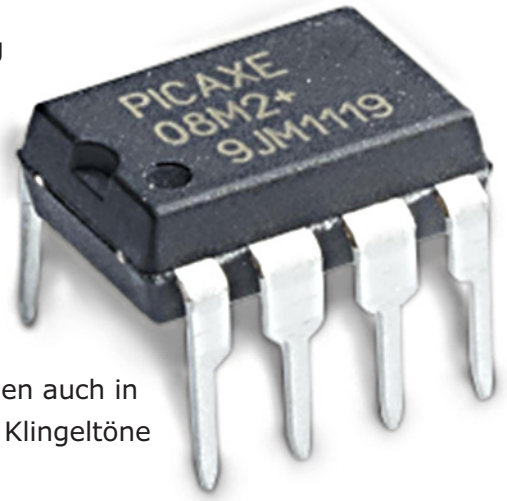
BASIC für PICs (3)

Über analoge Eingänge, PWM, Servos und musikalische PICs

Von Wouter Spruit
(Niederlande)



Im ersten Teil der Artikelserie ging es um die Programmierung des PICAXE-Chips und grundlegende Ein-und Ausgangsschaltungen (Elektor.POST Projekt Nr. 8). In der zweiten Tranche (Elektor.POST Projekt Nr. 16) legten wir mit der Steuerung verschiedener Arten von Schaltern nach und berechneten die Werte der dazugehörenden Bauteile. Dieses Mal beschäftigen wir uns mit dem Lesen und Verarbeiten analoger Eingangswerte und beschreiben Ausgangsschaltungen, die am Pulsweitenmodulator (PWM) angeschlossen sind. Wir werden auch in der Lage sein, mit PICAXE Servos zu steuern und monophone Klingeltöne zu erzeugen!

Wie in den vorangegangenen Teilen werden alle beschriebenen Experimente mit einem PICAXE 08M2-Mikrocontroller vorgenommen (Anschlussbelegung in **Bild 1**), der mit 5 V eines ATX-Netzteils versorgt wird. Beispiel-Schaltungen werden lötzinnfrei auf Steckboards aufgebaut. Die Programmierung des Chips wird über ein USB-zu-Seriell-Kabel vorgenommen, wie es in den vorangegangenen Artikeln beschrieben wurde [1]. Um Verwechslungen zu vermeiden, beziehen sich alle Pin-Nummern in den Schaltbildern auf die physikalischen Pin-Nummern des Chips (für alle Beispiele: Pin 5 ist ein PWM-Ausgang, Pin 6 der Analogeingang 1). Pin-Nummern in Listings beziehen sich auf interne Pin-Namen gemäß Bild 1. Wie man einen PICAXE programmiert, entnehmen Sie bitte den vorangegangenen Artikeln [1] oder dem PICAXE-Handbuch auf der Website [2]. PICAXE-Chips und diverse Peripherie sind im Online-Shop von Revolution Education [3] erhältlich.

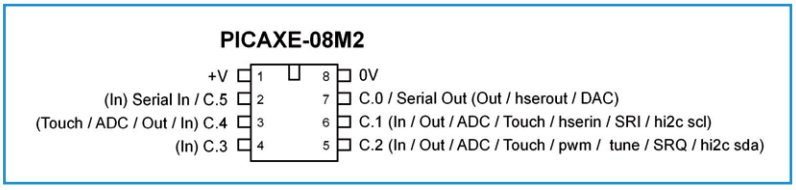


Bild 1.
Anschlüsse des PICAXE 08M2.

Analogeingang

In allen bisherigen Beispielen kam ein binärer Schalter zum Einsatz, um die Ausgangsschaltungen zu steuern. Allerdings ist in vielen Fällen eine feinere Steuerung vonnöten. Die Analog/Digital-Wandler (ADCs) des PICAXE können eine analoge Spannung normalerweise im Bereich von +5 V (Versorgungsspannung) und 0 V lesen und in einen digitalen 8-bit-Wert (bei einigen PICAXE-Chips 10 bit) verwandeln. Ein Potentiometer als einstellbarer Spannungsteiler wird als analoges Eingabegerät verwendet. Der Schleifer (mittlerer Anschluss) wird am ADC-Pin des PICAXE angeschlossen; die Spannung ist abhängig von seiner Position.
Dreht man den Schleifer zum 0-Ω-Anschluss, muss der Strom durch den ADC-Anschluss begrenzt werden, um eine Beschädigung der Bauteile zu vermeiden (wie in Teil 2 [1] erklärt).
Allerdings kann man nicht einfach Widerstände einfügen, um den Strom zu begrenzen, da dies auch Auswirkungen auf die ADC-Werte hätte. Ein 330-Ω-Widerstand zwischen

