Arduino verstehen und anwenden

Teil 6: Sensortechnik und Messwerterfassung

Der sechste Teil der Artikelserie „Arduino verstehen und anwenden" hat das weite Gebiet der Sensortechnik und der controllerbasierten Messwerterfassung zum Thema.

Hier ist der Mikrocontroller voll in seinem Element. Denn in diesem Anwendungsbereich unterscheiden sich Controller ganz wesentlich von ihren technischen Gegenspielern, den Prozessoren. Mikrocontroller zeichnen sich insbesondere dadurch aus, dass sie über integrierte Messwandlereinheiten verfügen. So besitzen auch die Controller der Arduino-Serie mehrere Analogeingänge. Damit lassen sich die unterschiedlichsten Messanwendungen realisieren. Zu den Themen dieses Beitrags zählen entsprechend die folgenden Aufgaben:

- Erfassung analoger Messdaten
- Auswertung von Sensorsignalen
- Ausgabe und Verarbeitung von Messwerten

Im Rahmen der Praxisbeispiele werden die folgenden Anwendungen vorgestellt:

- Erfassung einer Potentiometerposition durch Messung analoger Spannungswerte
- Elektronisches Computer-Thermometer mit Übertemperaturalarm
- Elektronisches Luxmeter

Erfassung von Messwerten

Die Erfassung von elektronischen Messdaten gehört mit zu den zentralen Aufgaben eines Mikrocontrollers. Das Einlesen von digitalen Signalen stellt die einfachste Möglichkeit dar. Binäre Spannungspegel (0 V oder 5 V) können von jedem I/O-Pin erfasst werden. Allerdings muss man auch bei dieser vergleichsweise einfachen Aufgabe häufig verschiedene Probleme beachten. So stellt etwa das sogenannte „Prellen", also die Mehrfacherfassung eines Schaltvorgangs aufgrund von mechanischen Effekten, ein nicht zu unterschätzendes Problem bei der Ansteuerung von einfachen Tastern dar.

Aber auch analoge Spannungswerte können von modernen Mikrocontrollern direkt gemessen werden. Die meisten gängigen Controllertypen haben dazu einen sogenannten A/D-Wandler (**A**nalog-zu-**d**igital-Umsetzer, auch ADC für engl. Analog to Digital Converter) integriert. In den entsprechenden Controllern sind meist noch sogenannte Analogmultiplexer vorhanden. Diese erlauben dann die nahezu simultane Erfassung von 6, 8 oder sogar mehr analogen Kanälen.

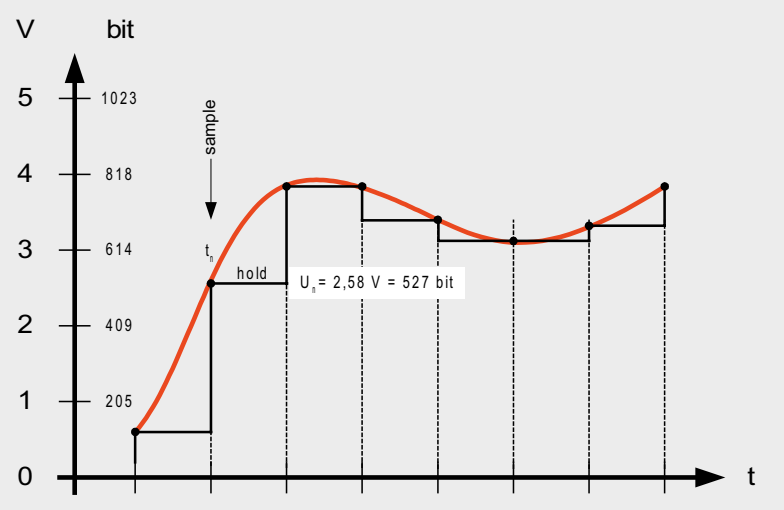

Bild 1: Erfassung analoger Messwerte

ADCs sind wichtige Bestandteile vieler Geräte der modernen Kommunikations-, Unterhaltungs- und Haushaltselektronik. Digitale Thermometer oder Wetterstationen können auf ADCs ebenso wenig verzichten wie MP3-Player, Mobiltelefone oder digitale Kameras. Zudem werden die Wandler in der Messwerterfassung in industriellen Anwendungen, zur Steuerung von Maschinen und in Kraftfahrzeugen benötigt.

