elektronische Adventskranzkerzen-Steuerung

Die Aufgabe

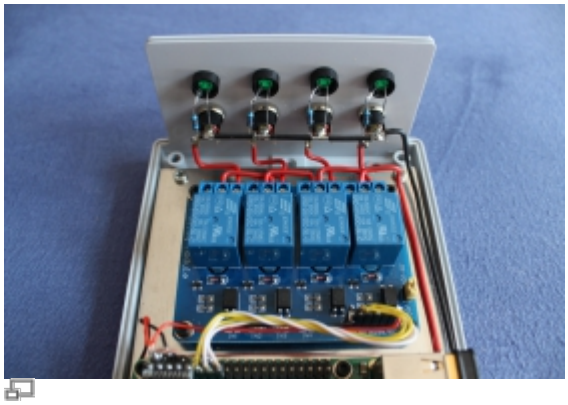
Nachrüstung von vier elektrischen Kerzen auf einem Adventskranz von ca. 3 Metern Durchmesser für den Rastplatz in Berne - Weserdeich.

Innenansicht, links R-Pi, rechts das Relais-Modul

Die Funktionen

Basisfunktionen

- Ansteuerung von vier elektrischen Kerzenlampen (bevorzugt 12V LEDs)
- Spannungsbereich 5 V, 12 V oder 24 V DC bzw. 230 V AC über Steckernetzteil
- Unabhänginger Betrieb –> Termingesteuerte Ein-/und Ausschaltung der vier Kerzen je nach Adventswoche
- Robustheit gegen Verlust der Versorgungsspannung (Netzausfall) bzgl. Uhrzeit und Daten (Betriebssystem und Software)



Innenansicht, Verdrahtung des Relais-Moduls

Zusatzfinktionen

- WLAN Access Point für Wartung und Servicezwecke
- Webserver für Darstellung einer Konfigurationsseite mir folgenden Funktonen:
  - Ein- und Ausschalten der einzelnen Kerzen (LEDs)
  - Reset der aktuellen Kerzensteuerung auf den aktuellen Tag
  - Anzeige der aktuellen Systemzeit des R-Pi
  - Setzten von Datum und Uhrzeit des R-Pi
  - Neustart des Systems

Beschriftung der LED-Ausgänge

Beschriftung der Spannungsversorgung

Die Technik

Komponente:	Quelle	Preis
Raspberry Pi B+	z.B. eBay	10 €

Komponente:	Quelle	Preis
USB-WLAN Adapter, EDIMAX, EW-7811UN, Chipsatz **RTL8188CUS**	diverse	~ 6,90 €
Netzteil 12V, 15 Watt	Reichelt	~ 8 €
Relaiskarte, Typ: 4-Kanal Channel Relais Relay 5V/230V	eBay	~ 7,50 €
Echtzeituhr als RTC-Modul, Typ: DS3231	eBay	~ 4,60 €
Gehäuse	Reichelt, Art.-Nr.: BOPLA KS 450	8,35 €
Befestigungsmaterial	diverse	~ 5 €
Kabel	diverse	2 €
Lampenfassung, 4x	ELV	? €
LEDs, 4x	ELV	? €
Hohlstecker-Buchsen, 4x für LEDs	Reichelt, Art.-Nr.: HEBL 25	2 €
Hohlstecker-Buchse, 1x Netzteil	Reichelt, Art.-Nr.: HEBL 21	0,30 €
Sicherungen inkl. Sicherungshalter, nx	Reichelt, Art.-Nr.: PL FPG1-40	1,19 €
Aluplatte 100 mm x 160 mm	http://www.AluFritze.de	4,92 €
DC/DC Wandler 9-36V/5V 4W für R-Pi	eBay	3,50 €


leuchtender Adventskranz

Die Software

- Raspbian GNU/Linux 8 Jessie Lite, Stand 23.09.2016
- eigene Shell Scripte
- eigene PHP Scripte

Die Installation des Raspbian Grundsystems ist in diversen Anleitungen im Netz oder hier beschrieben.

Raspbian

Da ich ein Raspbian Lite verwende, welches an sich schon auf das wesentliche reduziert ist, muss z.B. nur der ungenutzte Bluetooth-Kram entfernt werden. Ein bisschen notwendige Software kann auch gleich nachinstalliert werden.

```
sudo su
apt-get update
```

```
apt-get upgrade
apt-get install mc aptitude i2c-tools
apt-get purge pi-bluetooth
aptitude purge pi-bluetooth
```

Die Echtzeituhr

Echtzeituhr-Bausteins (RTC) DS3231SN am Raspberry Pi B+ in Betrieb nehmen:

```
sudo su
raspi-config
  --> Advanced Options
    --> I2C aktivieren
  --> Finish
# reboot
shutdown -r now && exit
```

Nach dem Neustart wird das I2C Kernelmodul automatisch geladen und steht zur Kommunikation mit der Echtzeituhr zur Verfügung. Damit die benötigten I2C Treiber (Module) beim Systemstart automatisch geladen werden, müssen sie in die Konfigurationsdatei `/etc/modules` eingetragen werden. Zusätzlich wird noch der Zugriff auf eine (nun vorhandene) Hardware Echtzeituhr ermöglicht (HWCLOCKACCESS=yes) und ein Gerätename (rtc0) für den späteren Zugriff auf die Uhr festgelegt.

```
sudo su
echo "i2c-dev" >> /etc/modules
echo "rtc-ds1307" >> /etc/modules
echo "i2c-bcm2708" >> /etc/modules
echo "HWCLOCKACCESS=yes" >> /etc/default/hwclock
echo "HCTOSYS_DEVICE=rtc0" >> /etc/default/hwclock
```