dem Schleifer und dem ADC-Pin begrenzt den Strom auf

$$\frac{5V}{330\Omega} = 0{,}0152\ A$$

was innerhalb der für den Pin erlaubten Stromaufnahme von 0,02 A liegt. In diesem Beispiel wird ein lineares 10-k-Poti eingesetzt. Somit wird die maximale Spannung über dem ADC:

$$5V \times \frac{10k\Omega}{330\Omega + 10k\Omega} = 5V \times 0{,}97 = 4{,}84V$$

Dieser relativ geringe Effekt kann durch die Software ausgeglichen werden. Über die Auswahl des Widerstands am ADC-Eingang wird in den PICAXE-Foren diskutiert [4]. Der Konsens dort ist, dass ein Wert von 10 kΩ das Optimum für ein Potentiometer am PICAXE-ADC (oder einem verwandten Microchip-PIC) darstellt.
Im Beispiel wird eine LED ein- und ausgeschaltet, der Potischleifer wie ein Schalter benutzt. **Bild 2** und **Bild 3** zeigen die Schaltungen von Ein- und Ausgang. Der erforderliche Code ist in **Listing 1** zu finden, während **Bild 4** den Aufbau dieses Experiments auf einem Steckboard zeigt.
Wenn in diesem Beispiel der ADC-Wert zwischen 2,5 V und 5 V liegt, wird die LED eingeschaltet, ansonsten bleibt sie dunkel. Es ist offensichtlich, dass es besser gewesen wäre, einen einfachen Ein/Aus-Schalter zu verwenden. Im nächsten Beispiel aber wird die gleiche Schaltung verwendet, um die Helligkeit der LED zu steuern. Der ADC des PICAXE verhält sich übrigens manchmal merkwürdig, wenn die Eingangsimpedanz etwa 20 kΩ übersteigt. Möchten Sie mehr darüber wissen, schauen Sie einmal in die Foren [4].

PWM-Grundlagen

PICAXE besitzt keine analogen Ausgänge. Allerdings kann ein PWM-Ausgang eine ganze Reihe von nützlichen Dingen und darunter auch etwas, was einem analogen Ausgang sehr nahe kommt. Der PWM-Pin gibt fortlaufend ein Rechtecksignal aus und dies im Gegensatz zu normalen PICAXE-Befehlen im Hintergrund, parallel zum Rest der Programmausführung. Ein PWM-Signal wird durch Frequenz und Tastverhältnis (duty cycle) definiert, wie in **Bild 5** gezeigt. Die Frequenz ist die Anzahl der sich wiederholenden Zyklen

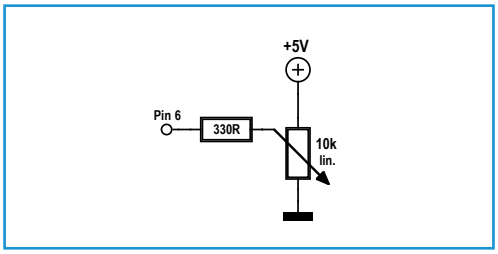


Bild 2.
Schaltung des Analogeingangs.

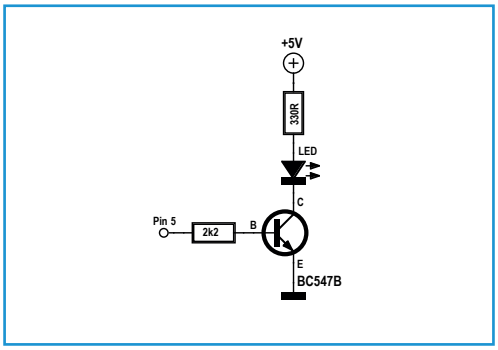


Bild 3.
Schaltung des PWM-Ausgangs.

