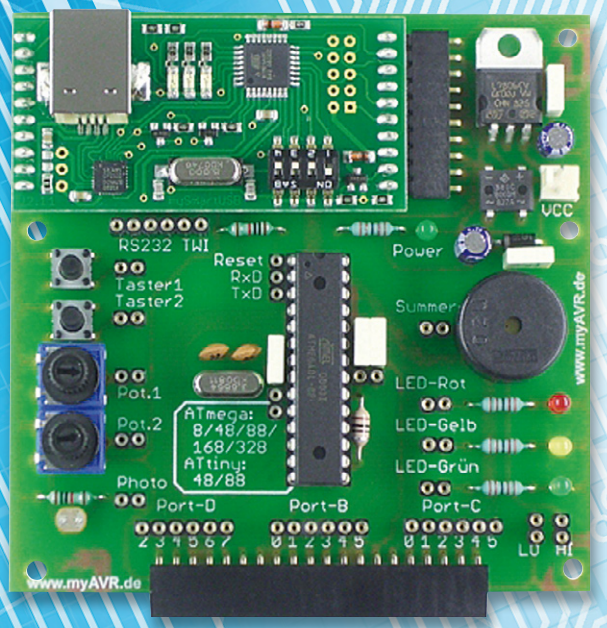
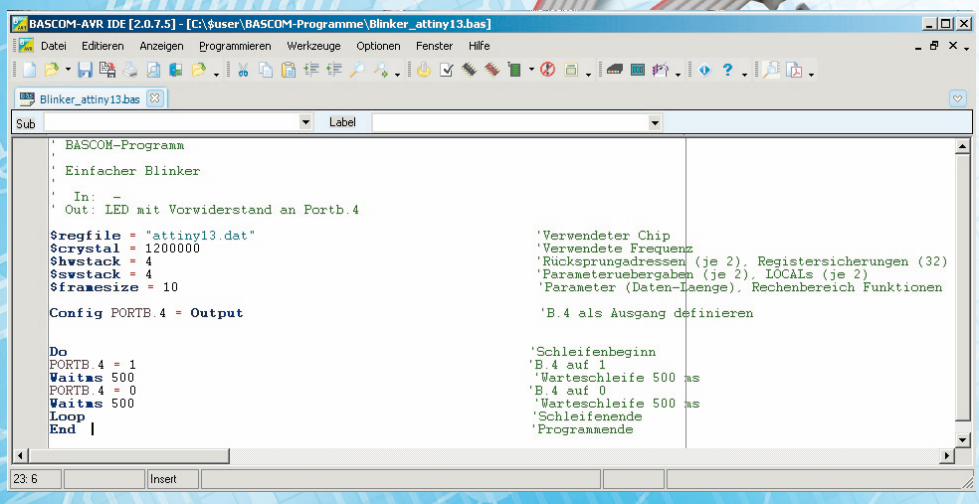
Mikrocontroller-Einstieg

Teil 2: Ein- und Ausgaben





mit BASCOM-AVR

Mikrocontroller können vor allem eines sehr gut: sehr schnell Befehle ausführen, die mit einer Programmiersprache wie beispielsweise BASCOM-AVR genau beschrieben – programmiert – wurden. Die schnellsten Rechnungen bringen allerdings nichts, wenn nicht Verbindungen zur Außenwelt existieren, und deshalb werden in diesem zweiten Teil der ELV-Artikelserie zum Thema Mikrocontroller-Einstieg mit BASCOM-AVR vielfältige Möglichkeiten beschrieben, wie Eingaben und Ausgaben (Input/Output, I/O) in Bezug auf einen Mikrocontroller verwirklicht werden können.

Eingabe – Verarbeitung – Ausgabe

Grundprinzip beim Einsatz von Mikrocontrollern ist, dass eine oder mehrere Umgebungsgrößen erfasst werden, per vordefiniertem Algorithmus aus diesen Größen etwas berechnet wird und das Ergebnis dann in irgendeiner Weise ausgegeben wird. Man spricht auch von den drei Komponenten Eingabe, Verarbeitung und Ausgabe (Bild 1).

Im einfachsten Fall könnte ein System aus einem Mikrocontroller (für die Verarbeitung) und einer Leuchtdiode (als Ausgabemedium) bestehen. Ohne Eingabekomponenten. Man kann damit schon eine blinkende oder blitzende LED oder eine optische Morsezeichenausgabe oder Ähnliches realisieren. Mit drei LEDs könnte man eine Ampel für die Modellbahnanlage erstellen. Vielfältiger werden die Möglichkeiten, wenn Eingaben dazukommen. So kann man zum Beispiel im Programm auf Tastendruck oder Schalterzustände (offen/geschlossen) oder auf Temperatur- oder Lichtsensoren reagieren und in Abhängigkeit der Eingabewerte programmgesteuert Ausgaben erzeugen. In diesem Artikel liegt der Schwerpunkt auf den in Bild 2 dargestellten Ein- und Ausgabemöglichkeiten.

Darüber hinaus kommen als Eingabe in Frage: serielle Signale, GPS-Sensor, Servosignale, I²C- oder 1-Wire-Sensoren, DFC77-Zeitmodul, Drucksensor, Entfernungssensor, Beschleunigungssensor, Kompassmodul, Gassensor, Fingerabdruck-Scanner, RFID-Leser, Funkmodule usw.

Weitere Beispiele für Ausgaben sind: Servos, serielle Devices, I²C-Aktoren, Schrittmotoren, Funkmodule, Drucker, Grafikdisplays usw.

