## Was ist der Arduino

Ein Microcontrollerboard (um 20 Euro oder um 3 in China) mit einer einfachen IDE zum Porgrammieren in C. Als IDE eignet sich aber auch VisualStudioCode mit der PlatformIO Erweiterung sehr gut, die viele andere Ports (ca 500) unterstützt.

Der Arduino besitzt mehrere digitale Ein und Ausgänge und ein paar analoge Eingänge, deren Werte von einem 10 Bit AD-Wandler in 1024 Stufen zwischen 0 und 5V dargestellt werden.  
Analoge Ausgänge lassen sich nur über Pulsweitenmodulation erzeugen:

* ein ständiger Wert am Ausgang erzeugt 5V
* wenn der Wert immer kurz gleichlang ein und ausgeschaltet wird 2,5V
* sonst eben im Verhältnis der Ein und Ausschaltdauer

Das fertig compilierte Programm wird dann auf den Arduino übertragen und rennt dort sobald die Übertragung fertig ist.  
Deshalb besteht ein Arduinoprogramm immer aus zwei Funktionen:

setup() … da werden Startwerte festgelegt

loop() … wie der Name schon sagt eine Endlosschleife mit der Funktionalität, das wäre wie main und darin ein while(true)

wichtige Befehle am Arduiono

digitalWrite(pin,spannung); setzt die Leitung auf 0 (LOW) oder 5V (HIGH) bzw

x=digitalRead(pin); schaut ob der pin HIGH oder LOW ist

x=analogRead(pin); liest die Spannung am pin ein. Von 0V (x ist 0) bis 5V (x ist 1024)

delay(ms); wartet ms lang

x=millis(); Zeitmessung, Millisekunden, die seit dem Start des Arduino vergangen sind

## Arduino lesen vom und schreiben auf den seriellen Port (USB)

Serielle (USB) Schnittstellenbefehle:

serial.available()

serial.begin()

serial.print(byte), serial.println(String), x= serial.read, y=serial.readbyte

## Zusammenarbeit Arduino Windows, Sprachsteuerung, Motor schalten

zB über die Windows Sprachsteuerung einen Aktor einschalten:

1. Sprachsteuerung ist in Windows integriert (ab Win 7 funktioniert sie gut, Probleme falls deutsches Windows und englische Visual Studio)
2. Lernen des Frequenzmusters des Benutzers (arbeitet lokal, nicht wie alexa und co)
3. Nur für Motor geht es aber gut
   * Wenige Worte, die sich deutlich unterscheiden
     + Möglichst einsilbig (ein, aus, stop, los)
     + Auch wenns laut ist im Hintergrund.
   * Gutes Mikrofon mit Rauschunterdrückung.
4. Windows GUI mit Buttons auf denen die Schlüsselwörter stehen

LOS

STOP

* + Im Event Handler des Buttons, OnButtonClick 🡪 USB Schnittstelle ansprechen
  + USB ansprechen (geht über die CSerial-Klasse, die in Visual Studio dabei ist.
    - x = open(Com#) , die Nummer sieht man in der Arduino IDE
    - x.read
    - x.write
  + am Arduino
    - SerialRead => Digital Pin ein oder Ausschalten (DigitalWrite) in Schleife

240 V

Relais

PC

Arduino

USB

USB

GND

0 5V

* + init{}  
    loop()  
    {
  + char s[10];
  + s=SerialRead…
  + If(strcmp(s,“LOS“)
  + DigitalOut //5V am Ausgang setzen…  
    }  
    Man muss sich halt ein eigenes Protokoll überlegen zB einen String, der die nötigen Befehle enthält
  + Relais schaltet mit einer kleinen Spannung eine Große
    - Ist echtzeitfähig
  + Arduino arbeitet in der Schleife, ist echzeitfähig, single Task Prozessor
  + USB ist Streitfall ob echtzeitfähig, zum isochronen Transfer zwingen (der ist echtzeitfähig)
  + Windows NEIN (Uhr ist zwar sehr genau, Windows schaut aber nur alle ca 40ms auf der „Uhr“ nach wie spät es ist, kleinere Zeiten werden nicht aufgelöst. Man weiß dann zwar, dass und um wieviel es zu spät ist, aber das darf bei echtzeitfähigen Systemen nicht passieren)

## Arduino GSM shield zum Versenden von SMS, Aufgabe Remote Temperaturanzeige

# Lösungen

* Hochwertigere Heizungssteuerungen bieten die Möglichkeit der Remote-Kontrolle, benötigen dazu aber eine permanente Internetverbindung und üblicherweise die Einrichtung einer vpn Verbindung oder einen dyndns Service, damit der Netzwerk von außen erreichbar bleibt, wenn sich die IP-Adresse ändert.
* Es gibt um 80 Euro fertige Module die über einen SIM-Kartenslot verfügen und bei Unterschreiten einer gewissen Temperatur eine SMS senden.