Listing 1: Analog-Eingang.

```
init:
low 2
main:
do
  readadc 1,b1
  if b1 > 127 then
    high 2
  else
    low 2
  endif
loop
```

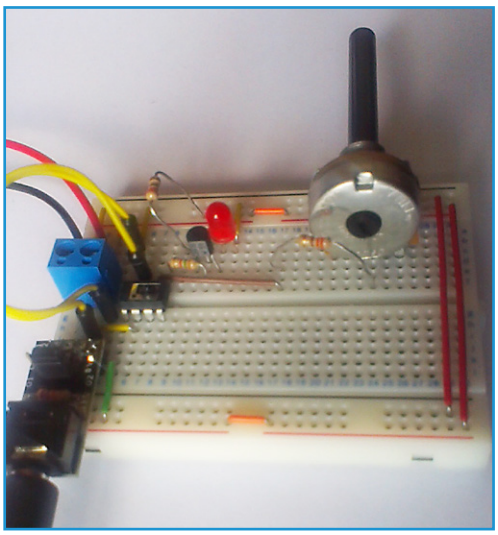


Bild 4.
Der PWM-Ausgang auf dem Breadboard.

pro Zeiteinheit (in der Regel Sekunden), die Periode ist die Zeitdauer eines Zyklus und das Tastverhältnis die Einschaltdauer (aktiver High-Zustand) im Verhältnis zu einem Zyklus (in Prozent). Obwohl es Software gibt (wie die RPi.GPIO-Python-Bibliothek für Software-PWM auf einem Raspberry Pi [5]), die es dem Benutzer erlaubt, die PWM-Frequenz und das Tastverhältnis direkt einzugeben, ist beim PICAXE das Einrichten der PWM eine größere Herausforderung. Es gibt einige diesbezügliche Befehle, aber der neueste und relevanteste ist „`pwmout`". Da dieser Befehl nicht so einfach zu verstehen ist, verweisen wir auf die grundlegende Erläuterung von `pwmout` in [2], anstatt hier alles von Grund auf zu erklären. Die Frequenz des PWM-Signals basiert auf der Frequenz des PICAXE-Takts (*resonator speed*). Die meisten PICAXE-Chips unterstützen mehrere Taktraten, aber der `pwmout`-Befehl funktioniert nur mit einer begrenzten Anzahl von Frequenzen. Üblicherweise ist der Standard-Takt für `pwmout` 4 MHz. Die meisten einfachen Anwendungen für den `pwmout`-Befehl sind „`pwmout pin,period,duty cycles`", um das Signal zu aktivieren und „`pwmout pin,OFF`", um es wieder auszuschalten. Die Variablen `period` und `duty cycles` werden verwendet, um die PWM-Frequenz (f_{PWM}), die Periode (T_{PWM}) und das Tastverhältnis (D_{PWM}) zu setzen:

$$\frac{1}{f_{PWM}} = T_{PWM} = (period + 1) \times 4 \times resonator\ speed$$

$$D_{PWM} = duty\ cycles \times resonator\ speed$$

mit `resonator speed` = 1/4000000 für 4 MHz.

Beachten Sie, dass beim `pwmout`-Befehl die Zeitdauer statt des Prozentsatzes zur Definition des Tastverhältnisses angegeben wird! Glücklicherweise hilft hier der PWMout-Assistent. Es gibt eine Reihe von Assistenten in der (LinAxePad-)Software, der für PWM zuständige findet sich unter PICAXE → Wizards → PWMout. Der Assistent ermöglicht es, die PWM-Frequenz in Hz und das Tastverhältnis wie gewohnt in Prozent anzugeben. Heraus kommt ein `pwmout`-Befehl komplett mit Variablen, der ein PWM-Signal nach Ihren Vorgaben erzeugt (begrenzt durch die Auflösung des `pwmout`-Befehls).

PWM ist nützlich, um LEDs zu dimmen oder die Drehzahl eines DC-Motors zu steuern. Eine LED gehorcht nicht dem Ohmschen Gesetz (siehe den vorherigen Artikel). Um ihre Helligkeit ohne eine einstellbare Stromquelle zu steuern, kann der PWM eingesetzt werden, der die LED schnell ein- und ausschaltet. Mit einem Tastverhältnis von 50% wird die LED nur für 50% der Zeit eingeschaltet. Da die Frequenz des PWM-Signals sehr hoch ist, sieht man kein Blinken, das träge menschliche Auge nimmt nur eine gemittelte Helligkeit wahr.