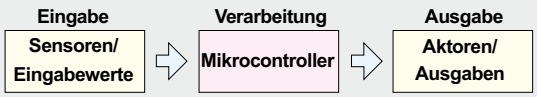


Bild 1: Eingabe – Verarbeitung – Ausgabe

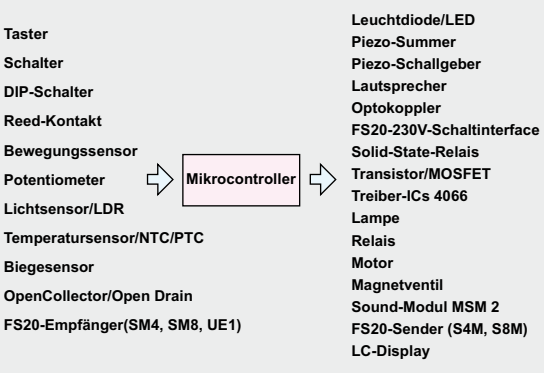


Bild 2: Einfache Eingaben und Ausgaben

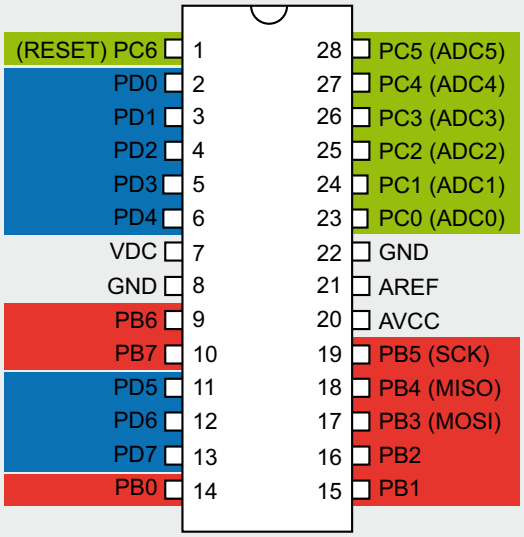


Bild 3: Ein-/Ausgabeports ATmega88

Portaufbau

Ein Mikrocontroller verfügt über eine je nach Typ unterschiedliche Anzahl von Pins als Verbindung zur Außenwelt. Die Ein-/Ausgabepins sind zu sogenannten Ports zu je maximal 8 Pins zusammengefasst (Bild 3). Ein ATmega88 hat (ebenso wie ein ATmega8) einen Port B, Port C und Port D, wobei die Pins B.0, B.1, B.2, B.3, B.4, B.5, B.6, B.7, C.0, C.1, C.2, C.3, C.4, C.5, C.6 sowie D.0, D.1, D.2, D.3, D.4, D.5, D.6 und D.7 herausgeführt sind.

Jeder dieser Portpins lässt sich als Eingang oder als Ausgang definieren und nutzen und ein Port kann auch als ganzer Port von BASCOM aus angesprochen werden.

Die Darstellung in Bild 4 dient einer schematischen Darstellung. Tatsächlich sind Eingabepins und Ausgabepins eines Ports jeweils an einem physikalischen Pin des ICs herausgeführt. Für jede Stelle eines Ports gibt es das in Bild 5 dargestellte Verschaltungsprinzip. Zu jedem Port gehören drei Register: DDR (= Data Direction Register), Port (Ausgaberegister) und Pin (Eingaberegister).

Durch Setzen oder Nicht-Setzen der Bits des DDR-Registers wird bestimmt, ob ein Portpin als Eingabe- oder Ausgabepin verwendet werden soll. Ist das entsprechende Bit des DDR-Registers logisch 0, dann funktioniert der Portpin als Eingang. Ist das Bit des DDR-Registers logisch 1, dann wird der Zustand des entsprechenden Bits des Port-Registers nach außen ausgegeben.

Ist DDRx.n logisch 0 (also Eingang), dann kann man durch Setzen des entsprechenden Portx.n einen internen Pull-up-Widerstand von 20 bis 50 kΩ zuschalten, damit der Eingang bei offenem Pin einen definierten 1-Zustand angibt.

In BASCOM definiert man beispielsweise alle Bits des Port B als Ausgänge, indem man im Programm schreibt:

DDRB = &b11111111 ' Alle Stellen von Port B als Ausgang

Die Schreibweise &b zeigt an, dass die Zahl als Binärzahl interpretiert werden soll. Die einzelnen Bits werden von rechts (0) nach links (7) gezählt.

Mit DDRB = &b11111010 würden B.0 und B.2 als Eingang und die restlichen Bits als Ausgang geschaltet.

Auch ein einzelnes Bit eines Ports lässt sich als Ausgang definieren – zum Beispiel mit DDRB.3 = 1.

BASCOM lässt auch diese Schreibweise zu:

CONFIG PORTB.3 = Output 'Die vierte Stelle von rechts als Ausgang definiert (Zählweise von rechts nach links: 0, 1, 2, 3).

Neben der Funktion als digitaler Ein- oder Ausgang sind den Pins noch weitere Funktionalitäten zugeordnet, die per BASCOM-Definitionsbefehl aktiviert werden können. Am IC-Pin 24 ist zum Beispiel sowohl das Bit 1 des digitalen Ein-/Ausgaberegisters Port C als auch ein analoger Eingangskanal (ADC1) verschaltet. Die IC-Pins 1, 17, 18 und 19 werden für die In-System-Programmierung des Mikrocontrollers verwendet, können aber auch als digitale Ein-/Ausgabepins benutzt werden.

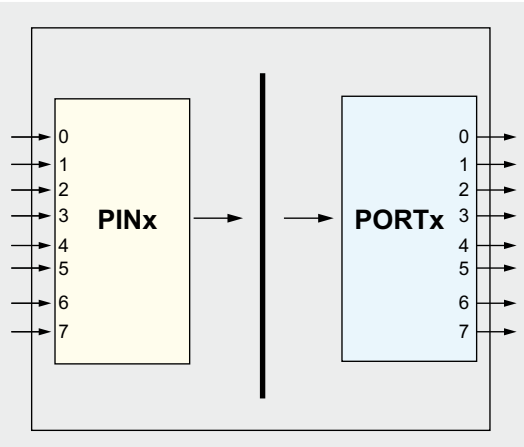


Bild 4: PINx, PORTx

Digitale Ausgabe

Eine Leuchtdiode kann (mit Vorwiderstand – siehe „Elektronikwissen LED-Vorwiderstand“) direkt an einen Portpin angeschlossen werden (Bild 6).

Ein einfaches Blinkprogramm für eine Leuchtdiode sieht in BASCOM folgendermaßen aus:

```
' BASCOM-Programm
' LED blinkt
'   In :  -
' Out : LED mit Vorwiderstand an C.0

$regfile = "M88def.dat"          'Verwendeter Chip
$crystal= 1000000                'Verwendete Frequenz
$hwstack= 40                     'Rücksprungadressen (je 2),
                                  Registersicherungen (32)
$swstack= 40                     'Parameteruebergaben (je 2), LOCALs (je 2)
$framesize = 60                  'Parameter (Daten-Laenge),
                                  Rechenbereich Funktionen

Config Portc.0 = Output          'Portpin C.0 als Ausgang

Do                               'Schleifenbeginn
  Portc.0 = 1                    'Das rechteste Bit von Port C
                                  auf logisch 1 setzen
  Wait 1                         'Eine Sekunde warten
  Portc.0 = 0                    'Das rechteste Bit von Port C
                                  auf logisch 0 setzen
  Wait 1
Loop                             'Schleifenende
End                              'Programmende
```

Erläuterungen:

Durch ein Hochkomma werden in BASCOM Kommentare eingeleitet. Sie dienen dem menschlichen Leser zum besseren Verständnis und sollten reichlich eingesetzt werden.

