# Standheizung Josty Anleitung

Inhaltsverzeichnis

[Standheizung Josty Anleitung 1](#_Toc23516409)

[Hardware 2](#_Toc23516410)

[Verbaute Module: 2](#_Toc23516411)

[Arduino Pinbelegung: 2](#_Toc23516412)

[Befehle 2](#_Toc23516413)

[GSM SMS Befehle: 2](#_Toc23516414)

[ESP8622 Befehle 3](#_Toc23516415)

[Einbindung in Fhem 3](#_Toc23516416)

[Programmierung: 3](#_Toc23516417)

[Befehle: 3](#_Toc23516418)

[Programmierung per SMS 3](#_Toc23516419)

[Bedienung: 4](#_Toc23516420)

[Steuerung per SMS 4](#_Toc23516421)

[Steuerung per WLan ( ESP8266 ) 4](#_Toc23516422)

[Einstellungen im ESP8622 ( ESPeasy ) 4](#_Toc23516423)

[Arduino Code 5](#_Toc23516424)

[Arduino QuellCode 5](#_Toc23516425)

[ESP8266 Quellcode 5](#_Toc23516426)

# Hardware:

## Verbaute Module:

SIM900 <https://www.makershop.de/module/gsm-gps/sim900a-quad-band-gsm/>

Arduino Mega 2560

SD Card Reader <https://www.makershop.de/module/schnittstellen/adafruit-microsd-card-breakout/>

Ublox NEO 6M-GPS Modul GY-GPS6MV2 UART <https://www.makershop.de/module/gsm-gps/ublox-neo-6m-gps/>

ESP8266 <https://www.makershop.de/plattformen/nodemcu/esp32-board-stiftleisten/>

LM2596S DC-DC Spannungsregler Step-Down Converter <https://www.ebay.de/itm/LM2596S-DC-DC-Spannungsregler-Step-Down-Converter-Regler-einstellbar-1-2V-30V-3A/392315076833?ssPageName=STRK%3AMEBIDX%3AIT&_trksid=p2057872.m2749.l2649>

## Arduino Pinbelegung:

Status Blinkled Pin 🡪 8

Taster Standheizung Start 🡪 Pin 7

Sim900 Pins TX RX 🡪 18RX 19TX

GPS Pins 🡪 16TX 17RX

SD Karte Pin 🡪 CS-5 SCK-13 MOSI-11 MISO-12

**CS**–  Pin für die Erfassung der SD-Karte, anzuschließen an Pin D5 des Arduinos  
**SCK** –  Pin für die Synchronisation des Datentransfers durch die Master Pins, anzuschließen an Pin D13 des Arduinos  
**MOSI** –  Pin für den Datentransfer (**M**aster **O**ut **S**lave **I**n) , anzuschließen an Pin D11 des Arduinos  
**MISO** –  Pin für den Datentransfer (**M**aster **I**n **S**lave **O**ut) , anzuschließen an Pin D12 des Arduinos

# 

# Befehle

## GSM SMS Befehle:

Groß/Kleinschreibung spielt keine Rolle da es im Arduino Code alles auf Kleinschreibung gewandelt wird.

on -> Schaltet Standheizung für 30 min ein.

off 🡪 Standheizung ( vorzeitig ) ausschalten.

start0730 🡪 Programmiert die Standheizung für den nächsten einmaligen Start

Timer15 🡪 Lässt Standheizung für xx Minuten laufen

Init 🡪 Initialisierung GSM900 Modul

## ESP8622 Befehle

Mittels http Befehl http://IPAdressevomESP/control?cmd=pulse,14,1,2000 kann die Standheizung Aktiviert werden. Die ESP8622 sendet einen 2000 ms impuls von D5 an den Arduino Eingangspin 6 und startet damit die Standheizung für die Standartlaufzeit ( 30 Min)

Bsp: http://192.168.4.151/control?cmd=pulse,14,1,2000

Mittels Haussteuerung wie z.b. Fhem kann der ESP ausgelesen werden.

# Programmierung:

## Programmierung Befehlsübersicht:

001#0172999999

clear#001

timezone#2

## Programmierung per SMS

Spannung auf das Modul geben danach hat man 2 min Zeit um eine SMS mit der eigenen Rufnummer in folgendem Format zu Senden. Diese wird dann auf Speicherplatz 001 bis 005 gespeichert.

