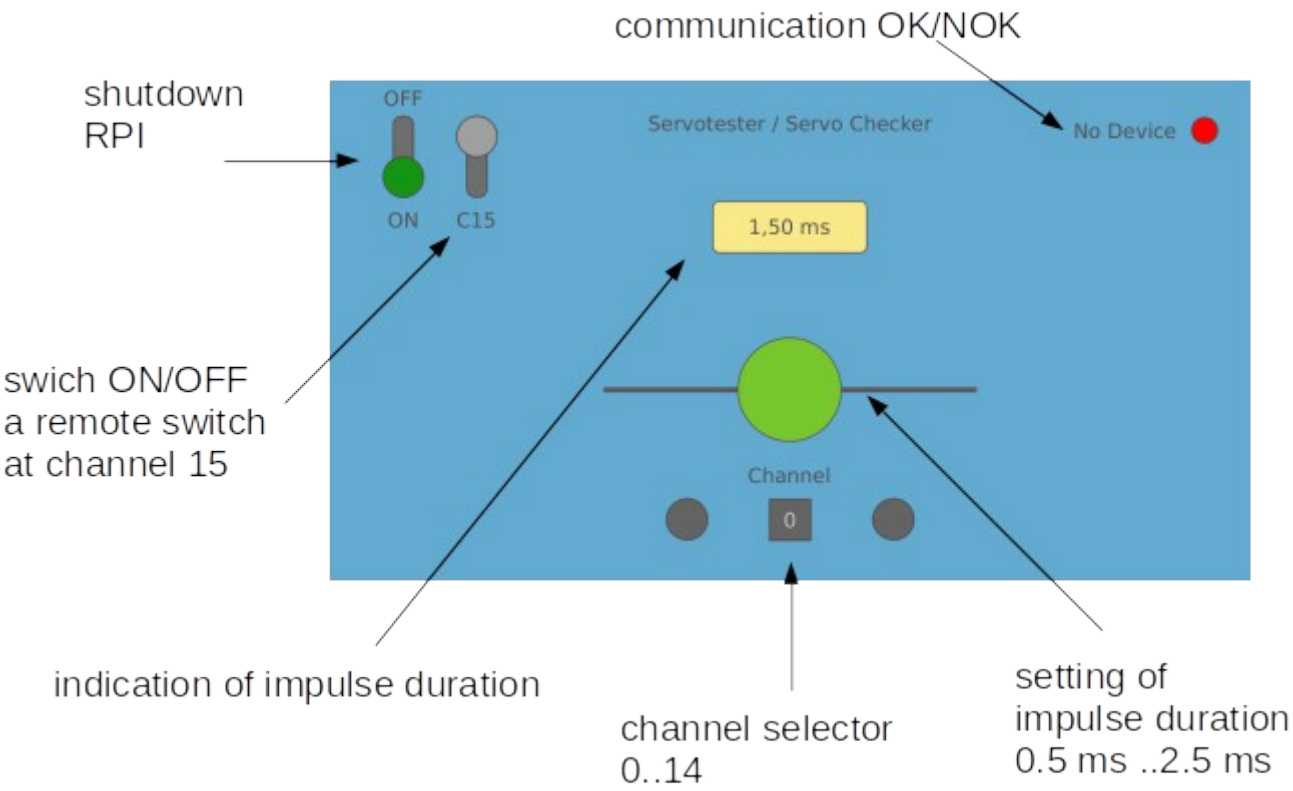
Pi Servo Tester - Pi Servo Checker

by Bernd Hinze - 2019



Der Pi Servo Tester erzeugt Servo-Impulse mit einer Dauer im Bereich von 0,5 ms bis 2,5 ms und einer Zyklus Rate von 50 ms. Im Modellbau wird er zur Funktionsprüfung von Servos benutzt. Der Servo Tester besteht aus einem **Raspberry Pi** mit einem 16 Kanlal PWM Modul und eine **Smartphone App**, die die Steuerung übernimmt. Die komplette Dokumentation dieser Konfiguration ist unter „https://github.com/monbera/TelDaControl“ zu finden. Die Raspberry Empfänger Software wird komplett übernommen und muss nur hinsichtlich der Konfiguration angepasst werden. Die Applikation des Smartphones wurde speziell für diese Anwendung entwickelt.