Es wird ein ATmega88 mit einer Taktfrequenz von 1 MHz verwendet und die Stackwerte werden großzügig deklariert, weil der ATmega88 mit 1 KByte SRAM reichlich Platz für Stack und Variablen hat.

CONFIG PORTC.0 = Output setzt das rechteste Bit (Bit 0) des Registers DDRC auf 1 und definiert dadurch diesen Portpin als Ausgang (identisch mit DDRC.0 = 1).

Die Schlüsselwörter DO und LOOP umklammern eine sogenannte Endlosschleife: Die Befehle zwischen diesen beiden Wörtern werden immer wieder (endlos) ausgeführt.

Durch PORTC.0 = 1 wird das rechteste (niedrigste) Bit des PORT-Registers C auf logisch 1 gesetzt. Durch PORTC.0 = 0 wird es auf logisch 0 gesetzt. (Wenn die Mikrocontrollerschaltung mit 5 V betrieben wird, dann entspricht eine logische 1 in etwa 5 V am Ausgangspin und eine logische 0 null Volt.)

Die Anweisung WAIT 1 bewirkt eine Wartezeit (Englisch wait = warten) von etwa einer Sekunde.

Also wird der Ausgang C.0 (nachdem der Portpin C.0 einmalig als Ausgang definiert wurde) zunächst auf 1 gesetzt, dann wird eine Sekunde gewartet, dann wird der Ausgang auf 0 gesetzt, es wird wieder eine Sekunde gewartet, und dieser Ablauf wiederholt sich unendlich lange.

Das END-Statement wird geschrieben, damit der Programmablauf nicht in sinnlose Speicherbereiche läuft, wenn der Programmierer möglicherweise die Endlosschleife nicht sauber programmiert.

Wenn nun (nach Kompilieren und Brennen) an den Portpin C.0 eine Leuchtdiode (= LED) angeschlossen wird, dann wird diese LED im entsprechenden Rhythmus blinken.

Werden die Wartezeiten verändert – zum Beispiel auf WAIT 2 –, dann blinkt eine angeschlossene Leuchtdiode langsamer.

Schreibt man für die Wartezeiten jeweils WAITMS 500 (= Warte 500 Millisekunden, also 500 Tausendstelsekunden, also eine halbe Sekunde), dann blinkt eine angeschlossene Leuchtdiode schneller.

Statt einer Leuchtdiode kann man auch einen Piezo-Schallgeber oder einen kleinen Lautsprecher mit davorgeschaltetem Widerstand anschließen. Statt des Blinkens der LED erhält man dann ein Knacken des Schallgebers. Wird die Wartezeit weiter verkürzt (z. B. WAITMS 1), geht das Knacken des Schallgebers in einen Ton über.

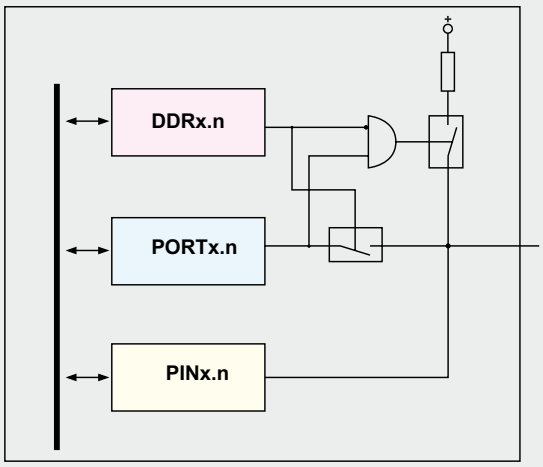


Bild 5: Prinzip eines einzelnen Portpins

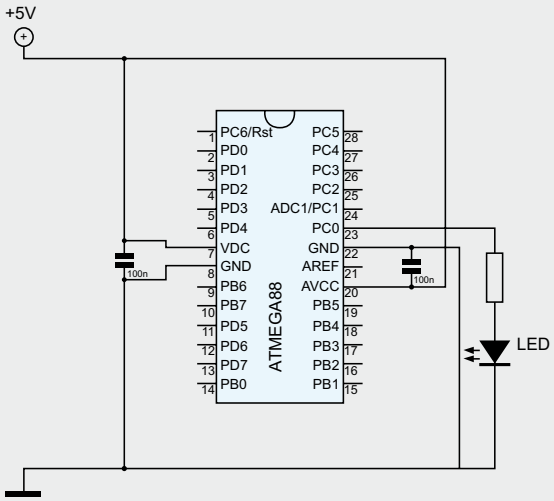


Bild 6: Basisschaltung mit LED

Ausgangsbeschaltung eines Pins

Ein einzelner Portpin eines ATmega88 verträgt einen Strom von maximal 40 mA. Insgesamt (alle Portpins zusammengerechnet) dürfen 200 mA nicht überschritten werden. Diese Werte stehen im Datenblatt des Mikrocontrollers (Absolute Maximum Ratings) und müssen unbedingt beachtet werden.

Möchte man mit einem Mikrocontroller 230-V-Geräte schalten, kann man statt der Leuchtdiode einfach ein ELV-230-V-Schaltinterface SI 230-2 (Best.-Nr. JU-09 20 35) an den Portpin anschließen. Im 230-V-Schaltinterface von ELV steckt in der Eingangsstufe ein Optokoppler CNY 17, der eingangsseitig nichts anderes als eine Leuchtdiode darstellt. Der berechnete und gemessene Strom bei diesem Schaltinterface ist ca. 10 mA und dadurch dem Portpin zumutbar. So kann man sehr einfach 230-V-Verbraucher bis 16 A mit einem kleinen Mikrocontroller ansteuern.

Auch kann man den Ausgangspin des Mikrocontrollers beispielsweise direkt mit dem F20-4/8-Kanal-Sender S8M verbinden. Wird dann am Ausgang länger als 100 Millisekunden ein Low-Signal ausgegeben, sendet der FS20-Sender den entsprechenden Funkbefehl an einen Empfänger aus dem FS20-Programm. Die Versorgungsspannung des S8M kann dieselbe sein wie die des Mikrocontrollers (5 V).