Im Anschluss zeigt ein erster Test, ob die Echtzeituhr am I2C-Bus des Prozessors gefunden wird und ansprechbar ist.

```
root@raspberrypi2:~# i2cdetect -y 1
     0  1  2  3  4  5  6  7  8  9  a  b  c  d  e  f
00:          -- -- -- -- -- -- -- -- -- -- -- -- --
10: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
20: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
30: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
40: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
50: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
60: -- -- -- -- -- -- -- -- 68 -- -- -- -- -- -- --
70: -- -- -- -- -- -- -- --
root@raspberrypi2:~#
```

Aah!, sehr schön. Die neue Echtzeituhr wird als I2C-Gerät mit der Adresse 0x68 erkannt.
Jetzt muss noch der Kernel über das neue Gerät (ds3231) und die Adresse (0x68) informiert werden unter der es erreichbar ist. Dann kann die Echtzeituhr auch per Kommando `hwclock` ausgelesen und eingestellt werden.

```
root@raspberrypi2:~# echo ds3231 0x68 | sudo tee /sys/class/i2c-
adapter/i2c-1/new_device
root@raspberrypi2:~# hwclock
Sa 29 Jan 2000 12:21:02 CET  -0.526365 seconds
```

Super!, die (noch unprogrammierte) Hardware-Uhr ist über die soeben ermittelte I2C Adresse per `hwclock`ansprechbar und liefert eine Zeit und ein Datum zurück.
Kleiner Test, ob die (System-) Uhrzeit des Raspberry Pi aktuell richtig gesetzt ist:

```
root@raspberrypi2:~# date
Mo 18. Jan 21:05:33 CET 2016
```

Ja, ist sie, dank vorinstalliertem NTP Dämon `ntpd` 
Systemzeit in die Echtzeituhr schreiben (-w wie write) und anschließend wieder auslesen (-r wie read):

```
root@raspberrypi2:~# hwclock -w
root@raspberrypi2:~# hwclock -r
Sa 29 Jan 2000 12:26:07 CET  -0.423200 seconds
```

Damit bei jedem Start des Raspberry Pi die Uhrzeit von der Echtzeituhr gelesen und als Systemzeit gesetzt wird, müssen folgende Zeilen in der `/etc/rc.local` oberhalb des `exit 0` hinzugefügt werden:

```
#!/bin/bash
sleep 5
/bin/echo ds3231 0x68 > /sys/class/i2c-adapter/i2c-1/new_device
sleep 2
/sbin/hwclock -s
sleep 2
/var/www/html/kerzeneinschalten.sh
exit 0
```

Mit der letzten Zeile `./var/www/html/kerzeneinschalten.sh` wird beim Systemstart einmalig ein Script zum initialen Einschalten der Adventskranzkerzen ausgeführt.

WLAN Access Point

Da das System im späteren Einsatz keinen (kabelgeführten) Internetanschluss besitzen wird, ein Zugriff z.B. per SSH aber trotzdem möglich sein soll (am besten natürlich drahtlos), wird ein WLAN Access Point eingerichtet. Hier mit einem USB-WLAN Adapter Typ EW-7811UN von EDIMAX.
Die notwendigen Softwareteile sind dann:

- `hostapd` - die Access Point Funktionalität an sich
- `dnsmasq` - DHCP Server damit der Client automatisch eine IP Adresse bekommt
- `wireless-tools` und `iw` zur Konfiguration des WLAN
- `firmware-realtek` - die notwendige Firmware für den USB-WLAN Adapter

```
sudo apt-get install hostapd dnsmasq firmware-realtek wireless-tools iw
```

Konfiguration der LAN- und der WLAN-Schnittstelle mit einer statischen IP Adresse und einer Subnetzmaske.

```
sudo su
vi /etc/network/interfaces
allow-hotplug eth0
iface wlan0 inet static
address 192.168.100.181
netmask 255.255.255.0

allow-hotplug wlan0
iface wlan0 inet static
address 192.168.0.1
netmask 255.255.255.0
```

Damit die Access Point Funktion auch beim Booten automatisch gestartet wird, muss sie unter `/etc/default/hostapd` aktiviert und der Speicherort der Konfigurationsdatei angegeben werden:

```
sudo su
vi /etc/default/hostapd
...
DAEMON_CONF="/etc/hostapd/hostapd.conf"
RUN_DAEMON="yes"
...
```

Über die Konfigurationsdatei `/etc/hostapd/hostapd.conf` werden dann die wesentlichen Funktionen wie z.B. der Access Point Name (SSID), die Verschlüsselung und natürlich das Password selber eingestellt. Bei WPA2 muss das Passwort mindestens 8 Zeichen lang sein, sonst verweigert `hostapd` den Dienst.

```
sudo su
cat <<EOT > /etc/hostapd/hostapd.conf
# Schnittstelle
interface=wlan0

# Treiber
driver=nl80211
# WLAN-Konfiguration
ssid=<SSID-DES-ACCESSPOINTS>
channel=5
hw_mode=g

# ESSID sichtbar
ignore_broadcast_ssid=0

# Verschlüsselung (hier: WPA2)
wpa=2
wpa_key_mgmt=WPA-PSK
```

```
rsn_pairwise=CCMP
wpa_passphrase=<PASSWORT>
auth_algs=1

# Schlüsselintervalle
wpa_group_rekey=600
wpa_ptk_rekey=600
wpa_gmk_rekey=86400

# Ländereinstellungen
country_code=DE
ieee80211d=1

# MAC-Authentifizierung (0=aus)
macaddr_acl=0
EOT
```