The Pi Servo Tester generates servo pulses with a duration of 0.5 ms to 2.5 ms and a cycle rate of 50 ms. In model making it is used for functional testing of servos.
The Servo Checker consists of a Raspberry Pi with a 16-channel PWM module and a smart phone App, which takes over the control.
The complete documentation of this configuration can be found at "https://github.com/monbera/TelDaControl". The Raspberry receiver software is taken over completely and only needs to be adapted with regard to the configuration. The smart phone application was developed especially for this application.
Translated with www.DeepL.com/Translator

Step 1

Folgende Materialien werden benötigt:

- Raspberry Pi Zero WH
- 16 Channel PWM Bonnet
- Stromversorgung 5V/1A
- Smartphone

The following materials are required:

- Raspberry Pi Zero WH
- 16 Channel PWM Bonnet
- Power supply 5V/1A
- Smart phone

Step 2

Installieren der ‚Servo Tester' APK Datei auf dem Smartphone. Die Installation von fremden Quellen in den Sicherheitseinstellungen zugelassen werden.

Install the 'ServoChecker' APK file on the smart phone. The installation of external sources must be allowed in the security settings.

Step 3

Kapitel 3.2.2.5 „Preparation of the Raspberry Pi" [1] beschreibt die notwendigen Schritte zur Inbetriebnahme des Raspberry Pi. Nur die Datei „rcmain.py" muss wie auf Seite 3 dargestellt angepasst werden. Nicht vergessen das Löten einer Brücke auf dem Servo+ PWM Bonnet Board entsprechend Bild 5 [1].
Zusammenstecken mit dem Raspberry Pi und Inbetriebnahme entsprechend Kapitel 3.2.2.5.3 ‚First Test' [1].

Chapter 3.2.2.5 "Preparation of the Raspberry Pi" [1] describes the necessary steps for commissioning the Raspberry Pi. Only the file "rcmain.py" has to be adapted as shown on page 3. Do not forget to solder a bridge on the Servo+ PWM Bonnet Board as shown in Figure 5 [1]. Plug together with the Raspberry Pi and commission according to the chapter 3.2.2.5.3 'First Tes't [1].

Step 4

Auf dem Handy sollte die Datenverbindung zum Provider ausschalten werden und die Hotspot Funktion des Smartphones eingeschaltet sein. Die Hotspot-Konfiguration muss mit den Daten der „wpa_supplicant.conf" auf dem Raspberry Pi

On the mobile phone the data connection to the provider should be switched off and the hot spot function of the smart phone should be switched on. The hot spot configuration must match the data of the "wpa_supplicant.conf" on the

übereinstimmen.
Nachdem die ‚ServoChecker' App gestartet wurde, leuchtet der Kommunikationsindikator rot. Nun kann der Servo Tester (Rpi) eingeschaltet werden. Nach dem Kommunikationsaufbau ändert sich der Status der Anzeige in ‚grün'. Nun könnt ihr Servos an den Kanälen 0..14 Servos testen.

Raspberry Pi.
After the 'ServoChecker' app has been started, the communication indicator lights up red. Now the Servo Tester (Rpi) can be switched on. After starting the communication the status of the indicator changes to 'green'. Now you can test servos on the channels 0..14 Servos.

Translated with www.DeepL.com/Translator

[1]	https://github.com/monbera/TelDaControl

rcmain.py

```
#! /usr/bin/env python
#-------------------------------------------------------------------------------
# Name:        Remote Control Receiver
# Purpose:     Servo Checker
# Author:      Bernd Hinze
#
# Created:     10.04.2019
# Copyright:   (c) Bernd Hinze 2019
# Licence:     MIT see https://opensource.org/licenses/MIT
# -------------------------------------------------------------------------------
import time
from rcapp import PWM_Controller, UDP_Client, Observer, SIM

def main():
    # Configuration general prototype
    # Channel 0..14: Servos
    # Channel 15: Digital Output
    if not SIM:
        time.sleep(10)
    L298Channels = []
    DIOs = [15]
    Inverted = []
    SC = PWM_Controller(0.5, 2.5, 50, L298Channels, DIOs, Inverted)
    SC.fail_safe()
    S = UDP_Client(SC,'', 6000, 6100, 10, "ServoChecker")
    O = Observer(SC, 30.0, "ServoChecker")

if __name__ == '__main__':
    main()
```