Die Auflösung eines ADCs bestimmt die maximale Genauigkeit, mit der das analoge Eingangssignal digitalisiert werden kann. Die effektive Präzision ist durch weitere Fehlerquellen des ADCs jedoch geringer als die theoretische Auflösung. So führen etwa Nichtlinearitäten oder Abgleichfehler zu zusätzlichen Abweichungen.

Die Wandlungsdauer für einen einzelnen Messwert hängt vom Wandlungsverfahren ab. Neben extrem schnellen Verfahren existieren auch langsame, integrierende Methoden, die sich dafür durch hervorragende Unterdrückung von Störeinkopplungen auszeichnen.

Die Auflösung, mit der die analoge Größe gemessen wird, bewegt sich zwischen 1 und 24 Bit. Die durch die Wandlung entstehende Abweichung zwischen dem tatsächlichen Messwert und dem digitalen Ausgabewert bezeichnet man als Quantisierungsfehler. Er entsteht u. a. durch unvermeidbare Rundung, Nichtlinearitäten sowie Störeinstreuungen etc.

Bild 1 zeigt, wie der rot dargestellte analoge Spannungsverlauf mit einem ADC digitalisiert wird. Der ADC führt zu bestimmten Zeitpunkten t_n eine Spannungsmessung durch. Dieser Spannungswert wird dann in einen Digitalwert umgewandelt. Bei einer Auflösung von 10 Bit und einer Referenzspannung von 5 V ergibt sich so für eine Spannung von

Tabelle 1

ADC-Wandlerverfahren

Verfahren	Arbeitsweise	Vorteile	Nachteile	Anwendung
Flash- oder Parallelwandler	pro Ausgangswert 1 Komparator	sehr schnell	teuer, hohe Leistungsaufnahme	Digitaloszilloskope, Forschung und Entwicklung
Sukzessive Approximation	stufenweise Annäherung, Wägeverfahren	schnell, genau	technisch aufwendig	Standardverfahren, interne A/D-Wandler von Mikrocontrollern
Single Slope, Dual Slope	Spannungsmessung erfolgt über Zeitmessung von Ladevorgängen	preisgünstig, gute Linearitäten	langsam	Multimeter
Delta-Sigma	Komparator und Logik-Elemente	preisgünstig	langsam	Präzisionsmessungen bis zu 24 Bit, Audio

beispielsweise 2,58 V ein abgerundeter digitaler Ergebniswert von Q = 527:

$$Q = 2{,}58\ V / 5\ V * 1023 = 527$$

Je nach Anwendungsfall werden unterschiedliche Anforderungen an einen ADC gestellt. Für die verschiedenen Aufgaben wurden daher unterschiedliche Verfahren entwickelt und implementiert. Die wichtigsten ADC-Verfahren sind in Tabelle 1 zusammengefasst. Wie diese zeigt, besitzen alle Verfahren ihre spezifischen Vor- und Nachteile. Die sukzessive Approximation stellt aber einen guten Kompromiss für viele Anwendungen dar. Sie wird daher in vielen Mikrocontrollern und insbesondere auch bei den im Arduino verwendeten AVR-Typen eingesetzt. Die im Arduino verwendeten ADCs zeichnen sich durch die in Tabelle 2 aufgeführten Leistungsmerkmale aus.

Tabelle 2

Eigenschaften des ADC-Wandlers im Arduino

Analogeingänge	**6 (A0–A5)**
Auflösung	10 Bit
Wandlungszeit	max. 260 µs

Praktische Anwendung: Erfassung einer Potentiometerposition

Nachdem nun die prinzipiellen Eigenschaften von ADCs dargelegt wurden, soll die Funktionsweise der Analog-digital-Converter anhand eines Anwendungsbeispiels verdeutlicht werden. Hierzu sollen die von einem Potentiometer erzeugten analogen Spannungswerte vom Arduino erfasst und zum PC übertragen werden.

Als Messkanal kommt der Analog-Eingang ADC0 zum Einsatz. Die dort anliegenden Spannungswerte sollen erfasst und in digitale Messwerte gewandelt werden. Anschließend werden die Werte an einen PC übertragen und am seriellen Monitor dargestellt.