Bsp. Speicherplatz#Nummer 🡪 001#0172999999

Damit wird die Nummer auf Speicherplatz 1 im Eeprom vom Arduino gespeichert.

Mittlels sms clear#001 wird die Rufnummer wieder gelöscht.

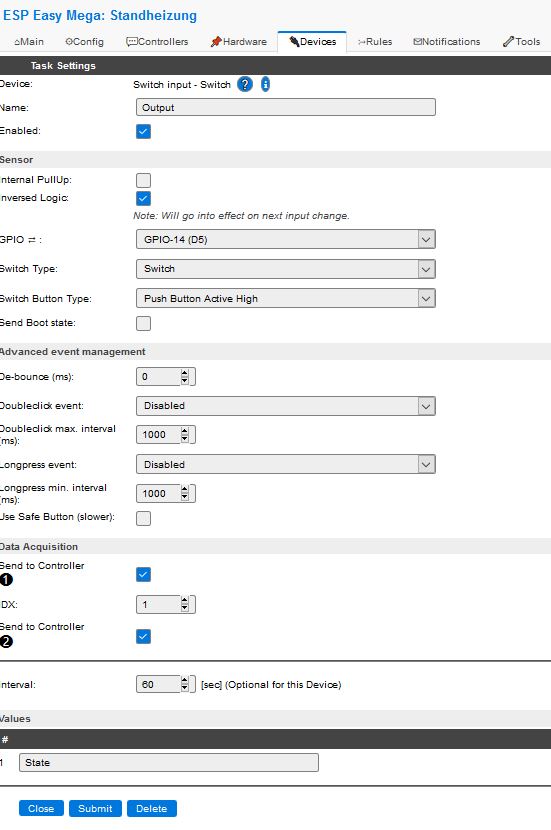
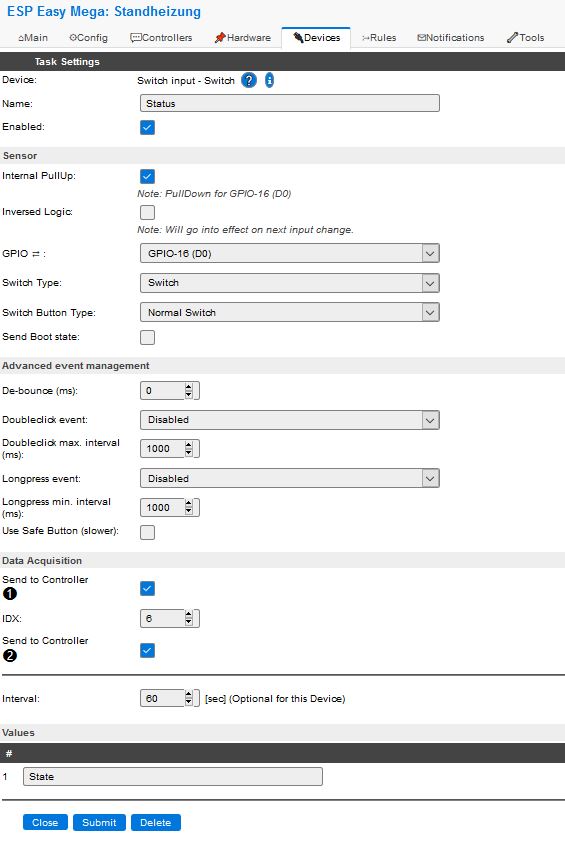
Die Uhrzeit bekommt das Modul vom GPS Empfänger diese Uhrzeit kommt in UTC