Ebenso lässt sich der FS20-2/4-Kanal-Sender S4M an den AVR-Mikrocontroller anschließen. Dieser Sen-

Elektronikwissen

LED-Vorwiderstand

Eine Leuchtdiode darf nie ohne Vorwiderstand betrieben werden. Um die Größe des Vorwiderstandes zu ermitteln, benötigt man die Betriebsspannung (zum Beispiel 5 V), die Durchlass-Spannung der Leuchtdiode (aus dem Datenblatt oder dem Katalog – zum Beispiel 1,5 V) und den Durchlassstrom I_F (aus dem Datenblatt oder dem Katalog – zum Beispiel 20 mA).

Der Vorwiderstand lässt sich dann wie folgt berechnen: $R_V = \frac{U_B - U_F}{I_F}$

Beispiel: $R_V = \frac{5\,V - 1{,}5\,V}{20\,mA} = 175\,\Omega$

Man nimmt dann den nächstgrößeren Norm-Widerstandswert – hier 220 Ω.

der ist ebenfalls für den Betrieb an 5 V geeignet. Da der eingebaute Mikrocontroller mit 3 V betrieben wird, muss eine Diode (1N4148, Best.-Nr. JU-00 23 04) in Sperr-Richtung zwischen den Portpin des AVR-Mikrocontrollers und den Eingangsanschluss des S4M geschaltet werden, um den Mikrocontroller des S4M-Moduls gegen die für ihn zu hohe 5-V-Spannung zu schützen.

Beide dieser FS20-Sender eignen sich hervorragend für die Integration in eigene Mikrocontroller-Anwendungen (Einschlaflicht, Belichtungstimer, Garagenlicht usw).

Wenn mehr als nur eine Handvoll Leuchtdioden angesteuert werden sollen – zum Beispiel ein Relais oder ein Motor oder sehr viele oder stärkere LEDs –, muss man an den Mikrocontroller-Ausgang eine Treiberschaltung anschließen. Standard ist ein NPN-Transistor, der an den Ausgangspin des Mikrocontrollers angeschlossen wird und der seinerseits – je nach Transistor – hohe Spannungen und Ströme schalten kann. Der Standard-Transistor BC547 (Best.-Nr. JU-00 53 37) kann 45 V und 100 mA schalten. Zum Schalten höherer Ströme kann man andere Transistoren einsetzen oder man schaltet die Last über ein Relais, welches seinerseits durch den Transistor angesteuert wird. Ebenso kann ein TTL-Level-MOSFET oder Logic-Level-MOSFET benutzt werden; in dem Fall ist kein Basiswiderstand erforderlich.

Es gibt auch integrierte Treiber-ICs wie zum Beispiel den ULN2003, ULN2803 (schalten beide gegen Masse) oder UDN2981 (schaltet gegen Plus). Diese Treiber-ICs stellen jeweils 7 bzw. 8 Treiberschaltungen zur Verfügung, die jeweils 500 mA schalten können. Sogar die Freilaufdioden/Schutzdioden für induktive Lasten wie Relais und Motoren sind bereits in die ICs integriert.

In manchen Situationen möchte man mit einem Mikrocontroller Wechselspannungen schalten, aber man möchte kein Relais einsetzen, weil entweder höhere Schaltfrequenzen erforderlich sind oder weil man geräuschlos schalten möchte. Es eignet sich dann zum Beispiel ein 4fach bidirektionaler CMOS-Schalter 4066 (ELV-Best.-Nr. JU-00 55 35), welcher 25 mA Wechselstrom schalten kann. Damit lassen sich zum Beispiel Audiosignale schalten, oder – was sehr effektvoll ist – man kann mit einem 4066 z. B. das ELV MP3-Sound-Modul MSM 2 (Best.-Nr. JU-09 28 53) schalten!

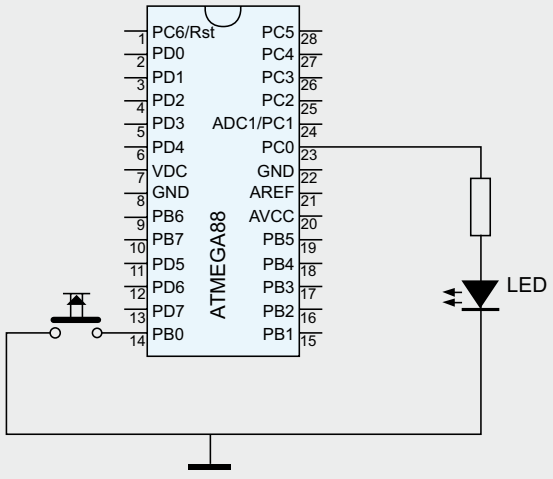


Bild 7: Anschluss eines Tasters und einer LED

Digitale Eingabe

Wie eingangs beschrieben, sollen in der Regel Eingaben wie zum Beispiel Taster (Bild 7), Schalter, Alarm-Drähte, Reed-Kontakte oder Ähnliches erfasst und deren Zustände verarbeitet werden.

Aus Gründen der Übersichtlichkeit wird im Schaltplan auf die wiederholte Darstellung der Anschlüsse der Betriebsspannung verzichtet. Grundsätzlich werden alle GND-Pins mit 0 V verbunden. Alle VCC/AVCC werden mit +5 V verbunden. Außerdem sind Abblock-Kondensatoren von 100 nF sehr nahe am IC vorzusehen.

Das folgende BASCOM-Programm lässt eine LED blitzen und erfasst einen Tastendruck.

```
' BASCOM-Programm
'
' LED blitzt bzw. wenn Taste gedrückt blinkt.
'
' In:  Taster an B.0
' Out: LED an C.0
'
$regfile = "M88def.dat"              'Verwendeter Chip
$crystal= 1000000                    'Verwendete Frequenz
$hwstack= 40                         'Rücksprungadressen (je 2),
                                      Registersicherungen (32)
$swstack= 40                         'Parameteruebergaben (je 2),
                                      LOCALs (je 2)
$framesize = 60                      'Parameter (Daten-Laenge),
                                      Rechenbereich Funktionen

Config Portb.0 = Input               'Port-Pin B.0 als Eingang
Portb.0 = 1                          'Pullup-Widerstand intern
Config Portc.0 = Output              'Portpin C.0 als Ausgang

Do                                   'Schleifenbeginn
  Portc.0 = 1                        'Das rechteste Bit von Port C
                                      auf logisch 1 setzen
  Waitms 20

  If Pinb.0 = 0 Then Waitms 980      'Wenn Taste gedrückt:
                                      Zusätzlich 980 Millisekunden

  Portc.0 = 0                        'Das rechteste Bit von Port C
                                      auf logisch 0 setzen
  Wait 1
Loop                                 'Schleifenende
End                                  'Programmende
```