Das Programm dazu kann so aussehen:

```
// ADC

int ADC0=0;
int value;

void setup()
{ Serial.begin(9600);
}

void loop()
{ value=analogRead(ADC0);
  Serial.print("ADC0 = ");
  Serial.println(value);
  delay(5000);
}
```

Die zugehörige Schaltung ist in Bild 2 zu sehen. Das Potentiometer sollte einen Bahnwiderstand von ca. 100 kΩ besitzen. Nach dem Laden des Programms und dem Starten des seriellen Monitors werden die Messwerte sichtbar (Bild 3). Da direkt der ADC-Wert ausgegeben wird, liegen die angezeigten Zahlenwerte im Bereich von 0 bis 2^{10}-1 = 1023.

Temperaturmessung

Temperaturfühler zählen zu den wichtigsten Sensoren in der elektronischen Messtechnik. Sie überwachen die Temperaturen in den Prozessoren moderner PCs genauso wie in Automotoren, Großrechenanlagen oder in den Prozess-Steuerungen der chemischen Industrie.

Einfache Temperatursensoren bestehen meist aus mit Bindemitteln versetzten, gepressten und gesinterten Metalloxiden. Der Nominalwiderstand eines solchen Messelements lässt sich durch das Mischverhältnis verschiedener Materialien in weiten Bereichen einstellen.

So entstehen sogenannte NTCs, also Temperatursensoren mit negativem Temperaturkoeffizienten. Ein Standardtyp dieses Bauelements

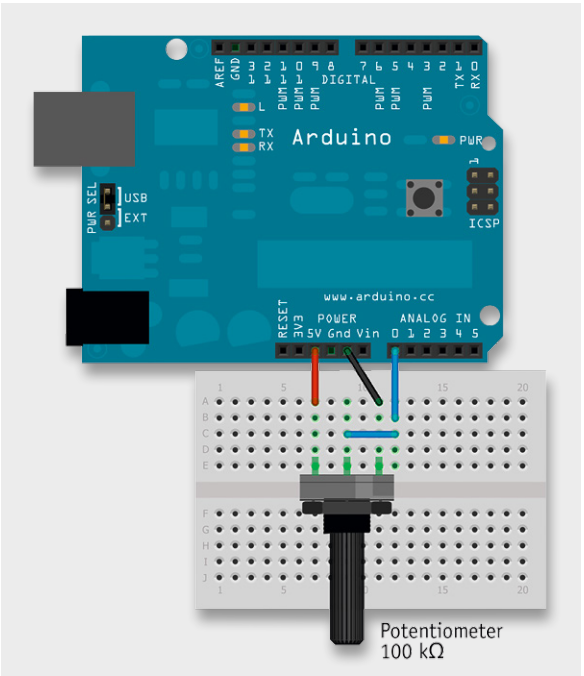

Bild 2: Aufbau für die Erfassung analoger Messwerte

weist beispielsweise einen Widerstandswert von 4,7 kΩ bei einer Temperatur von 25 °C auf.

Elektronisches Thermometer mit Übertemperaturalarm

Ein solcher NTC eignet sich hervorragend zum Aufbau eines elektronischen Computer-Thermometers. Die Temperaturmesswerte sollen dabei über den Mikrocontroller erfasst und am PC dargestellt werden.

Neben der Darstellung der aktuellen Temperatur ermöglicht ein solches Thermometer auch die zeitaufgelöste Messung von Temperaturen. Typische Anwendungen dafür sind die Temperaturüberwachung von Leistungstransistoren, Mikroprozessoren oder ganzen klimatisierten Serverräumen.

Zusätzlich kann ein Übertemperaturalarm integriert werden, über welchen im Ernstfall ein zusätzliches Kühlaggregat aktiviert oder aber sogar die Leistung einer kompletten Serverfarm heruntergefahren werden könnte. Die Umrechnung des ADC-Werts in eine Temperatur erfolgt über 2 Konstanten:

cal = 6.8
offset = 296

Die beiden Werte ergeben sich aus dem Offset sowie aus der Steilheit der Kennlinie des NTCs. Falls die Temperatur nicht korrekt angezeigt wird, können die beiden Konstanten an das verwendete Sensorexemplar angepasst werden.