Damit jeder Client automatisch eine IP Adresse bekommt, wird in der `/etc/dnsmasq.conf` der DHCP Adressbereich (hier von 192.168.0.2 bs.10) und eine Gültigkeitsdauer (12h) eingestellt. Der DHCP Server soll dabei nur über die WLAN Schnittstelle IP Adressen verteilen (interface=wlan0), nicht aber über die LAN Schnittstelle (no-dhcp-interface=eth0) - da lauscht der DHCP-Client für den Fall, das der R-Pi an das heimischen Netz angeschlossen wird.

```
sudo su
vi /etc/dnsmasq.conf
interface=wlan0
dhcp-range=interface:wlan0,192.168.0.2,192.168.0.10,255.255.255.0,12h
no-dhcp-interface=eth0
```

Mit der Folgenden Zeile wird das Powermanagement des USB-WLAN Adapters deaktiviert. Macht man das nicht kann es u.U. zu Verbindungsabbrüchen kommen.

```
sudo echo "options 8192cu rtw_power_mgnt=0 rtw_enusbss=0" >>
/etc/modprobe.d/8192cu.conf
```

Den DHCP-Client benötigen wir bei statischer Konfiguration der Netzwerkschnittstellen nicht, daher wird er deaktiviert:

```
systemctl daemon-reload
service dhcpcd stop
systemctl disable dhcpcd

nano /etc/systemd/system/rc-local.service

[Unit]
 Description=/etc/rc.local Compatibility
 ConditionPathExists=/etc/rc.local

[Service]
```

```
 Type=forking
 ExecStart=/etc/rc.local start
 TimeoutSec=0
 StandardOutput=tty
 RemainAfterExit=yes
 SysVStartPriority=99

[Install]
 WantedBy=multi-user.target

systemctl enable rc-local
systemctl start rc-local.service
systemctl status rc-local.service
```

WiringPi

Zur Ansteuerung der GPI/O-Pins gibt es ein komfortables Tool namens `wiringPi`, zu finden unter www.wiringpi.com.

```
sudo su
apt-get install wiringpi
```

Web-GUI und Scripte

Zum testen der LEDs des Adventskranzes sollen diese einzelnen per Web-Interface schaltbar sein. Zu diesem Zweck wird ein Webserver benötigt. Der Einfachheit halber habe ich mich für den allseits bekannten Apache2 entschieden. Dieser bringt auch gleich ein PHP-Modul mit welches für die Ausführung diverser anderer Funktionen benötigt wird.

```
sudo apt-get install apache2 php5
```

Damit für das Schalten der LEDs auch „Schaltelemente“ in Form sog. „Slider“ oder auch „Schiebeschalter“ zur Verfügung stehen, bediene ich mich der Java Script Funktionen aus dem jQuery Framework.

```
sudo su
cd /var/www/html/
wget http://code.jquery.com/jquery-1.12.4.min.js
wget http://code.jquery.com/mobile/1.4.5/jquery.mobile-1.4.5.js
wget http://dev.jtsage.com/cdn/datebox/1.4.5/jqm-datebox-1.4.5.all.js
wget http://code.jquery.com/mobile/1.4.5/jquery.mobile-1.4.5.css
```

Shell Script zum Schalten der Kerzen

Mit Hilfe dieses Shell Scriptes wird in Abhängigkeit des aktuellen Datums ermittelt, welche Adventskerzen eingeschaltet werden müssen. Das Script ist dazu in folgende Teile bzw. Funktionen

unterteilt:

- function debug()

Hilfsfunktion zur Ausgabe von Debug Meldungen, kann über den Parameter debug beim Start aktiviert werden. Fehlt der Parameter so werden keine Textmeldungen während der Scriptlaufzeit ausgegeben.

- function random_candles()

Damit nicht jeden Tag im Advent die selbe Kerze (LED) leuchtet wird in dieser Funktion eine zufällige Reihenfolge festgelegt. Es werden immer alle vier Kerzen zufällig verteilt und dabei nicht unterschieden, welche Adventswoche gerade ist.

- function switch_gpio ()

Hier wird der jeweilige GPI/O Pin des Prozessor respektive das daran angeschlossene Relais ein oder aus geschaltet.

- function preset_candles ()

Um einen definierten Startzustand zu erhalten werden alle GPI/Os (Relais) zu Beginn in den AUS-Zustand geschaltet.

- function get_advent_days ()

Die Berechnung der Adventssonntage erfolg in Abhängigkeit des jeweiligen Kalenderjahres jeweils beim Start des Scriptes. Berechnete Daten werden nicht abgespeichert.

- function print_advent_days ()

Hilfsfunktion zur textuellen Ausgabe der berechneten Adventssonntage.

- function switch_candles ()

In Abhängigkeit der berechneten Adventssonntage wird in dieser Funktion die jeweilige Anzahl an Kerzen geschaltet.