Erläuterungen:

Durch Config Portb.0 wird der Portpin B.0 als Eingang definiert (das entspricht DDRB.0 = 0). Mit Portb.0 = 1 wird für den als Eingang definierten Portpin der interne Pull-up-Widerstand geschaltet. Dadurch wird erreicht, dass bei offenem (unbeschaltetem) Port-Pin ein definiertes High-Signal anliegt. Der Portpin C.0 wird als Ausgang definiert. In der Endlosschleife (Do – Loop) wird die LED für 20 Millisekunden eingeschaltet und dann für eine Sekunde ausgeschaltet. Wenn (IF) ein an Portpin B.0 angeschlossener Taster gedrückt wird und dadurch logisch 0 am Pin anliegt, dann (THEN) wird zusätzliche 980 Millisekunden gewartet. Ganz wichtig: Abfrage des Portpins mit IF **PIN**x.y! Ausgabe am Portpin mit **PORT**x.y. Bei offenem Taster blitzt die LED also (20 ms an – 1 Sekunde aus) und bei geschlossenem Taster blinkt die LED (20 ms + 980 ms an – 1 Sekunde aus).

In vielen Situationen müssen Taster und andere mechanische Kontakte „entprellt" werden, weil die mechanischen Kontaktzungen beim Ein- und Ausschalten nicht sofort sauber den Zustand ändern, sondern ein paarmal zwischen den Zuständen hin- und herschalten, bevor der gewünschte Zustand endlich erreicht ist. Das kann zu unerwünschten Mehrfachereignissen führen. BASCOM stellt den Befehl DEBOUNCE zur Verfügung, um das Prellen (welches sich im Millisekundenbereich befindet) wirkungsvoll softwaretechnisch auszugleichen.

Analoge Eingabe

Neben den digitalen Zuständen von Schaltkontakten oder Eingangsmodulen sollen sehr oft analoge Messgrößen erfasst werden, die zum Beispiel von Potentiometern bzw. Spannungsteilern mit Temperatur-, Licht- oder anderen Sensoren kommen (Bild 8 und Bild 9).

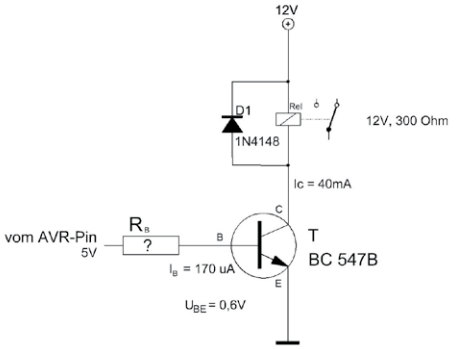
Mithilfe eines kurzen BASCOM-Programms lässt sich die Stellung eines Potentiometers wie auch ein Helligkeitswert oder eine Temperatur ermitteln.

```
' BASCOM-Programm
'
' LED leuchtet je nach Spannung am ADC
' Poti-Spannungsteiler an ADC1   5V -- Poti -- Gnd
 'Poti z.B. 47k
' oder
' LDR -Spanungsteiler an ADC1     5V – LDR --+-- 10k/2k -- Gnd
' PTC-Spanungsteiler an ADC1     5V – 2k --+-- KTY81 -- Gnd
'
' In:  Poti-Mittelabgriff oder LDR-Spannungsteiler an C.1
' Out: LED an C.0
'
$regfile= "M88def.dat"                  'Verwendeter Chip
$crystal= 1000000                       'Verwendete Frequenz
$hwstack= 40                            'Rücksprungadressen (je 2),
                                         Registersicherungen (32)
$swstack= 40                            'Parameteruebergaben (je 2),
                                         LOCALs (je 2)
$framesize = 60                         'Parameter (Daten-Laenge),
                                         Rechenbereich Funktionen

Config Portc.0 = Output                 'Port-Pin C.0 als Ausgang

Config Adc = Single , Prescaler = Auto ,
  Reference = Avcc                      'Analogwandler definieren
Dim Analogwandler As Word               'Variable für
                                         Analog-zu-Digital-Wandler

Do                                      'Schleifenbeginn
  Analogwandler = Getadc(1)             'AD-Wandler einlesen.
                                         Werte zwischen 0 und 1023
  If Analogwandler < 500 Then
    Portc.0 = 1                         'LED an
  Else
    Portc.0 = 0                         'LED aus
  End If
Loop                                    'Schleifenende
End                                     'Programmende
```

Erläuterungen:

Mit Config Adc wird der Analogwandler definiert. Mit dem DIM-Statement wird eine Variable vom Typ Word definiert, die im Programm den Wert des Analogwertes aufnehmen wird. Eine Variable vom Typ Byte (0 bis 255) würde hier nicht ausreichen, weil der Analogwandler Werte im Bereich 0 bis 1023 ausgibt. Durch Getadc(1) wird der am Portpin anliegende Spannungswert eingelesen und dieser Wert wird in die Variable Ana-

Elektronikwissen



Basiswiderstand

Um einen Verbraucher mit höherem Stromverbrauch – zum Beispiel ein Relais, einen Motor oder eine größere Glühlampe – an einen Ausgangspin eines Mikrocontrollers anzuschließen, schaltet man den Verbraucher üblicherweise über einen Transistor.
Die Diode D1 ist bei induktiven Verbrauchern wie Relais oder Motoren unbedingt zum Schutz der Elektronik vor Spannungsspitzen vorzusehen.
Für die Bestimmung des Basiswiderstands geht man in mehreren Schritten vor:

1.) Strom durch den Verbraucher ermitteln. Beispiel: Relais 12 V, 300 Ω ergibt:

$$I_C = \frac{U_{Relais}}{R_{Spule}}$$

$$I_C = \frac{12\,V}{300\,\Omega} = 40\,mA$$

2.) Transistor auswählen, der für den notwendigen Strom und die vorliegende Spannung ausgelegt ist. Zum Beispiel einen BC547B mit Verstärkungsfaktor von ca. 240.