Prinzipiell ist die Kalibrationsfunktion eines NTCs nichtlinear. Für einen Temperaturbereich von ca. 0 bis etwa 50 °C können die Abweichungen aber weitgehend vernachlässigt werden. Für höchste Ansprüche bezüglich der Messpräzision kann selbstverständlich auch eine nichtlineare Funktion zur Berechnung der Temperatur implementiert werden. Der nachfolgende Sketch zeigt, wie ein Programm zur Temperaturmessung inklusive eines Übertemperaturalarms aussehen könnte.

```
// Temperature sensor
// for serial interface
// using NTC temperature sensor
// incl. overtemp indication

int ADC0=0;          // select ADC channel
int ADC_value;
float temp;
float cal=6.8;       // calibration factor for NTC 4k7
float offset=296;    // offset constant for NTC 4k7

float overtemp = 30;
int overtempLED = 13;

void setup()
{ Serial.begin(9600);
  pinMode(overtempLED,OUTPUT);
}

void loop()
{ ADC_value=analogRead(ADC0);
  temp=((ADC_value-offset)/cal);
  Serial.print("Temp = ");
  Serial.print(temp);
  Serial.println(" C ");

  if(temp>overtemp)digitalWrite(overtempLED,HIGH);
  if(temp<overtemp)digitalWrite(overtempLED,LOW);

  delay(1000);
}
```

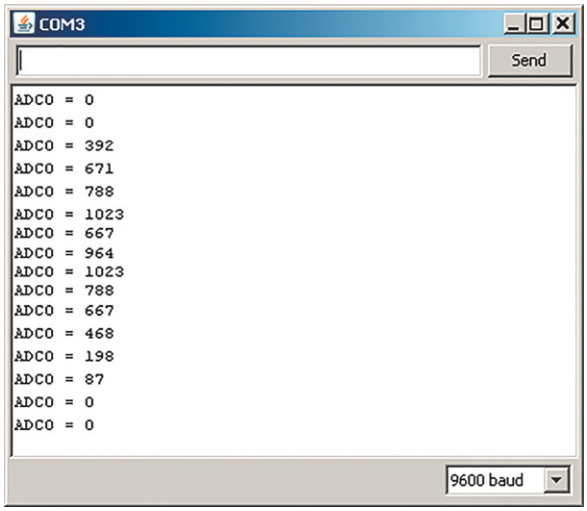

Bild 3: Darstellung der Messwerte im seriellen Monitor

Den Schaltungsaufbau dazu zeigt Bild 4. Der Temperatursensor wird hier in einer einfachen Spannungsteiler-Schaltung betrieben.

Der 4,7-kΩ-Festwiderstand sorgt dafür, dass die Eingangsspannung für den ADC bei Raumtemperatur bei ca. der halben Betriebsspannung, d. h. bei etwa 2,5 V, liegt. Damit lässt sich der Temperaturbereich von 0 bis 50 °C gut abdecken.

Nach dem Starten des Programms und des seriellen Monitors werden die jeweils aktuellen Temperaturwerte im Abstand von ca. 1 s auf dem Bildschirm angezeigt. Berührt man den Sensor mit den Fingerspitzen, wird der damit einhergehende Temperaturanstieg unverzüglich sichtbar. Überschreitet die Temperatur den in der Zeile

float overtemp = 30;

festgelegten Schwellwert von 30 °C, so wird die LED als Warnanzeige aktiviert.

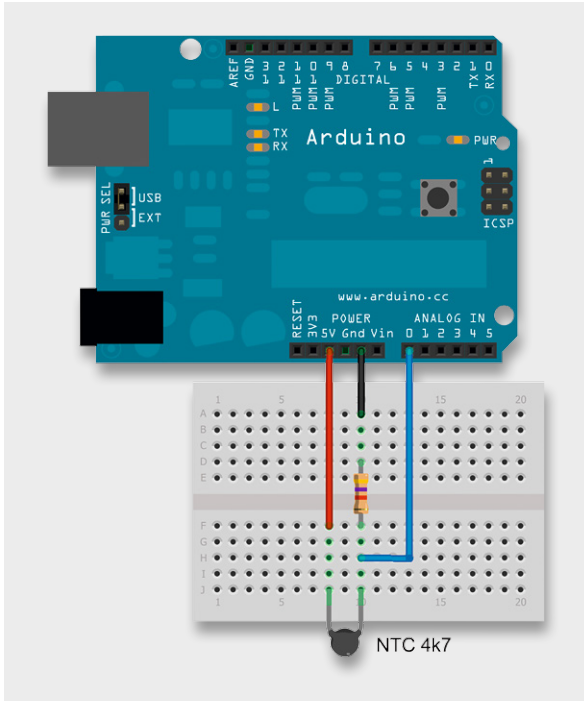

Bild 4: Aufbau für ein elektronisches Computer-Thermometer mit Übertemperaturalarm