kerzeneinschalten.sh

```
#!/bin/bash
#
#
## Debugging parameters:
#TODAY="$1"
##TODAY="20161204"
#CURRENT_YEAR="2017"

# normal parameters:
DATEFORMAT="%Y%m%d"
TODAY=`date +$DATEFORMAT`
```

```
CURRENT_YEAR=`date +%Y`
CHRISTMAS="${CURRENT_YEAR}1224"
TREE_KINGS="${CURRENT_YEAR}0106"

# Zuordnung Adventskerze <-> GPIO Pin:
CANDLE_ADDRESS[0]="17"
CANDLE_ADDRESS[1]="18"
CANDLE_ADDRESS[2]="27"
CANDLE_ADDRESS[3]="22"

# Zufallszahl zuruecksetzten
rand=""

# Verwendete Kerzen zuruesetzten
CANDLE_TO_USE[0]=""
CANDLE_TO_USE[1]=""
CANDLE_TO_USE[2]=""
CANDLE_TO_USE[3]=""

# Debugging ja/nein
DEBUG=""

function debug()
{
MESSAGE=$1
echo "$MESSAGE"
}


function random_candles()
{
if [ -n "$DEBUG" ]; then debug "Entering randomization of candle
usage:"; fi
COUNTER=0
while [ $COUNTER -lt 4 ]; do
  rand=`echo $(($RANDOM % 4))`
       if [ "${CANDLE_TO_USE[$rand]}" = ""  ]; then
         CANDLE_TO_USE[$rand]=$COUNTER
         if [ -n "$DEBUG" ]; then debug "$COUNTER. Candle to use:
${CANDLE_ADDRESS[$rand]} "; fi
         COUNTER=$((COUNTER + 1))
       fi
done
}

function switch_gpio ()
{
i=$1
case $2 in
  on)
   state=0
```

```
  ;;
  off)
   state=1
  ;;
  *)
esac

if [ -n "$DEBUG" ]; then debug "Switch candle ${CANDLE_ADDRESS[$i]}
$state"; fi
   gpio -g mode ${CANDLE_ADDRESS[$i]} out
   sleep .1
   gpio -g write ${CANDLE_ADDRESS[$i]} $state
   sleep .1
}

function preset_candles ()
{
if [ -n "$DEBUG" ]; then debug "Initial state: all candles off!"; fi
for i in $(seq 0 3);
do
  switch_gpio $i off
done
}

function get_advent_days ()
{
X=`date -d "$CURRENT_YEAR-12-24 - 3 weeks" +%w`
ADVENT[0]=`date -d "$CURRENT_YEAR-12-24 - 3 weeks -$X day"
+$DATEFORMAT`

X=`date -d "$CURRENT_YEAR-12-24 - 2 weeks" +%w`
ADVENT[1]=`date -d "$CURRENT_YEAR-12-24 - 2 weeks -$X day"
+$DATEFORMAT`

X=`date -d "$CURRENT_YEAR-12-24 - 1 weeks" +%w`
ADVENT[2]=`date -d "$CURRENT_YEAR-12-24 - 1 weeks -$X day"
+$DATEFORMAT`

X=`date -d "$CURRENT_YEAR-12-24 - 0 weeks" +%w`
ADVENT[3]=`date -d "$CURRENT_YEAR-12-24 - 0 weeks -$X day"
+$DATEFORMAT`
}

function print_advent_days ()
{
if [ -n "$DEBUG" ]; then debug "1. Advent in $CURRENT_YEAR ist der:
${ADVENT[0]}"; fi
if [ -n "$DEBUG" ]; then debug "2. Advent in $CURRENT_YEAR ist der:
${ADVENT[1]}"; fi
```

```
if [ -n "$DEBUG" ]; then debug "3. Advent in $CURRENT_YEAR ist der:
${ADVENT[2]}"; fi
if [ -n "$DEBUG" ]; then debug "4. Advent in $CURRENT_YEAR ist der:
${ADVENT[3]}"; fi
}

function switch_candles ()
{

if [ $TODAY -le $TREE_KINGS ] ; then
  if [ -n "$DEBUG" ]; then debug "Weihnachten ist vorbei aber die
Heiligen drei Koenige sind auch noch nicht..."; fi
  if [ -n "$DEBUG" ]; then debug "Trotzdem alle Kerzen an! ;-)"; fi
    for i in $(seq 0 3);
    do
      switch_gpio ${CANDLE_TO_USE[$i]} on
    done

elif [ $TODAY -lt ${ADVENT[0]} ] ; then
  if [ -n "$DEBUG" ]; then debug "noch keine Adventszeit :-("; fi
elif [ $TODAY -ge ${ADVENT[0]}  ] && [ $TODAY -lt ${ADVENT[1]} ] ; then
  if [ -n "$DEBUG" ]; then debug "erste Adventswoche :-)"; fi
  for i in $(seq 0 0);
    do
      switch_gpio ${CANDLE_TO_USE[$i]} on
  done
elif [ $TODAY -ge ${ADVENT[1]} ] && [ $TODAY -lt ${ADVENT[2]} ] ; then
  if [ -n "$DEBUG" ]; then debug "zweite Adventswoche :-)"; fi
  for i in $(seq 0 1);
    do
      switch_gpio ${CANDLE_TO_USE[$i]} on
    done
elif [ $TODAY -ge ${ADVENT[2]} ] && [ $TODAY -lt ${ADVENT[3]} ] ; then
  if [ -n "$DEBUG" ]; then debug "dritte Adventswoche :-)"; fi
  for i in $(seq 0 2);
    do
      switch_gpio ${CANDLE_TO_USE[$i]} on
    done
elif [ $TODAY -ge ${ADVENT[3]} ] && [ $TODAY -le $CHRISTMAS ] ; then
  if [ -n "$DEBUG" ]; then debug "vierte Adventswoche :-)"; fi
  for i in $(seq 0 3);
    do
      switch_gpio ${CANDLE_TO_USE[$i]} on
    done
elif [ $TODAY -gt $CHRISTMAS ] ; then
  if [ -n "$DEBUG" ]; then debug "Weihnachten ist vorbei :-("; fi
  if [ -n "$DEBUG" ]; then debug "Trotzdem alle Kerzen an! ;-)"; fi
    for i in $(seq 0 3);
    do
      switch_gpio ${CANDLE_TO_USE[$i]} on
    done
```

```
fi
}

###### Main
#
# Check, if debugging is enabled:
if [ "$1" = "debug" ]; then
  DEBUG="true"
fi

if [ -n "$DEBUG" ]; then debug "Today: $TODAY"; fi
if [ -n "$DEBUG" ]; then debug "Year: $CURRENT_YEAR"; fi
if [ -n "$DEBUG" ]; then debug "Three Kings day: $TREE_KINGS"; fi
if [ -n "$DEBUG" ]; then debug "Christmas: $CHRISTMAS"; fi

#if [ $TODAY -lt ${ADVENT[0]} ] ; then

  preset_candles
  get_advent_days
  print_advent_days
  random_candles
  switch_candles
#echo "Zufallszahl: $rand"
```