3.) Basisstrom berechnen:

$$I_{Basis} = \frac{I_C}{Verstärkungsfaktor\ h_{FE}}$$

$$I_{Basis} = \frac{40\,mA}{240} = 170\,\mu A$$

4.) Basiswiderstand berechnen:

$$R_{Basis} = \frac{U_{Basis} - U_{BE}}{I_{Basis}}$$

Da zwischen Basis und Emitter ca. 0,6 V abfallen (bei Darlington-Transistoren das Doppelte) und vereinfacht von einer Pin-Ausgangsspannung von 5 V ausgegangen wird, ist die Spannung über dem Basisvorwiderstand 4,4 V.
Am Beispiel also: $R_{Basis} = \frac{4{,}4\,V}{170\,\mu A} = 25{,}9\,k\Omega$

Aus der Normreihe ginge ein 22-kΩ-Widerstand, wobei man zum sicheren Durchschalten des Transistors einen Übersteuerungsfaktor von 5 bis 10 ansetzen sollte.

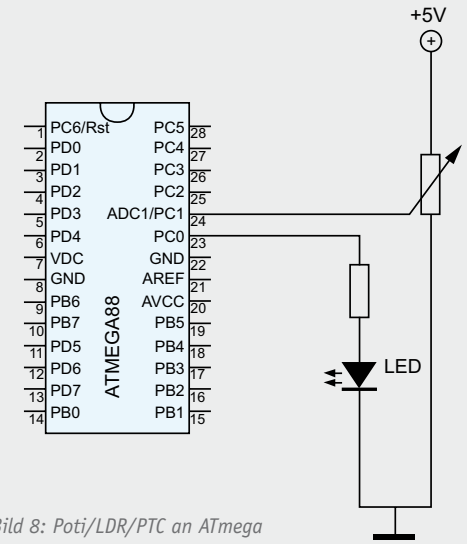


Bild 8: Poti/LDR/PTC an ATmega

logwandler geschrieben. Wenn der eingelesene Wert kleiner als 500 ist, dann wird die LED eingeschaltet – sonst wird die angeschlossene LED ausgeschaltet. Wenn man nun am Potentiometer dreht, dann sieht man, dass die LED je nach Potentiometerstellung an oder aus ist. Statt des Potentiometers kann man auch einen Spannungsteiler aus einem Festwiderstand und einem lichtempfindlichen Widerstand (LDR) oder einen Spannungsteiler aus einem Festwiderstand und einem temperaturempfindlichen Widerstand anschließen und beobachten, wie die LED auf Licht- bzw. Temperaturänderungen reagiert.

Elektronikwissen

Pull-up/Pull-down

Damit ein als Eingang geschalteter Portpin einen definierten Zustand annimmt, wenn er unbeschaltet oder ein angeschlossener Taster/Schalter offen ist, wird der Portpin mit einem Widerstand nach +UB oder nach GND geschaltet. Ein Widerstand nach +UB wird Pull-up-Widerstand genannt. Ein Widerstand nach GND wird Pull-down-Widerstand genannt. Dadurch hängt der unbeschaltete Portpin nicht in der Luft (und wirkt als kleine Antenne, durch die unvorhersehbare Signalzustände erzeugt und außerdem unnötig Strom verbraucht würde), sondern der Portpin liegt an logisch 1 (Pull-up-Widerstand) bzw. logisch 0 (Pull-down-Widerstand). Im Fall des Pull-up-Widerstands wird mit dem nach GND geschalteten Taster/Schalter dann von 1 auf logisch 0 geschaltet. Bei Verwendung eines Pull-down-Widerstands wird mit einem Taster/Schalter entsprechend nach +UB geschaltet, um logisch 1 zu erhalten.
Ein Pull-up-Widerstand ist bereits für jeden Portpin intern im AVR vorhanden und bei als Eingang definiertem Portpin mit Portx.n=1 aktivierbar. Man sieht daher sehr häufig nach GND geschaltete Taster/Schalter, weil man sich einen externen Pull-up-Widerstand sparen kann.

LC-Display

Eine interessante und einfache Möglichkeit ist das Anschließen eines LC-Displays an den Mikrocontroller zur Anzeige von Texten und Daten. Es werden lediglich sechs Leitungen zwischen Mikrocontroller und LC-Display benötigt. (Der Anschluss kann an beliebige Portpins erfolgen.) Dazu +5 V, GND und ein Potentiometer (z. B. 10 k) für die Kontrasteinstellung des Displays (Bilder 10 und 11). Geeignet sind Text-LC-Displays mit Standard-Controller (HD44780) wie zum Beispiel das ELV LC-Display 2x 16 (ELV STN-LCD, Best.-Nr. JU-05 41 84) oder das LCD-Addon von myAVR. Außer Displays mit zwei Zeilen à 16 Zeichen (2 x 16) gibt es auch Displays mit 2x 8, 2x 20, 4x 16 und 4x 20 Zeichen. Der Anschluss und die Ansteuerung mit BASCOM ist immer gleich. Das folgende BASCOM-Programm zeigt die Ansteuerung eines LC-Displays. Zunächst wird eine „Begrüßung" angezeigt („ELVjournal", „BASCOM-AVR"), und dann wird angezeigt, ob eine Taste gedrückt ist.

```
' BASCOM-Programm
'
' LCD-Basis
' In:  Taster an B.0
' Out: LCD an D.2 bis D.7
'
$regfile= "M88def.dat"                    'Verwendeter Chip
$crystal= 1000000                         'Verwendete Frequenz
$hwstack= 40                              'Ruecksprungadressen (je 2),
                                           Registersicherungen (32)
$swstack = 40                             'Parameteruebergaben (je 2),
                                           LOCALs (je 2)
$framesize = 60                           'Parameter (Daten-Laenge),
                                           Rechenbereich Funktionen

Config Portb.0 = 0                        'Port-Pin B.0 als Eingang
Portb.0 = 1                               'Pullup-Widerstand

'LCD-Konfiguration:
Config Lcdpin = Pin , Db4 = Portd.4 , Db5 = Portd.5 , Db6 = Portd.6 , _
                      Db7 = Portd.7 , E = Portd.3 , Rs = Portd.2
Config Lcd = 16 * 2
Cursor Off                                'Cursor ausschalten
Cls                                       'LC-Display löschen

Upperline : Lcd "ELVjournal"              '1.Zeile Ausgabe
Wait 1
Lowerline : Lcd "BASCOM-AVR"              '2. Zeile Ausgabe
Wait 3

Display Off                               'LCD-Modul ausschalten
Waitms 500
Display On                                'LCD-Modul einschalten
Wait 2
Cls                                       'LC-Display löschen

Locate 1 , 7                              'Positionieren
Lcd "ELV  "                               'Ausgabe

Do                                        'Schleifenbeginn
  Locate 2 , 1
  If Pinb.0 = 0 Then                      'Wenn Taste gedrückt dann..
    Lcd " Taste gedr{245}ckt"             '     (245=ü 225=ä 239=ö)
  Else                                    'sonst..
    Lcd Space(16)                         '16 Leerzeichen ausgeben
  End If
Loop                                      'Schleifenende
End                                       'Programmende
```