Selber programmieren:

* Ein PC mit einem AD/Wandler und Internetstick. Einfach zu programmieren aber teuer.
* Ein Raspberry Pi mit AD Wandler und USB Internet Stick (kommt auf ca 70 Euro)
* Selber bauen mit einem Arduino, einen einfachen Temperatursensor und einem GSM-Shield. Hardware-Kosten Arduino Klon aus China 5 Euro, Temperatursensor 2 Euro, GSM Shield 30 Euro, Netzteil 5 Euro, ergibt < 50 Euro.

# Umsetzung:

# Arduino

Arduino ist eine quelloffene Plattform für Hardwarenahe Programmierung. Grundsätzlich bauen die Boards auf einen Atmel AVR Mikrocontroller auf. Ein Arduino verfügt über mehrere Analoge und Digitale Ein- und Ausgänge von denen eine bestimmte Anzahl auch PWM-Signale erzeugen können.

Die Software für Arduino-Boards wird in der kostenlosen IDE entwickelt. Durch diverse Bibliotheken wird der C-Code simplifiziert.

Arduinos besitzen digitale Ein- und Ausgänge und Analoge Eingänge. Ein digitaler Ausgang wird auf 5V gesetzt, wenn er gesetzt wird. Das kann als Spannungsversorgung für einen Sensor verwendet werden. Die Spannung am Temperatursensor würde dann je nach Temperatur zwischen 0V und der Versorgungsspannung 5 V liegen.  
Die Analogeingänge des Arduino wandeln eine Spannung in eine Zahl um. Bei  
5 V wird eine 1023 erzeugtb bei 0 V eine 0. So bekommt man zur aktuellen Temperatur einen Messwert zwischen 0 und 1024

# Temperaturerfassung

Für die Messung reicht ein einfacher PTC – Widerstandssensor, den man bei jedem Elektronik Händler um unter 2 Euro bekommt (PTC- positiver Temperatur Koeffizient bedeutet, dass je höher die Temperatur desto höher der Widerstand und desto höher die Spannung ist).  
Zum Kalibrieren liest man den Analogwert bei zwei verschiedenen Temperaturen ein und stellt zwei Gleichungen auf. Da der Zusammenhang linear ist gilt  
T1 = k \* Analogwert1 + d  
T2 = k \* Analogwert2 + d

Daraus kann man k und d ermitteln und erhält eine Formel für die Umrechnung der analogwert in Temperaturen

temp = (k \* analogRead(tempPin) + d)

# Kommunikation Arduino – GSM-Modem

Das Senden der SMS erfolgt über ein sogenanntes GSM shield, das einfach auf den Arduino aufgesteckt wird. Shields sind kleine Platinen deren Anschlüsse so angebracht sind, dass sie übereinander gestapelt auf das Arduionboard aufgesteckt werden können und dessen Funktionalität auf diese Weise erweitern.

Die Kommunikation erfolgt über den I2C bus oder eine serielle Schnittstelle zwischen Arduino und shield. Der Arduino sendet über die serielle Schnittstelle AT Befehle, die die Funktion des shields steuern

Dieser AT-Befehlssatz dient zur Steuerung eines Modems. Für die Versendung von SMS werden zB folgende Befehle benötigt:

* AT+CMGF=1
* AT+CMGS="+436642753600"

Der Erste AT-Befehl signalisiert dem Chip, dass eine SMS verschickt werden soll. „AT+CMGS“ startet den SMS Prozess mit der gegebenen Telefonnummer. Nach diesem Befehl signalisiert der Chip mit „>“ dass eine Nachricht eingegeben werden kann. Anschließend wird die Nachricht mit Strg + Z oder dem Hexcode „0x1a“ bestätigt werden.

Neben der Möglichkeit Kurznachrichten zu senden kann man übrigens auch Anrufe tätigen. Der Audio Out- und Input erfolgt über die Klinkenbuchsen des Shields.

Eventuell wäre für den Arduino noch eine kleine unabhängige Spannungsversorgung (eine Powerbank) praktisch, die noch solange Strom liefert, dass der Arduino noch schnell eine SMS senden kann, wenn zB der Strom ausfällt. Auch das lässt sich leicht messen wenn zB ein Stromdruchflusszähler um das Stromkabel, das zum Arduino führt, gelegt wird und dieser Sensor ebenfalls an eine Analogeingang angeschlossen wird. Bei Stromausfall würde die Spannung auf 0 fallen.

Beispielcode (Pseudocode) am Arduino

ein Arduino Programm hat immer die Funktionen

setup zum Initialisieren und loop zum Ausführen der Aufgabe

void setup() //das ist die Standardroutine die immer am Beginn ausgeführt wird

{

k und d setzen  
Grenztemperatur setzen, unterhalb der eine SMS gesendet werden soll  
serielle Schnittstellenparameter zum shield setzen  
digitalen Port auf true setzen, der den Sensor mit Strom versorgt

}

void loop() //das wird ununterbrochen ausgeführt

{

T= k \* AnalogRead(Sensorpin) + d  
 if(T<Grenzwert und gesendet==false)  
 {

AT Befehle an das shield senden um SMS zu starten  
Warten bis > zurückkommt  
SMS Text an das shield senden  
0x1a senden, damit sms abgeschickt wird  
gesendet auf true setzen, damit nicht alle 5 Sekunden eine SMS gesendet wird

}

delay(3600); /jede Minute eine Messung machen

}