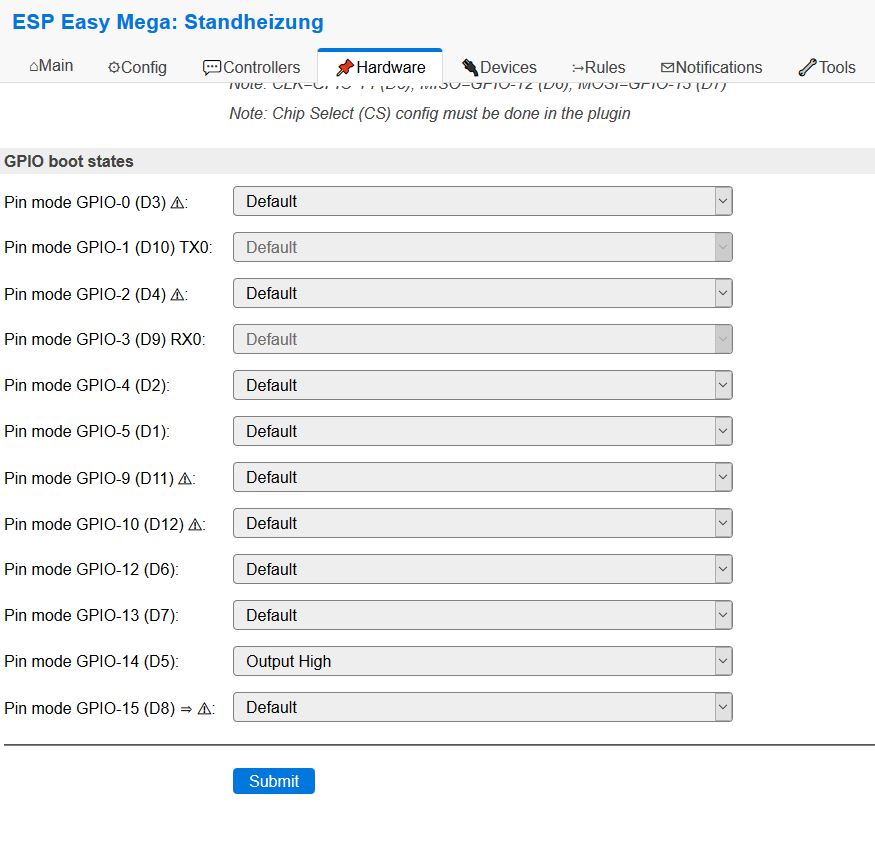
Mittels Befehl SMS Befehl timezone#-2 wird z.b. 2 stunden dazugezählt.

# Bedienung:

## Steuerung per WLan ( ESP8266 )

# Einstellungen im ESP8622 ( ESPeasy )





# Arduino Code

## Arduino QuellCode

// RTC Date and time functions using a DS1307 RTC connected via I2C and Wire lib

#include <Wire.h>

#include "RTClib.h"

RTC\_DS1307 rtc;

// Serielle schnittstelle zum Sim900

#include <SoftwareSerial.h>

#define Debug

// Configure software serial port for SIM900 Modult

static const int RXPin = 2, TXPin = 3;

static const uint32\_t SIM900Baud = 9600;

SoftwareSerial SIM900(RXPin, TXPin);

static const uint32\_t SerialBaud = 19200; // Baudrate für PC Serial Baud

char inchar; // variable to store the incoming character

int Relay = 10; // Pin für Relays Standheizung an

int switchs = 7; // Pin für Stanheizung Starttaster

int esppin = 6;

const int HeizungLed = 8; // Led Standheizung Aktiv

int HeizLedState = LOW; // ledState used to set the LED

unsigned long previousMillis = 0; // will store last time LED was updated

const long interval = 200; // interval at which to blink Heizungs Led (milliseconds)

unsigned long currentMilliss = millis();

String startzeitH = "HH";

String startzeitM = "MM";

unsigned long startTime;

unsigned long shutdownTime;

bool RelayOn;

int timer;

int tmpSerialData;

String serialInputBuffer;

int val = 0;

int esp = 0;

const bool EIN = true;

const bool AUS = false;

const unsigned long DEFAULTTIME = 5000; // Standart Laufzeit der Standheizung 30 Min 1800000ms

void setup()

{

pinMode(HeizungLed, OUTPUT);

pinMode(Relay, OUTPUT); //Relays Pin auf Output stellen

// Taster zum Standheizung Aktivieren

pinMode(switchs, INPUT); // Taster als Input

digitalWrite(switchs, HIGH); // Interner Pull Up

pinMode(esppin, INPUT);

digitalWrite(esppin, LOW);

digitalWrite(Relay, LOW); // Relays Aus am Start

Serial.begin(SerialBaud);

// wake up the GSM shield

SIM900.begin(SIM900Baud);

delay(6000);

//#################### start RTC ##################

if (! rtc.begin()) {

Serial.println("Couldn't find RTC");

while (1);

}

// rtc.adjust(DateTime(2019, 10, 29, 12, 00, 00));

rtc.adjust(DateTime(F(\_\_DATE\_\_), F(\_\_TIME\_\_))); // Sketchzeit erstellung

if (! rtc.isrunning()) {

Serial.println("RTC is NOT running!");

rtc.adjust(DateTime(F(\_\_DATE\_\_), F(\_\_TIME\_\_))); //???

rtc.adjust(DateTime(2019, 10, 29, 12, 8, 0));