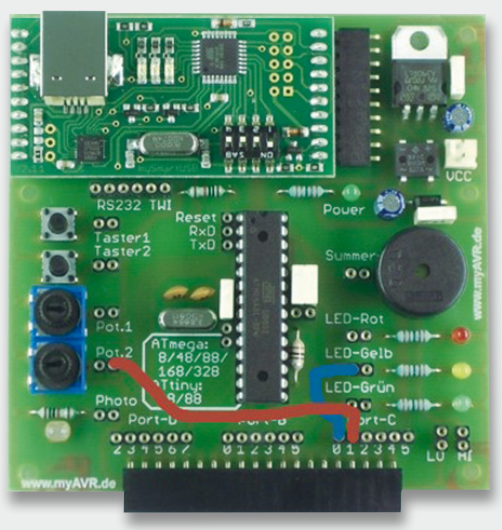


Bild 9: myAVR Board mit Potentiometeranschluss

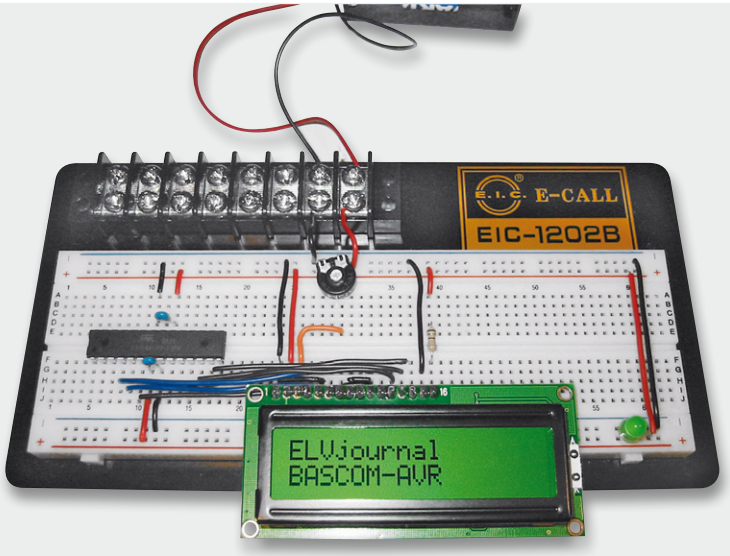


Bild 10: LCD am ATmega88

Erläuterungen:

Mit Config Lcdpin bzw. Config Lcd werden die Anschlusspins und der Typ des LC-Displays beschrieben. (Durch den Unterstrich am Ende einer Zeile ist es möglich, ein BASCOM-Statement – hier CONFIG – auf mehrere Zeilen zu verteilen.) Cursor Off bewirkt, dass kein Cursor auf dem Display angezeigt wird, was im Normalfall schöner aussieht, und mit CLS wird die Anzeige auf dem Display komplett gelöscht.

UPPERLINE bzw. LOWERLINE setzen die aktuelle Schreibposition auf die erste Stelle in der oberen bzw. der unteren Zeile des Displays. Mit LOCATE wird die Schreibposition an eine beliebige Stelle des Displays gesetzt. Die eigentliche Ausgabe von Text oder Variableninhalten wird mit dem Befehl LCD erreicht. Geschrieben wird an der jeweiligen (unsichtbaren) Schreibposition. Standard-LC-Displays können einen bestimmten, intern abgelegten Zeichensatz darstellen. Im Listing sieht man ein Beispiel, wie auch bestimmte Zeichen des Zeichensatzes – hier Zeichen Nummer 245 für „ü" – dargestellt werden können. Den kompletten Zeichensatz findet man im Datenblatt des Displays. Wenn die Taste an Pinb.0 gedrückt wird, wird der entsprechende Text und sonst 16 Leerzeichen angezeigt. Man könnte auch LCD " " schreiben statt LCD Space(16).

Die Ausgabe einer analogen Spannung ist nicht in den AVR-Mikrocontrollern vorbereitet. Zur Ausgabe einer analogen Spannung verwendet man ein R2R-Widerstandsnetzwerk, erzeugt ein PWM-Signal, welches mit einem Tiefpass geglättet wird, oder benutzt einen dedizierten Umwandlerbaustein.

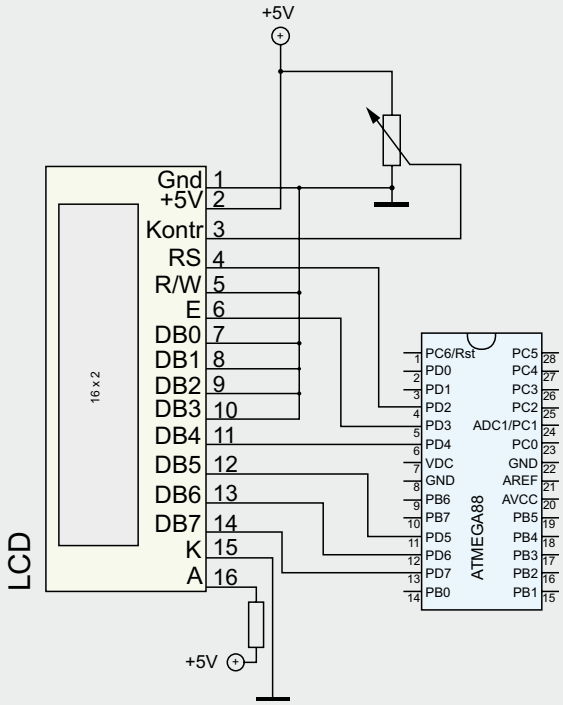
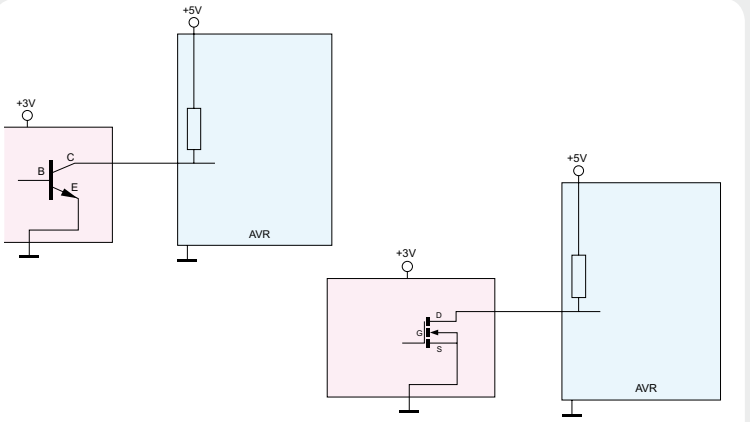