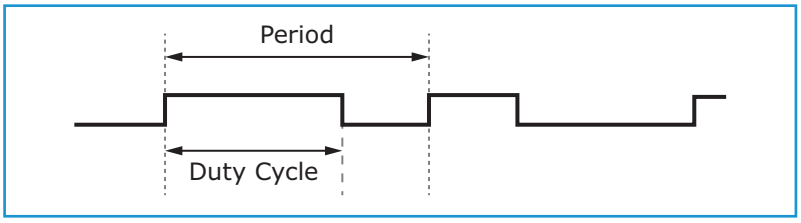


Bild 5.
Struktur des PWM-Signals.

Die PWM-Frequenz, die man zum Dimmen einer LED verwendet, sollte also hoch genug sein, so dass man sie nicht mehr wahrnehmen kann. Beachten Sie die Ausgabe-Auflösung: Je näher die Frequenz an die maximale Frequenz des PICAXE rückt, desto geringer wird die Anzahl der Bits, um das Tastverhältnis einzustellen.

Dieses Beispiel verwendet eine LED, um das Prinzip hinter der Steuerung der „Ausgangsintensität" durch PWM zu demonstrieren, aber das gleiche gilt beispielsweise für die Steuerung der Geschwindigkeit eines DC-Motors. PICAXE liefert nicht genügend Strom für einen Motor, so dass ein Transistor (wie im vorherigen PICAXE-Artikel erläutert) verwendet werden muss, um den Motor zu betreiben. Dort haben wir auch die Transistor-Schaltgeschwindigkeit erwähnt: Diese sollte man etwas im Auge behalten, wenn wir die PWM-Frequenz festlegen. Ist der Transistor nicht schnell genug, wird das PWM-Ausgangssignal verzerrt und daher unbrauchbar. Eine Senkung der PWM-Frequenz oder ein schnellerer Transistor dürfte das Problem lösen. Vergessen Sie nicht, dass Darlington-Paare noch langsamer als einzelne Transistoren sind.

Die Schaltung ist identisch mit dem vorigen Beispiel (Bild 2 und Bild 3 für Ein- und Ausgang), aber dieses Mal ist die Software ein wenig komplizierter, wie man in **Listing 2** sehen kann. Die Berechnungen passen die Eingabemöglichkeiten (0-255) auf den Bereich der Ausgangswerte (0-100) an. Im nächsten Abschnitt machen wir Gebrauch davon.

Das neue Programm verwendet das 8-bit-ADC-Eingangssignal (Schleiferposition des Potis), um das Tastverhältnis des PWM-Ausgangssignal festzulegen und die LED zu dimmen. Sie sind nun in der Lage, die Helligkeit mit dem Potentiometer zu steuern!

Am Servo drehen

Ein Servomotor ist ein Aktuator für die präzise Einstellung einer Position. Ein typischer Servo wird über drei Leitungen, schwarz für Erde, rot für die positive Versorgungsspannung und üblicherweise weiß für Daten angeschlossen. Ein Servo versucht, eine Position einzunehmen, die durch ein Signal am Datenanschluss vorgegeben wird. Wenn das Signal ausbleibt, hält der Servo eine Position nicht. Der Servo erwartet ein Rechteck-Trägersignal mit einer Periode von 20 ms (Frequenz = 1 / 0,020 = 50 Hz). Das Tastverhältnis (Periode von 0,75 ms bis 2,25 ms) bestimmt den Winkel. Wir verwenden dazu nicht den `pwmout`-Befehl, sondern zwei andere Kommandos. Der erste Befehl „`servo pin, pulse`“ initialisiert einen PWM–fähigen Ausgang für den Servoanschluss. Als nächstes wird der Befehl „`servopos pin,pulse`“ verwendet, um die Servo-Position zu ändern. Beim Anschluss des Servos wird die Datenleitung mit dem (PWM-fähigen) PICAXE-Ausgangspin über einen Strombegrenzungswiderstand (zum Beispiel 330 Ω) verbunden. Der Servo selbst benötigt eine separate Stromversorgung: Er zieht einen recht hohen Strom und sorgt für ein ordentliches Rauschen auf der Stromversorgungsleitung. In der Praxis bedeutet dies, dass sich der PICAXE fehlerhaft verhalten könnte, wenn die Stromversorgungen nicht getrennt sind! Viele Servos benötigen ohnehin eine höhere Spannung als der Rest der Elektronik. 0 V beider Stromversorgungen sollten miteinander verbunden werden, um einen gemeinsamen Bezugspunkt bereitzustellen. Der Kondensator an der Versorgungsspannung des Servos reduziert die Welligkeit. Für die Servo-Stromversorgung wurden hier vier 1,5 V-Mignonzellen (total 6 V) verwendet (keine Akkus, sie haben niedrigere Nennspannungen von etwa 1,2 V).
Die Werte für die Variable `pulse` werden voraussichtlich innerhalb des Bereichs 75...225 liegen, entsprechend der Periode von 0,75...2,25 ms, die der Servo erwartet. Wie ein Servo angeschlossen wird, zeigt die Schaltung in **Bild 6**. Wieder bleibt die analoge Eingangsschaltung die gleiche wie in Bild 2. Der Code, um den Servo durch ein Poti am analogen Eingang zu steuern, ist in **Listing 3** zu sehen, der Aufbau auf dem Steckboard in