Damit das Shell Script vom Webserver gestartet werden kann muss es ausführbar gemacht werden.

```
sudo touch /var/www/html/kerzeneinschalten.sh && chmod +x
/var/www/html/kerzeneinschalten.sh
```

Aktuell wird das Script `kerzeneinschalten.sh` nur einmalig beim Systemstart ausgeführt. Läuft der Raspberry jedoch über den Tageswechsel oder sogar über den Wechsel Sa./So., muss also täglich um 00:00 Uhr geprüft werden, ob zusätzliche Lichter am Adventskranz einzuschalten sind. Folgender Eintrag in der Datei `/etc/crontab` erstellt einen sog. CRON-Job. Dieser startet stets um 00:00 Uhr das Script `kerzeneinschalten.sh` und schaltet somit die richte Anzahl LEDs ein.

```
sudo su
nano /etc/crontab
0 0   * * *   www-data  /var/www/html/kerzeneinschalten.sh
```

Die HTML-Seite

Der Speicherort für die Webseite sowie alle PHP-Dateien und das Kerzen-Schalt-Script /ist `var/www/html/`. Mit Hilfe der folgenden HTML-Seite können die vier Kerzen jeweils einzeln ein und wieder aus geschaltet werden. Auch das Setzen von Datum und Uhrzeit des R-Pi ist darüber möglich.

index.html

```
<!DOCTYPE html>
<html>
<head>
<meta name="viewport" content="width=device-width, initial-scale=1,
maximum-scale=1">
<meta http-equiv="content-type" content="text/html; charset=UTF-8" />
<script type='text/javascript' src='jquery-1.12.4.min.js'></script>
<script type='text/javascript' src='jquery.mobile-1.4.5.js'></script>
<script type='text/javascript' src='jqm-datebox-1.4.5.all.js'></script>
<link href="jquery.mobile-1.4.5.css" rel="stylesheet" />

<script type='text/javascript' src='GPIOson.js'></script>

<script>
  $(document).ready(function() {
    $("#settimedate").click(function(){
      $.post("settimedate.php",
      { time: $("#notifTime").val(),
        date: $("#notifDate").val()},
        function(data, status){
        window.location.reload(true)
        });
    });
  });
</script>

<script>
  $(document).ready(function() {
    $("#resetcandles").click(function(){
      $.post("resetcandles.php",
      {},
        function(){
        window.location.reload(true)
        });
    });
  });
</script>

<script>
  $(document).ready(function() {
    $("#reboot").click(function(){
      $.post("reboot.php",
      {},
        function(){
        window.location.reload(true)
        });
    });
  });
</script>

<script>
```

```
function displaytime(){
 $.ajax({
    url: "gettimedate.php",
    type: "POST",
    data: "",
    success: function(response) {
      var json = response;
      var obj = JSON.parse(json);
      document.getElementById("serverdatetime").innerHTML= obj.time + "
- " + obj.date;
    }
 });
}
</script>

<script>
window.onload=function(){
setInterval("displaytime()", 1000);
};
</script>

<style type="text/css">
    .ui-header .ui-title {margin: 0 10%}
</style>


</head>

<body>
<div data-role="page" id="Kerzensteuerung">
  <div data-role="header">
   <h2 >Adventskranzsteuerung</h2>
    <div data-role="navbar">
      <ul>
        <li><a href="#timedate">Datum / Uhrzeit</a></li>
        <li><a href="#about">Über...</a></li>
      </ul>
    </div>
  </div>

<!--  <div data-role="main" class="ui-content"> -->
  <div data-role="main">
<!--<fieldset> -->
<!--<div data-role="fieldcontain"> -->
<label style="display:inline" for="checkbox-based-flipswitch">1. Kerze
(GPIO 17):  </label>
<input type="checkbox" id="GPIO17" data-role="flipswitch"
onchange="switchValue(17)">
<!--</div> -->
```

```
<!--</fieldset> -->

<hr>

<!--<fieldset>-->
<!--<div data-role="fieldcontain">-->
<label style="display:inline" for="checkbox-based-flipswitch">2. Kerze
(GPIO 18):</label>
<input type="checkbox" id="GPIO18" data-role="flipswitch"
onchange="switchValue(18)" >
<!--</div>-->
<!--</fieldset>-->

<hr>

<!--<fieldset> -->
<!--<div data-role="fieldcontain"> -->
<label style="display:inline" for="checkbox-based-flipswitch">3. Kerze
(GPIO 27):</label>
<input type="checkbox" id="GPIO27" data-role="flipswitch"
onchange="switchValue(27)" >
<!--</div> -->
<!--</fieldset> -->

<hr>

<!--<fieldset> -->
<!--<div data-role="fieldcontain"> -->
<label style="display:inline" for="checkbox-based-flipswitch">4. Kerze
(GPIO 22):</label>
<input type="checkbox" id="GPIO22" data-role="flipswitch"
onchange="switchValue(22)" >
<!--</div> -->
<!--</fieldset> -->

<hr>

<!--<fieldset> -->
<!--<div data-role="fieldcontain"> -->
<button id="resetcandles">Reset Kerzen</button>
<!--</div> -->
<!--</fieldset> -->

<hr>

<!--<fieldset> -->
<!--<div data-role="fieldcontain"> -->
<button id="reboot">Reboot System</button>
<!--</div> -->
<!--</fieldset> -->
```

```
<hr>

  </div>

  <div data-role="footer">
    <h2>made by ChrvTh</h2>
  </div>
</div>

<div data-role="page" id="timedate">
  <div data-role="header">
    <h2>Systemzeit- und Datum einstellen:</h2>
  </div>
  <div data-role="navbar">
      <ul>
        <li><a href="#Kerzensteuerung">Zurück</a></li>
      </ul>
  </div>
  <div data-role="main">
    <br>
    <b>Systemzeit / Datum:</b> <p style="display:inline"
id="serverdatetime"> </p>

    <div data-role="content">

<!--    <div class="ui-field-contain"> -->
      <label for="notifTime">Systemzeit einstellen:</label>
      <input name="notifTime" id="notifTime" type="text" data-
role="datebox" data-options='{"mode":"timebox", "useClearButton":true,
"useLang":"de", "useNewStyle":true, "overrideTimeFormat": 24,
"themeButton": "b", "themeInput": "a", "theme": "b", "themeHeader":
"b"}' />
<!--    </div> -->
<!--    <div class="ui-field-contain"> -->
     <label for="notifDate">Systemdatum einstellen:</label>
     <input name="notifDate" id="notifDate" type="text" data-
role="datebox" data-options='{"mode":"datebox", "useClearButton":true,
"overrideDateFormat":"%d.%m.%Y"   ,  "useNewStyle":true,
"themeButton": "b", "themeInput": "a", "theme": "b", "themeHeader":
"b"}' />
<!--    </div> -->
<!--    <form> -->
     <button id="settimedate">Uhrzeit und Datum übernehmen</button>
<!--    </form> -->
    </div>

  </div>
</div>
```

```
<div data-role="page" id="about">
  <div data-role="header">
    <h2>Über diese Seite ...</h2>
  </div>
    <div data-role="navbar">
      <ul>
        <li><a href="#Kerzensteuerung">Zurück</a></li>
      </ul>
    </div>
  <div data-role="main"> <!-- class="ui-content"> -->
    <p>Webseite zur Steuerung eines Adventskranzes mit vier
elektrischen Kerzen.<br>
       <br>
       Entwickelt von:<br>
       Christoph von Thülen<br>
       Einumerstr. 15<br>
       31135 Hildesheim<br>
       E-Mail:<a href="mailto:Christoph@von-Thuelen.de">Christoph@von-
Thuelen.de </a>
    </p>
  </div>

</div>

</body>
</html>
```