Bild 5 zeigt ein Temperaturüberwachungsgerät für eine kleine Klimakammer. Hier wurden 5 der 6 vorhandenen Analogkanäle mit Temperatursensoren ausgestattet. Der jeweils aktive Kanal wird über eine LED angezeigt. Als Display für die Temperaturwerte kam hier eine 4-stellige 7-Segment-Anzeige zum Einsatz. Weitere Details zur Ansteuerung einer solchen Anzeige werden in einem späteren Beitrag in dieser Artikelserie erläutert.

Der Phototransistor als optischer Sensor

Ein weiterer sehr interessanter Sensor ist der Phototransistor. Dieser gestattet die Erfassung von Helligkeitswerten. Typische Anwendungen dieses Bauelements sind Lichtschranken, Belichtungsmesser für Fotokameras oder Empfänger für Infrarotfernbedienungen.

Elektronisches Luxmeter

Mit dem Phototransistor kann die Umgebungshelligkeit bestimmt und der aktuelle Helligkeitswert auf einem PC-Bildschirm dargestellt werden. Der Sensortyp BPW40 eignet sich bestens für diese Aufgabe.

Anstelle einer numerischen Anzeige kann man für die Helligkeiten auch eine grafische Darstellung wählen. Im Beispiel wird eine sogenannte ASCII-Grafik verwendet. Das bedeutet, dass eine Zeile aus X-Zeichen ihre Länge proportional zur aktuellen Umgebungshelligkeit verändert.

Das Prinzip dieser einfachen Grafikausgabe wurde bereits im Artikel „Programmierpraxis: Befehle, Variablen und Funktionen" erläutert, sodass das Arduino-Programm dazu keine Verständnisschwierigkeiten mehr bereitet.

```
// Luxmeter
// Rv = 10 k

#define AlarmLED 13

int ADC_value;
int ADC_0=0;
int threshold = 300;

void setup()
{ Serial.begin(9600);
  pinMode(AlarmLED, OUTPUT);
}

void loop()
{ ADC_value=analogRead(ADC_0);
  if (ADC_value>0)
  {  for(int i=0; i<=(ADC_value/10); i++)
       Serial.print("X");
       Serial.println("");
  }
  if(ADC_value < threshold)
    { digitalWrite(AlarmLED,HIGH);
    }
  else digitalWrite(AlarmLED,LOW);
  delay(300);
}
```

Bild 5: Im Eigenbau erstelltes µC-basiertes Computer-Thermometer mit Übertemperaturalarm und 5 Alarmausgängen zur Überwachung von Serverräumen

Den Hardwareaufbau dazu zeigt Bild 6. Im Gegensatz zum NTC weist der Phototransistor eine Polung auf. Falls sich die Messwerte also nicht wie erwartet mit der Umgebungshelligkeit verändern, sollte man den Lichtsensor umpolen.

Bild 7 zeigt, wie sich die Länge des Anzeigebalkens verändert, wenn der Sensor mit einer Lampe beleuchtet wird. Unterschreitet die Umgebungshelligkeit einen festgelegten Schwellwert, wird die LED 13 als „Zusatzbeleuchtung" eingeschaltet. Natürlich könnte man auf diese Weise auch eine leuchtstarke Power-LED als echte Zusatzlichtquelle ansteuern.

Grafische Darstellung von Messwerten auf dem PC-Bildschirm

Wenn man die Messwerte von Sensoren nicht nur über den seriellen Monitor in numerischer Form oder als einfache ASCII-Grafiken darstellen will, sondern echte grafische Ausgaben erwünscht sind, so kann man das Programm „Processing" einsetzen.

Dieses äußerst nützliche und universell einsetzbare Programm kann unter [1] kostenlos aus dem Internet geladen werden. Dass das Programm der Arduino-IDE stark ähnelt, ist kein Zufall. Die Arduino-Programmieroberfläche ist sehr eng an das am Massachusetts Institute for Technology (MIT) entwickelte Processing angelehnt. Dies hat den Vorteil, dass sich routinierte Arduino-Anwender schnell zurechtfinden. Zudem werden ähnlich wie bei der Arduino-IDE auch bei Processing viele interessante Beispielanwendungen mitgeliefert. Alleine das Austesten dieser Beispielprogramme ist schon eine sehr interessante und lehrreiche Beschäftigung und erleichtert den Einstieg in dieses Programm ganz erheblich.