Listing 2: PWM-Ausgang.

```
init:
low 2
main:
do
  readadc 1,b1  ;get analog value
  if b1 > 250 then  ;clamp value
    b1 = 250
  endif
  w1=b1*10   ;w1 is a 16-bit register (b2 with b3)
  w1=w1/25   ;needed for integer arithmetic
  pwmout 2,24,b2;b2 is the significant part of w1
loop
```

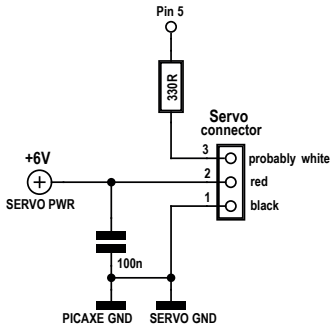


Bild 6.
Schaltung des Servomotors.

Listing 3: Servosteuerung.

```
init:
servo 2,100   ;init servo
main:
do
  readadc 1,b1  ;get analog input
  if b1 > 250 then  ;clamp value
    b1 = 250
  endif
  w1=b1*3  ;scale value
  w1=w1/5
  b2=b2+75
  servopos 2,b2 ;update servo position
loop
```

Bild 7. Da die Register des PICAXE nur 8-bit- oder 16-bit-Integervariablen erlauben, müssen wir etwas tricksen, um diese Einschränkung zu umgehen und den ADC-Eingang (0-255) auf den Ausgangsbereich (0-100) umzurechnen. In diesem Fall ist der Bereich der Ausgangswerte 75...225, also muss ein Bereich von 225-75 = 150 skaliert werden. Da 150/250 = 3/5 ist, multiplizieren wir den ADC-Wert mit 3 (erfordert eine 16-Bit-Variable, da 3 × 255 natürlich nicht in einen 8-Bit-Wert von 255 passen) und teilen dann durch 5. Dann addieren wir den Minimalwert von 75 hinzu und es ergibt sich der skalierte Ausgangswert `b1` im richtigen Bereich. Beachten Sie, dass `b2` in dem Code die unteren 8 bit von `w1` darstellen, da `w1` ein 16-bit-Wert ist, der aus den zwei 8-bit-Werten `b2` und `b3` besteht, wie im vorigen Teil [1] beschrieben. Eine präzise Steuerung des Servowinkels wäre zum Beispiel in der Robotik sehr nützlich, aber solche Projekte würden wahrscheinlich viel mehr Servos verwenden als ein PICAXE PWM-Ausgänge zur Verfügung stellen könnte. Eine Lösung für dieses Problem wären spezielle Servo-Treiber-ICs. PICAXE könnte diese Treiber ansprechen und diese dann die Positionen für mehr als 20 Servos einstellen. Zum Beispiel erlaubt der AXE031 die Steuerung von bis zu 21 Kanälen [3]. Im nächsten Artikel werden wir besonders darauf eingehen, wie der PICAXE mit solchen peripheren Bausteinen kommunizieren kann.

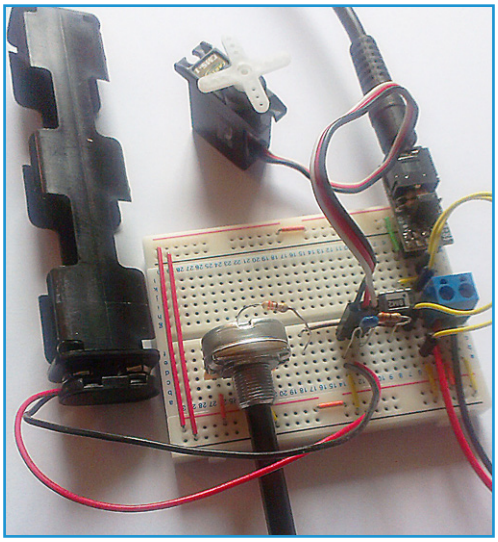


Bild 7.
Der Servomotor auf dem Steckboard.