/var/www/html/settimedate.php

settimedate.php

```
<?php
$result='';
if(count($_POST) > 0){

$date = $_POST['date'];
$time = $_POST['time'];

if($date == '') {$date = date("d.m.Y");}
if($time == '') {$time = date("H:i");}

$hour = '';
$minute = '';
$day = '';
$month = '';
$year = '';

$hour = substr($time, 0, strpos($time, ':'));
```

```
$minute = substr($time, (strpos($time, ':') + "1" ), strlen($time));

$day = substr($date, 0, strpos($date, '.'));
$date = substr($date, (strpos($date, '.') + "1" ), strlen($date));
$month = substr($date, 0, strpos($date, '.'));
$year = substr($date, (strpos($date, '.') + "1" ), strlen($date));

/*
exec('/bin/echo ' . '"Neue Stunde: ' . $hour . '"' . ' >>
/var/tmp/debug.log');
exec('/bin/echo ' . '"Neue Minuten: ' . $minute . '"' . ' >>
/var/tmp/debug.log');
exec('/bin/echo ' . '"Neuer Tag: ' . $day . '"' . ' >>
/var/tmp/debug.log');
exec('/bin/echo ' . '"Neuer Monat: ' . $month . '"' . ' >>
/var/tmp/debug.log');
exec('/bin/echo ' . '"Neues Jahr: ' . $year . '"' . ' >>
/var/tmp/debug.log');
*/
//exec('/bin/echo ' . '"Neue Uhrzeit: ' . $_POST['time'] . '"' . ' >>
/var/tmp/debug.log');
//exec('/bin/echo ' . '"Neues Datum: ' . $_POST['date'] . '"' . ' >>
/var/tmp/debug.log');

//exec('sudo /bin/echo ' . '"date ' . $month . $day . $hour . $minute .
$year . '"' . '>> /var/tmp/debug.log');
exec('sudo /bin/date ' . $month . $day . $hour . $minute . $year);
exec('sudo /sbin/hwclock -w');
exec('sudo /var/www/html/kerzeneinschalten.sh');

$result = 'Neue Uhrzeit: ' . $_POST['time'] . ';' . 'Neues Datum: ' .
$_POST['date'];
}
echo json_encode($result);
?>
```

/var/www/html/gettimedate.php

gettimedate.php

```
<?php
$date = date("d.m.Y");
$time = date("H:i:s");
$result = array("date" => $date, "time" => $time);
echo json_encode($result);
```

```
?>
```

/var/www/html/resetcandles.php

resetcandles.php

```
<?php
exec('sudo /var/www/html/kerzeneinschalten.sh');
echo json_encode();
?>
```