Bild 11: LCD-Anschluss

Elektronikwissen

Open Collector/Open Drain

In sehr vielen integrierten Schaltkreisen bzw. Modulen ist der Ausgang als Transistor mit offenem/unbeschaltetem Kollektor (Open Collector) oder als MOSFET mit offenem Drain-Anschluss (Open Drain) realisiert. Das hat den Vorteil, dass die angeschlossene Schaltung (z. B. der Mikrocontroller) auch mit einer anderen Spannung betrieben werden kann als das Modul. Im aktiven Zustand wird der Anschluss auf GND gezogen. Ansonsten ist der Pin offen und sollte daher zur Erzeugung eines ordentlichen Logiksignals mit einem (internen) Widerstand auf logisch 1 gezogen werden.



Ausblick

Weitere Verbindungen (Seriell, I²C, 1-Wire, SPI ...) basieren wie die in diesem Artikel beschriebenen I/Os im Prinzip auf digitalen Ein- und Ausgaben, kennzeichnen sich aber durch komplexe Protokolle und sind für jeweils spezielle Anwendungen geeignet. Sie werden teilweise in nachfolgenden Artikeln noch behandelt werden.

In der nächsten Folge der ELV-Artikelserie „Mikrocontroller-Einstieg mit BASCOM-AVR" werden Programmstrukturen und entsprechende BASCOM-Befehle erläutert werden.

ELV

Stückliste

Empfohlene Produkte/Bauteile:	Best.-Nr.	Preis
BASCOM-(Demo-)Lizenz von MCS Electronics [3]	–	–
Experimentier-Board 1202B	JU-07 72 89	€ 12,95
Schaltdraht-Sortiment	JU-05 47 68	€ 5,95
Atmel AVRISP mkII Programmer	JU-10 03 55	€ 39,95
oder myAVR Board MK2	JU-10 90 00	€ 49,–
ATtiny13	JU-10 03 39	€ 1,95
ATmega8	JU-05 29 71	€ 3,20
ATmega88	JU-10 07 62	€ 3,95
100-nF-Kondensator	JU-10 03 17	€ 0,08
Batteriehalter für 3x Mignon	JU-08 15 30	€ 0,75
Batterieclip für 9-V-Block-Batterie	JU-08 01 28	€ 0,30
LED-Set	JU-10 66 60	€ 1,65
oder Leuchtdioden	JU-10 63 56	€ 3,95
und Widerstände	JU-10 66 57	€ 1,85
Piezo-Signalgeber	JU-00 73 87	€ 0,95
Netzteil für myAVR Board MK2	JU-10 90 01	€ 6,95
Mikroschalter und -taster	JU-10 66 67	€ 2,80
Leistungsrelais 5 V/64 Ω -> 250 VAC 1x um	JU-00 97 47	€ 2,20
Relais A5W-K 5 V/178 Ω -> 125 VAC 2x um	JU-01 91 95	€ 4,85
Relais 1x um 12 V/360 Ω -> 250 VAC	JU-06 62 68	€ 1,95
Single-Inline-Relais 5 V/500 Ω -> 200 V	JU-00 66 77	€ 2,70
NPN Transistor BC547B		
Kollektorstrom IC max. 100 mA	JU-00 53 37	€ 0,04
Treiber 7fach ULN2003, je Treiber 500 mA	JU-01 84 61	€ 0,29
Treiber 7fach ULN2803, je Treiber 500 mA	JU-00 70 36	€ 0,87
CMOS CD4066 Digital-Analog-Switch	JU-00 55 35	€ 0,40
Mini-Erschütterungssensor MES 1		
Open-Drain-Ausgang Spannung 7–15 V	JU-07 38 22	€ 11,95
Dual-Opto-Koppler TLP 627-2		
Kollektorstrom bis 150 mA	JU-01 46 59	€ 1,15
Opto-Koppler CNY17 Kollektorstrom max. 50 mA	JU-00 56 13	€ 0,19
Temperatursensor KTY81, 1000 Ω	JU-00 61 83	€ 0,58
Temperatursensor	JU-03 70 75	€ 2,95
LC-Display 2x 16	JU-05 41 84	€ 9,95
oder myAVR LCD Add-On		
230-V-Schaltinterface SI230-2 5 V -> 230 V/16 A	JU-09 20 35	€ 24,95
MP3 Sound-Modul MSM 2	JU-09 28 53	€ 19,95
FS20-2/4-Kanal-Sendemodul FS20 S4M		
2 Sender/4 Kanäle	JU-06 68 17	€ 17,95
FS20-4/8-Kanal-Sendemodul FS20 S8M		
4 Sender/8 Kanäle	JU-09 22 04	€ 19,95
Diode 1N4148	JU-00 23 04	€ 0,02
FS20-1-Kanal-Universal-Empfänger FS20 UE1		
1x Open Collector	JU-08 58 27	€ 24,95
FS20-Funkschaltmodul FS20 SM 4		
Empfänger 4x Open Collector	JU-08 57 22	€ 19,95
FS20-8-Kanal-Schaltmodul FS20 SM8		
Empfänger 8x Open Drain 12 V/1 A	JU-09 22 10	€ 24,95
FS20-Fernbedienung FS20 S4 2/4-Kanal Handsender	JU-05 75 64	€ 19,95
FS20-Funkschaltsteckdose FS20 ST		
Empfänger 230 V	JU-10 45 37	€ 22,95
Nützlich:		
Pin-Ausrichter	JU-00 84 63	€ 4,95

Alle Infos zu den Produkten/Bauteilen finden Sie im Web-Shop.

Preisstellung Dezember 2012 – aktuelle Preise im Web-Shop

Weitere Infos:

[1] Stefan Hoffmann: Einfacher Einstieg in die Elektronik mit AVR-Mikrocontroller und BASCOM. Systematische Einführung und Nachschlagewerk mit vielen Anregungen. ISBN 978-3-8391-8430-1

[2] www.bascom-buch.de

[3] www.mcselec.com

[4] www.atmel.com