Singender PICAXE: Melodien und Sounds erzeugen

Eine weitere PWM-basierte Funktionalität des PICAXE ist der „`tune`"-Befehl. Die Syntax des Befehls ist bei diversen PICAXE-Typen wegen der unterschiedlichen Anzahl der PWM-Ausgänge verschieden. Wir verwenden die Version für 8-polige PICAXE-Chips. Der „`tune`"-Befehl erzeugt einstimmige (monophone) Melodien in einem angeschlossenen Lautsprecher, ähnlich den altmodischen Handy-Klingeltönen. Es ist möglich, auch andere Ausgänge mit dem Befehl zu schalten, beispielsweise um eine LED als „Lichtorgel" zu betreiben. Der Befehl „`tune`" kann nur auf einen PWM-fähigen Ausgang bezogen werden, im Falle des PICAXE 08M2 gibt es davon nur einen. Der Befehl „`play`" ist ebenfalls verfügbar, spielt aber vorprogrammierte Melodien.
Das PICAXE-Handbuch Teil 2 [6] enthält eine umfangreiche Dokumentation über den „`tune`"-Befehl sowie Beispielschaltungen mit einem piezoelektrischen Summer, einem Lautsprecher und einem externen Audio-Verstärker. Für das „`tune`"-Beispiel verwendet der Autor einen ausrangierten Kopfhörer. Die Schaltung im Handbuch empfiehlt 40...80-Ω-Lautsprecher. Wenn Sie einen niederohmigen Lautsprecher einsetzen wollen, müssen Sie einen Widerstand in Reihe schalten, um mindestens 40 Ω zu erhalten.
Es steht ein weiterer Assistent zur Verfügung (unter PICAXE → Wizards → Ring Tone Tunes), der beim Komponieren einer Melodie mit dem „`play`"-Befehl (und dem Import von Klingeltönen im RTTTL-Format, zum Beispiel von [7]) behilflich ist. Auch wenn dieser Befehl Melodien einfach und schnell erstellen kann, ist auch der „`PWMout`"-Befehl in der Lage, Melodien manuell zu erzeugen und Sound-Effekte zu erstellen, ähnlich denen, die in alten Atari-Spielen verwendet werden. Hinweis: Verändern Sie die Wave-Parameter schnell in For-Schleifen mit unterschiedlichen STEP-Parametern und verschiedenen „eingebetteten" For-Schleifen.
Die Schaltung mit einem Lautsprecher (aus dem PICAXE-Handbuch Teil 2 in [6]) ist in **Bild 8** dargestellt, der Code für eine einzelne Melodie in **Listing 4**. Es ist wichtig, dass dieser Code in nur einer einzigen Zeile untergebracht ist (erkennen Sie die Melodie?), damit es funktioniert. Der Versuchsaufbau ist in **Bild 9** gezeigt. Lassen Sie den Komponisten in sich heraus und programmieren Sie eigene Melodien! Die Eingangsschaltung wird

übrigens hier nicht gebraucht, aber bauen Sie sie nicht ab, wir benötigen sie für die folgende Aufgabe.

Der PWM-Ausgang wird analog

PICAXE besitzt nur digitale Ausgänge. Aber es ist doch möglich, eine analoge Spannung mit dem PWM und einem Tiefpassfilter auszugeben. In den vorherigen Teilen [1] haben wir das Prinzip des Spannungsteilers kennengelernt. Wenn ein Potential an eine Anzahl von Widerständen in Reihe angelegt wird, stehen die Spannungsabfälle über die einzelnen Widerstände im Verhältnis zu ihren jeweiligen Werten. Ein PWM-Rechtecksignal durch einen Tiefpass-Filter funktioniert genauso. Der zweite Widerstand des Spannungsteilers ist durch einen Kondensator ersetzt, so dass die Spannung über dem Kondensator proportional zu dem Tastverhältnis des PWM-Signals ist. Allerdings gilt dies nur, wenn die Frequenz der PWM deutlich größer ist als die Grenzfrequenz des Tiefpassfilters. Die Formel für diese Grenzfrequenz ist:

$$f_C = \frac{1}{2\pi\ RC}$$

Eine größerer RC-Wert ergibt eine glattere Ausgangsgleichspannung, aber das Filter ist auch langsamer: Wenn sich die PWM-Frequenz ändert, braucht die analoge Ausgangsspannung mehr Zeit, um sich entsprechend zu ändern. Um die Bauteile auszuwählen, können wir den ersten Widerstand zu berechnen, um den Strom zu begrenzen, und dann einen Wert für C bestimmen mit der Formel

$$C = \frac{factor}{2\pi\ R\ f}$$

wobei „factor“ als Multiplikator dient, um die Grenzfrequenz des Filters weit entfernt von der PWM-Frequenz festzulegen. Auf der Website [8] kann man die Auswirkungen von PWM-Frequenz und RC-Wert visualisieren. Die Simulation zeigt, wie sich der Analogausgang mit unterschiedlichen Signal- und Schaltungsparametern ändert. Bei der Berechnung des RC-Werts ist es ratsam, mindestens einen Faktor 100 zu verwenden, um ein Ausgangssignal mit relativ geringer Welligkeit zu erhalten.
Es ist nicht sinnvoll, die höchstmögliche PWM-Frequenz einzusetzen, weil der PICAXE seine Feinkontrolle über das Tastverhältnis dann verliert. Um das Tastverhältnis in 100 Schritten verändern zu können, ist eine Frequenz von 4 MHz / 100 = 40 **k**Hz sinnvoll. Der Wert des Parameters `duty cycles` im `pwmout`-Befehl ist jetzt bequemerweise gleich dem Tastverhältnis in Prozent. Mit einer PWM-Frequenz von 40 kHz und einem (Strombegrenzungs-)Widerstand R von 10 kΩ ist

$$C = \frac{100}{2\pi \times 10000 \times 40000} =$$

$$3{,}98 \times 10^{-8} = 39{,}8nF$$

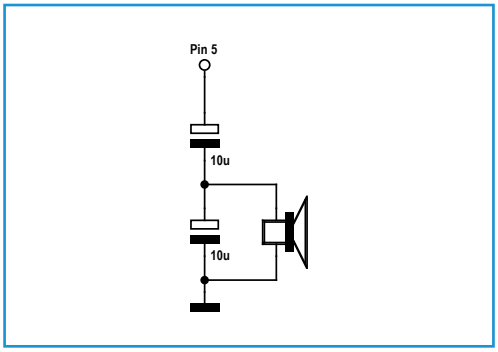


Bild 8.
PICAXE als Tongenerator.

Listing 4: Erzeugung einer Melodie (eine Zeile)

```
tune 1,4,($67,$69,$40,$69,$04,$4C,$04,$22,$4C,$67,$69,$40
,$69,$02,$4C,$02,$00,$4C,$6B,$29,$67,$69,$40,$69,$C0,$02,
$2B,$29,$27,$27,$C2,$E0,$67,$69,$40,$69,$04,$4C,$04,$22,$
4C,$67,$69,$40,$69,$C7,$2B,$20,$6B,$29,$67,$69,$40,$69,$C
0,$02,$2B,$29)
```

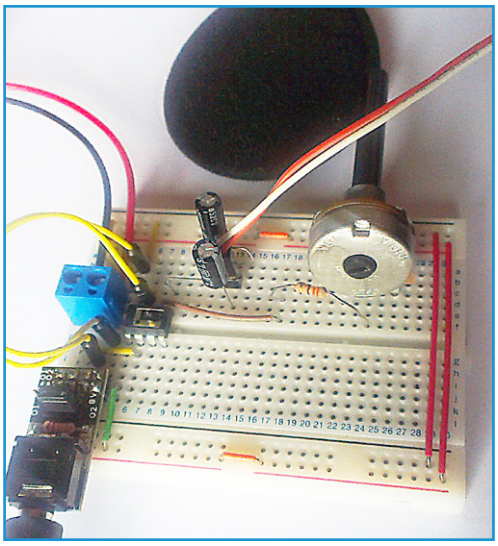


Bild 9.
Steckboard mit Lautsprecher (ein ausgedienter Kopfhörer).