/var/www/html/reboot.php

reboot.php

```
<?php
exec('sudo shutdown -r now');
echo json_encode();
?>
```

/var/www/html/GPIOson.js

GPIOson.js

```
var g_GpioPins = [
{
gpio: '17', mode: 'out', value: 1
},
{
gpio: '18', mode: 'out', value: 1
},
{
gpio: '27', mode: 'out', value: 1
},
{
gpio: '22', mode: 'out', value: 1
},
{
gpio: '24', mode: 'in', value: 0
}
];
```

```
var g_RefreshTimer = null;

function debug(debug_data) {
  $.ajax({
    url: 'debug.php',
    type: 'post',
    datatype: 'json',
    data: debug_data,
    success: function(result) {
      // Do something with data that came back.
      var objData = jQuery.parseJSON(result);
      $('#UpdateResult').html(objData.status);
    }
  });
}

function getSetPin(pinData) {
  $.ajax({
    url: 'gpio.php',
    type: 'post',
    datatype: 'json',
    data: pinData,
    success: function(result) {
      // Do something with data that came back.
      var objData = jQuery.parseJSON(result);
      $('#UpdateResult').html(objData.status);
    }
  });
}

function switchValue(gpio) {
  for (var i_Pin in g_GpioPins) {
    if (g_GpioPins[i_Pin].gpio == gpio) {
      switch (g_GpioPins[i_Pin].mode) {
      case 'out':
        switch (g_GpioPins[i_Pin].value) {
          case 0:
            g_GpioPins[i_Pin].value = 1;
          break;
          case 1:
            g_GpioPins[i_Pin].value = 0;
          break;
        }
      break;
      case 'in':
        console.log('auslesen' + g_GpioPins[i_Pin].gpio);
      break;
      }
      getSetPin(g_GpioPins[i_Pin]);
```

```
      break;
    }
  }
  debug(gpio);
}

function autoRefresh() {
  //console.log($('#AutoRefresh').val());
  if ($('#AutoRefresh').val() == 'On') {
    autoRefreshStart();
  }
  else {
    autoRefreshStop();
  }
}

function autoRefreshStop() {
  //console.log('Auto OFF!');
  clearInterval(g_RefreshTimer);
  g_RefreshTimer = null;
}

function autoRefreshStart() {
  //console.log('Auto On!');
  g_RefreshTimer = setInterval('getSetPin(g_GpioPins[1])', 100);
  getSetPin(g_GpioPins[1]);
}
```

/var/www/html/gpio.php

gpio.php

```
<?php
if(count($_POST) > 0){
  $result=array();
  switch ($_POST['mode']){
   case 'out':
     exec('/usr/local/bin/gpio -g ' . 'mode ' . $_POST['gpio'] . '
out');
     exec('/usr/local/bin/gpio -g ' . 'write ' . $_POST['gpio'] . ' ' .
$_POST['value'],$gpioValue);
     //$result ='Success: Pin ' . $_POST['gpio'];
     $result['value'] = ($gpioValue);
   break;
   case 'in':
     exec('/usr/local/bin/gpio -g ' . 'mode ' . $_POST['gpio'] . '
in');
     exec('/usr/local/bin/gpio -g ' . 'read ' .
```

```
$_POST['gpio'],$gpioValue);
     //$result['value'] = ($gpioValue[0]);
     $result=array(status => $gpioValue[0], gpio=>$_POST['gpio']);
   break;
   default:
     $result='Error:wrong mode';
  }
}
else{
  $result='Error:';
}
echo json_encode($result);
?>
```

```
sudo su
ln -s /usr/bin/gpio /usr/local/bin/gpio
cd /var/www/html/
chown -R root:www-data *
```

```
sudo su
cd /
echo "www-data ALL=(ALL) NOPASSWD: /bin/date" >> /etc/sudoers
echo "www-data ALL=(ALL) NOPASSWD: /bin/echo" >> /etc/sudoers
echo "www-data ALL=(ALL) NOPASSWD: /sbin/hwclock" >> /etc/sudoers
echo "www-data ALL=(ALL) NOPASSWD: /sbin/shutdown" >> /etc/sudoers
echo "www-data ALL=(ALL) NOPASSWD: /usr/bin/gpio" >> /etc/sudoers
echo "www-data ALL=(ALL) NOPASSWD: /var/www/html/kerzeneinschalten.sh" >>
/etc/sudoers
```