Serial.println("RTC adjusted!");

}

DateTime now = rtc.now();

Serial.print(now.hour(), DEC);

Serial.print(':');

Serial.print(now.minute() / 10 , DEC);

Serial.print(now.minute() % 10 , DEC);

Serial.print(':');

Serial.print(now.second() / 10 , DEC);

Serial.print(now.second() % 10 , DEC);

Serial.println();

//############ ende RTC ###############

Serial.println("+++ Sende AT +++");

SIM900.println("AT"); //Once the handshake test is successful, it will back to OK

delay(300);

while(SIM900.available())

{

Serial.println(SIM900.read());//Forward what Software Serial received to Serial Port

}

Serial.println("+++ Warte auf Sim900 INIT ..... +++");

// InitSim900 ();

}

void HeizungEin(bool state, unsigned long shutdownValue)

{

if (state)

{

// Serial.println("Abschaltzeit: " + shutdownValue);

Serial.println("Relay ON");

digitalWrite(Relay, HIGH);

RelayOn = true;

startTime = millis();

shutdownTime = shutdownValue;

// Serial.println(startTime);

// Serial.println(shutdownTime);

// Serial.println(startTime + shutdownTime);

}

else

{

Serial.println("Relay OFF");

digitalWrite(Relay, LOW);

RelayOn = false;

}

}

void InitSim900()

{

#ifdef Debug

Serial.println("+++ Begin Setup +++");

#endif

SIM900.flush();

delay(120);

SIM900.println("AT+CNMI=2,2,0,0,0");

delay(140);

while(SIM900.available())

{

Serial.write(SIM900.read());//Forward what Software Serial received to Serial Port

}

SIM900.println("AT+CSQ"); //Signal quality test, value range is 0-31 , 31 is the best

delay(140);

while(SIM900.available())

{

Serial.write(SIM900.read());//Forward what Software Serial received to Serial Port

}

SIM900.println("AT+CCID"); //Read SIM information to confirm whether the SIM is plugged

delay(140);

while(SIM900.available())

{

Serial.write(SIM900.read());//Forward what Software Serial received to Serial Port

}

SIM900.println("AT+CREG?"); //Check whether it has registered in the network

delay(130);

while(SIM900.available())

{

Serial.write(SIM900.read());//Forward what Software Serial received to Serial Port

}

SIM900.println("AT+CLIP=1"); //rufnummer anzeige aktivieren CLIP

delay(130);

while(SIM900.available())

{

Serial.write(SIM900.read());//Forward what Software Serial received to Serial Port

}

SIM900.println("AT+CMGF=1"); // set SMS mode to text

delay(130);

while(SIM900.available())

{

Serial.write(SIM900.read());//Forward what Software Serial received to Serial Port

}

SIM900.println("AT+CMGD=1,4"); // delete all SMS

Serial.println("+++ All SMS geloescht +++");

delay(130);

while(SIM900.available())

{

Serial.write(SIM900.read());//Forward what Software Serial received to Serial Port

}

Serial.println("+++ Ende Setup +++");

}

void DeleteAllSMS()

{

SIM900.println("AT+CMGD=1,4"); // delete all SMS

Serial.println("Alle SMS geloescht void");

delay(500);

SIM900.println("AT+CMGF=1");

delay(200);

while(SIM900.available())

{

Serial.write(SIM900.read());//Forward what Software Serial received to Serial Port

}

}

void loop()

{

DateTime now = rtc.now();

if ((millis() == 20000) or (millis() == 20001)) {

Serial.println(millis());

InitSim900();

}

// ++++++++++++++ Per ESP Pin standheizung einschalten ++++++++++++++++++++++++++

esp = digitalRead(esppin);

// Serial.println(esp);

if (esp == 1)

{

if (not RelayOn)

Serial.println("+++ ESP ausgeloest +++");

HeizLedState = HIGH;

HeizungEin(EIN, DEFAULTTIME);

InitSim900();

// DeleteAllSMS();

}

// ++++++++++++++ Per Taster standheizung einschalten ++++++++++++++++++++++++++

val = digitalRead(switchs); // read the input pin

// Serial.println(val);

if (val == 0)

{

if (not RelayOn)

HeizLedState = HIGH;

HeizungEin(EIN, DEFAULTTIME);

InitSim900();

// DeleteAllSMS();

DateTime now = rtc.now();

Serial.print("Aktuelle Uhrzeit ");