Der nächstgelegene Kondensator-Wert ist 47 nF. Ein (wesentlich) größerer Kondensator-Wert kann verwendet werden, um eine geringere Welligkeit zu erreichen.
Diese Ausgangsschaltung ist in **Bild 10** dargestellt, während die Eingangsschaltung gleich Bild 2 bleibt. Wir verwenden den gleichen Code wie in Listing 2, weil die PWM-Frequenz bereits 40 kHz beträgt und die richtige Skalierung ebenfalls enthalten ist. Den Aufbau auf dem Breadboard zeigt **Bild 11**.
Der analoge Eingang wird verwendet, um die analoge Ausgangsspannung über das PWM-Tastverhältnis zu steuern. Die Welligkeit des Ausgangssignals kann mit einem Oszilloskop dargestellt, oder, wenn die Ausgangsspannung glatt genug ist, mit einem Standard-Voltmeter angezeigt werden. Es wäre möglich, den Ausgang direkt wieder zu einem analogen Eingang des PICAXE zu führen, um die Ausgangsspannung mit dem ADC zu messen, aber wir haben hier keine Möglichkeit, die Ergebnisse anzuzeigen.

Heute und morgen

Neben der Programmierung eines PICAXE-Chips und der Steuerung einfacher Elektronik können wir nun analoge Eingänge nutzen. Wir haben eine feinere Kontrolle über unsere Ausgangsschaltungen erlangt. Wir können einen PWM-Ausgang verwenden, um ein analoges Ausgangssignal zu erzeugen, einen „gedimmten“ Ausgang realisieren, altmodische Klänge programmieren und sogar die Position eines Servos genau festlegen. Doch im letzten Beispiel sind wir auf ein Problem gestoßen: Wir können noch keine Daten anzeigen, zum Beispiel als Teil einer Schnittstelle zu komplexeren PICAXE-Programmen oder Werte, die von einem analogen Eingang gesammelt werden. Don‘t panic! In der nächsten Folge von Elektor.POST werden wir weitere PICAXE-Funktionen nutzen, um mit verschiedenen Peripheriegeräten zu kommunizieren. Damit können wir ein PICAXE-basiertes Projekt um ein alphanumerisches OLED-Display erweitern.

(130262)

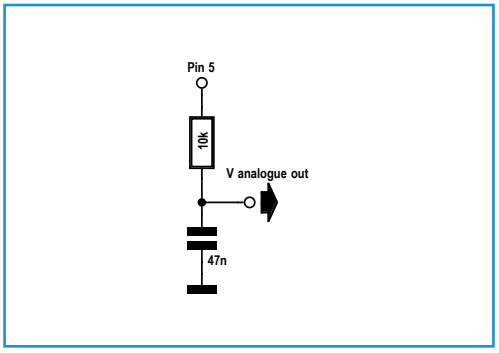


Bild 10.
Schaltung des Analogausgangs.

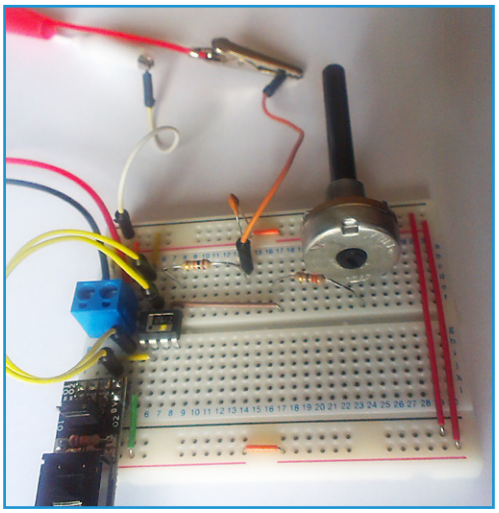


Bild 11.
Der Analogausgang auf dem Breadboard.

Weblinks

[1] „BASIC für PICs“, (1) und (2), Elektor. POST Projekte Nr. 8 und Nr.16. www.elektor-magazine.com/extra/post

[2] www.picaxe.com/Getting-Started/PICAXE-Manuals

[3] www.techsupplies.co.uk/PICAXE

[4] www.picaxeforum.co.uk/forum.php

[5] https://code.google.com/p/raspberry-gpio-python/wiki/PWM

[6] www.picaxe.com/docs/picaxe_manual2.pdf

[7] www.picaxe.com/RTTTL-Ringtones-for-Tune-Command/

[8] http://sim.okawa-denshi.jp/en/PWMtool.php