R-Pi absichern (bei Stromausfall)

```
sudo su
apt-get remove --purge wolfram-engine triggerhappy logrotate dphys-swapfile
fake-hwclock
apt-get autoremove --purge
apt-get install busybox-syslogd
dpkg --purge rsyslog
# !ACHTUNG!: Syslog ab jetzt nur noch per "logread -f" lesbar
```

```
sudo su
cat /boot/cmdline.txt
dwc_otg.lpm_enable=0 console=serial0,115200 console=tty1 root=/dev/mmcblk0p2
rootfstype=ext4 elevator=deadline fsck.repair=yes rootwait
vi /boot/cmdline.txt
dwc_otg.lpm_enable=0 console=serial0,115200 console=tty1 root=/dev/mmcblk0p2
rootfstype=ext4 elevator=deadline fsck.repair=yes rootwait fastboot noswap
```

```
ro
```

```
sudo su
rm -rf /var/lib/dhcp/ /var/run /var/spool /var/lock /var/lib/misc/
/var/lib/ntp
ln -s /tmp /var/lib/dhcp; ln -s /tmp /var/run; ln -s /tmp /var/spool; ln -s
/tmp /var/lock; ln -s /tmp /var/lib/misc; ln -s /tmp /var/lib/ntp
```

```
sudo su
rm /var/lib/systemd/random-seed
ln -s /tmp/random-seed /var/lib/systemd/random-seed
nano /lib/systemd/system/systemd-random-seed.service
# add before "ExecStart=/lib/systemd/systemd-random-seed load"
ExecStartPre=/bin/echo "" >/tmp/random-seed
# reload services:
systemctl daemon-reload
```

```
sudo su
nano /etc/ntp.conf
# change "driftfile /var/lib/ntp/ntp.drift" to "driftfile
/var/tmp/ntp.drift"
```

```
sudo su
insserv -r bootlogs
insserv -r console-setup
```

```
sudo su
# change ''/etc/fstab'' from:
proc            /proc           proc    defaults          0       0
/dev/mmcblk0p1  /boot           vfat    defaults          0       2
/dev/mmcblk0p2  /               ext4    defaults,noatime  0       1
# to:
proc            /proc           proc    defaults            0       0
/dev/mmcblk0p1  /boot           vfat    defaults,ro         0       2
/dev/mmcblk0p2  /               ext4    defaults,noatime,ro 0       1

# For Debian Jessie
tmpfs           /tmp            tmpfs   nosuid,nodev        0       0
tmpfs           /var/log        tmpfs   nosuid,nodev        0       0
tmpfs           /var/tmp        tmpfs   nosuid,nodev        0       0
```

```
sudo su
nano /root/.bashrc
######
add below:
######
# set variable identifying the filesystem you work in (used in the prompt
below)
fs_mode=$(mount | sed -n -e "s/^.* on \/ .*(\(r[w|o]\).*/\1/p")
# alias ro/rw
```

```
alias ro='mount -o remount,ro / ; fs_mode=$(mount | sed -n -e "s/^.* on \/
.*(\(r[w|o]\).*/\1/p")'
alias rw='mount -o remount,rw / ; fs_mode=$(mount | sed -n -e "s/^.* on \/
.*(\(r[w|o]\).*/\1/p")'
# setup fancy prompt
export
PS1='\[\033[01;32m\]\u@\h${fs_mode:+($fs_mode)}\[\033[00m\]:\[\033[01;34m\]\
w\[\033[00m\]\$ '
# aliases for mounting boot volume
alias roboot='mount -o remount,ro /boot'
alias rwboot='mount -o remount,rw /boot'

finish() {
#echo "exec on exit"
mount -o remount,rw /
history -a
hwclock -w
mount -o remount,ro /
mount -o remount,ro /boot
}

trap finish EXIT
######
```

```
sudo su
touch /etc/bash.bash_logout
nano /etc/bash.bash_logout
# add:
mount -o remount,rw /
history -a
mount -o remount,ro /
mount -o remount,ro /boot
```

```
sudo su
nano /etc/apache2/envvars
# add below "export APACHE_LOG_DIR=/var/log/apache2$SUFFIX"
mkdir -p /var/log/apache2
```

Informationsquellen

GPI/Os ansteuern

- http://raspberrypiguide.de/howtos/raspberry-pi-gpio-how-to/
- http://www.tacticalcode.de/2012/12/erstes-raspberrypi-projekt-blinkende-led-in-5-schritten.html
- http://www.element14.com/community/servlet/JiveServlet/previewBody/73950-102-4-309126/GPIO_Pi2.png?01AD=3088KQEkrZwOCmZU3AsiTV2KJUhF8xmZNJmvdXgbwBChli9RrxAb99Q&01RI=45FA47C78F20BB4&01NA=na

- http://www.elektronx.de/tutorials/schalten-der-gpio-ein-und-ausgange/
- https://projects.drogon.net/raspberry-pi/wiringpi/download-and-install/
- http://www.raspberry-pi-geek.com/Archive/2014/07/PHP-on-Raspberry-Pi - Gut, funktioniert!

Webseite, PHP, Java Script und JQuery

- https://api.jquery.com/jquery.get/ - Wichtig!
- http://demos.jquerymobile.com/1.4.5/
- http://dev.jtsage.com/DateBox/jqueryui/timebox/
- http://www.w3schools.com/
- http://www.tinkerforge.com/de/doc/Kits/WeatherStation/PHPToWebsite.html
- https://wiki.selfhtml.org/wiki/Referenz:HTML/Zeichenreferenz

Date und Co.

- http://linuxcommando.blogspot.de/2009/11/fun-with-date-arithmetic.html

Access Point

- https://menzerath.eu/artikel/raspberry-pi-als-wlan-access-point-nutzen/
- http://www.tacticalcode.de/2013/02/raspberry-pi-als-accesspoint-oder-wlan-bridge.html
- http://jankarres.de/2015/06/raspberry-pi-wlan-access-point-einrichten/

Absichern gegen Stromausfall

- https://hallard.me/raspberry-pi-read-only/
- https://k3a.me/how-to-make-raspberrypi-truly-read-only-reliable-and-trouble-free/
- http://petr.io/en/blog/2015/11/09/read-only-raspberry-pi-with-jessie/

alter Kram - nicht mehr verwendet!

```
#NTP Dienst abstellen und deinstallieren, alternativ ''ntpdate''
installieren um per Netzwerk/Internet die Systemzeit aktualisieren zu
können.:
#root@raspberrypi2:~# update-rc.d ntp disable
#root@raspberrypi2:~# update-rc.d fake-hwclock disable
#apt-get remove fake-hwclock
#rm /etc/cron.hourly/fake-hwclock
#rm /etc/init.d/fake-hwclock
#apt-get install ntpdate
#update-rc.d hwclock.sh enable
```

Test:

```
root@raspberrypi2:~# date -s "Jan 01 2016 12:00:00"
root@raspberrypi2:~# date
```

```
root@raspberrypi2:~# hwclock -w
root@raspberrypi2:~# hwclock -r
root@raspberrypi2:~# reboot
root@raspberrypi2:~# hwclock -r
```

Manuelles Synchronisieren der Systemzeit:

```
#root@raspberrypi2:~# ntpdate ptbtime1.ptb.de
```