Durch vorgefertigte Bibliotheken wird die Einbindung des Arduinos in Processing wesentlich vereinfacht. So können über die virtuelle serielle Schnittstelle problemlos Messdaten übertragen und dann sehr ansprechend auf einem PC-Bildschirm dargestellt werden. Bild 8 zeigt beispielsweise die simultane Darstellung aller 6 Arduino-Analogkanäle in Form einer Balkengrafik.

Aber auch die zeitliche Veränderung von Messwerten kann grafisch aufbereitet werden. So zeigt Bild 9 einen sogenannten Temperatur-Logger. Damit können Temperaturverläufe über Stunden oder Tage hinweg aufgezeichnet und dargestellt werden. Der Überwachung und Optimierung der hauseigenen Heizungsanlage steht so nichts mehr im Wege.

Ausblick

In diesem Artikel wurde gezeigt, wie man verschiedene Sensorwerte über die Analogkanäle des Arduinos auslesen und verarbeiten kann. Die Anwendungen dieses weiten Teilbereichs der Mikrocontrollertechnik sind nahezu unbeschränkt. Von einfachen Temperaturüberwachungen über die Steuerung von Solarzellenanlagen bis hin zu komplexen Aufgaben in der Mess- und Regelungstechnik – überall werden Sensoren benötigt und eingesetzt.