Serial.print(now.hour(), DEC);

Serial.print(':');

Serial.print(now.minute() / 10 , DEC);

Serial.print(now.minute() % 10 , DEC);

Serial.print(':');

Serial.print(now.second() / 10 , DEC);

Serial.print(now.second() % 10 , DEC);

Serial.println();

}

// Relays nach 30 min Abschalten

if (RelayOn)

{

if (startTime + shutdownTime < millis())

{

HeizungEin(AUS, 0);

}

}

//If a character comes in from the GSM...

if (SIM900.available())

{

// Lese alles bis zum Zeichen \n

serialInputBuffer = SIM900.readStringUntil('\n');

delay(10);

//Serial.println(test);

//Serial.println(test.substring(1,20));

// ################################### Anruf ############################

// Ein Anruf kommt rein mit der nummer 491622742063

if (serialInputBuffer.substring(1, 21) == "CLIP: \"491622742063\"")

{

Serial.println(serialInputBuffer.substring(1, 20));

HeizungEin(EIN, DEFAULTTIME);

}

//Serial.println(test);

//Serial.write(SIM900.read()); // -> Zeichenfolge

//Serial.println(SIM900.read()); // -> einzeles Zeichen

//######### SMS Empfang #########

// on off timer15 start0730

//HeizungSteuernUhrzeit(bool Zustand, string time)

// alle sms befehle in kleinschreibung umwandeln

serialInputBuffer.toLowerCase();

// Serial.println(serialInputBuffer.substring(0,2));

// Serial.println("#### Start Debug Info ###");

// Serial.println("Input Buffer: " + serialInputBuffer);

// Serial.println("Input Buffer Stelle 1-6: " + serialInputBuffer.substring(0, 5));

// Serial.println("Input Buffer Stelle 7-8: " + serialInputBuffer.substring(6, 8));

// Serial.println((serialInputBuffer.substring(6, 8).toInt()) \* 60000);

// Serial.println("#### End Debug Info ###");

if (serialInputBuffer.substring(0, 3) == "off")

{

HeizungEin(AUS, 0);

DeleteAllSMS();

}

else if (serialInputBuffer.substring(0, 2) == "on")

{

HeizungEin(EIN, DEFAULTTIME);

DeleteAllSMS();

}

else if (serialInputBuffer.substring(0, 5) == "timer")

{

//Serial.println("Stelle 7-8: " + serialInputBuffer.substring(5, 7));

HeizungEin(EIN, (serialInputBuffer.substring(5, 7).toInt()) \* 60000);

DeleteAllSMS();

}

else if (serialInputBuffer.substring(0, 4) == "init")

{

InitSim900();

}

else if (serialInputBuffer.substring(0, 5) == "start")

{

startzeitH = serialInputBuffer.substring(5, 7);

startzeitM = serialInputBuffer.substring(7, 10);

Serial.println("startzeit programmiert auf " + startzeitH + " " + startzeitM);

}

}

// Serial.println(now.minute());

// if startzeit.substring =

//

// Serial.print(now.hour(), DEC);

// Serial.print(':');

// Serial.print(now.minute() / 10 , DEC);

// Serial.print(now.minute() % 10 , DEC);

// Serial.print(':');

// Serial.print(now.second() / 10 , DEC);

// Serial.print(now.second() % 10 , DEC);

// Serial.println();

// #################### Blinkende LED für Standheizung Aktiv ###############################

// check to see if it's time to blink the LED; that is, if the difference

// between the current time and last time you blinked the LED is bigger than

// the interval at which you want to blink the LED.

unsigned long currentMillis = millis();

if (currentMillis - previousMillis >= interval)

{

previousMillis = currentMillis; // save the last time you blinked the LED

if (RelayOn) {

if (HeizLedState == LOW)

{

HeizLedState = HIGH;

// Serial.println("test Heizung An");

} else

{

HeizLedState = LOW;

// Serial.println("Led Heizung Aus");

}

digitalWrite(HeizungLed, HeizLedState); // set the LED with the ledState of the variable:

}

else digitalWrite(HeizungLed, LOW);

}

}

## ESP8266 Quellcode

Es wird die Datei ESP\_Easy\_mega-20191028\_normal\_core\_241\_ESP8266\_4M1M.bin

Mit dem ESP Flashtool auf den ESP8266 geladen, dieser spannt beim erstmaligen Start einen Accesspoint auf und damit kann man diesen Konfigurieren.