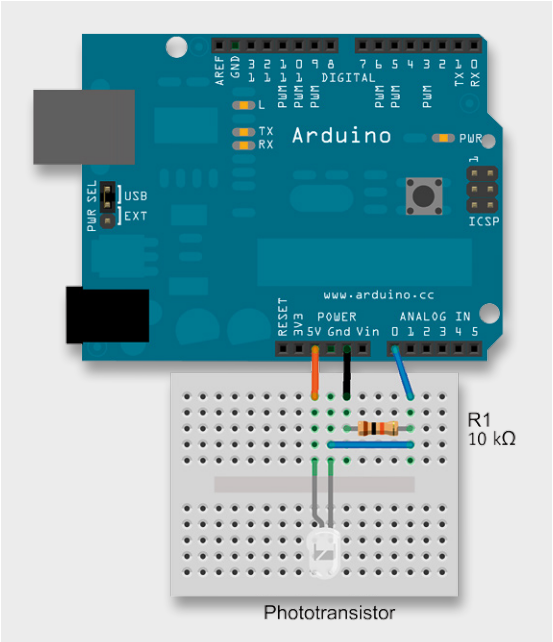

Bild 6: Aufbau für das elektronische Luxmeter

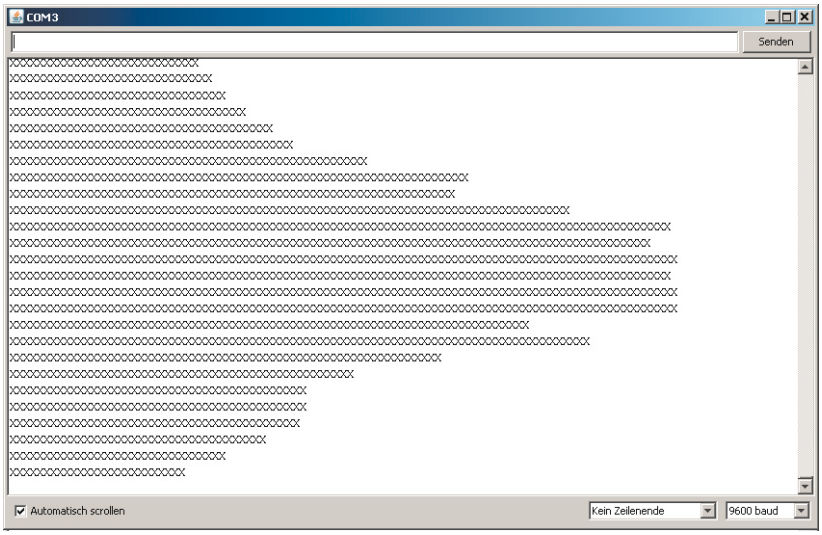

Bild 7: Ausgabe von Helligkeitswerten als ASCII-Grafik

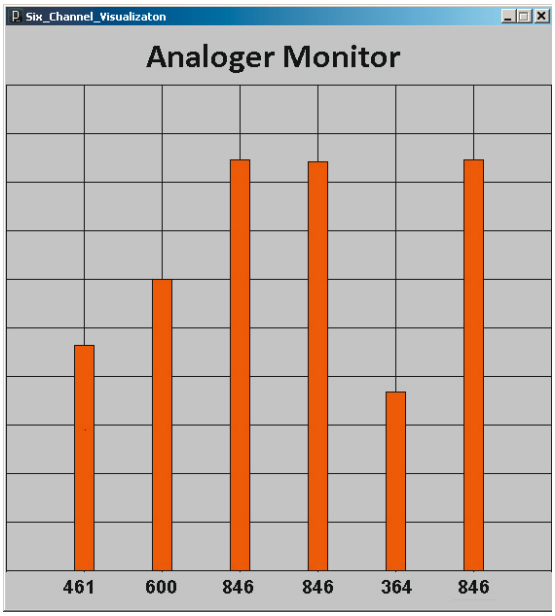

Bild 8: Darstellung von Analogkanälen als Balkengrafik

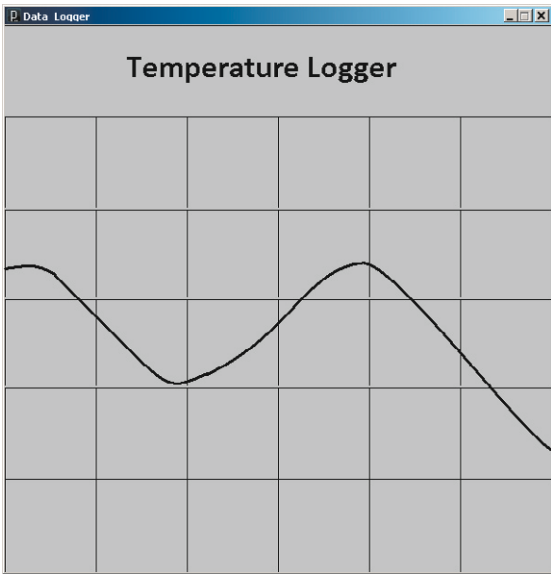

Bild 9: Temperatur-Logger mit Processing

In neuerer Zeit hat sich für diese Anwendungen auch der Begriff „Physical Computing" etabliert. Hierunter versteht man die Erfassung und physikalische Beeinflussung bzw. Steuerung der Umwelt mittels elektronischer Sensoren und Aktuatoren. Der erste Teil, die Sensorik, wurde bereits in diesem Artikel behandelt. Die Ansteuerung von Aktuatoren wie Schrittmotoren oder Servos folgt in einem späteren Beitrag.

Ein weiterer wichtiger Teilbereich des Physical Computing ist die Übertragung von Daten über Schnittstellen und Interfaces wie USB, RS232 und I²C. Diese Themen werden im nächsten Artikel dieser Serie behandelt.

Empfohlenes Material

Franzis-Lernpakete wie etwa „Elektronik mit ICs" enthalten Materialien wie ein lötfreies Steckbrett, Widerstände und LEDs etc., die für den Aufbau von Schaltungen mit dem Arduino gut geeignet sind. ELV

Weitere Infos:

[1] www.processing.org

G. Spanner: Arduino – Schaltungsprojekte für Profis, Elektor-Verlag, 2012, Best.-Nr.: J5-10 94 45

G. Spanner: AVR-Mikrocontroller in C programmieren, Franzis-Verlag, 2010, Best.-Nr.: J5-09 73 52

Preisstellung August 2014 – aktuelle Preise im Web-Shop

Empfohlene Produkte/Bauteile:	Best.-Nr.	Preis
Arduino-Uno-Platine R3	J5-10 29 70	€ 27,95
Mikrocontroller-Onlinekurs	J5-10 20 44	€ 99,–
Lernpaket Mikrocontroller programmieren	J5-10 68 46	€ 129,–

Alle Arduino-Produkte wie Mikrocontroller-Platinen, Shields, Fachbücher und Zubehör finden Sie unter: www.arduino.elv.de