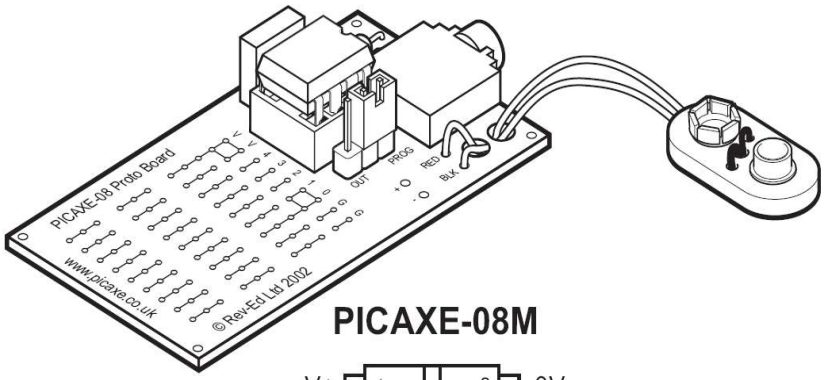
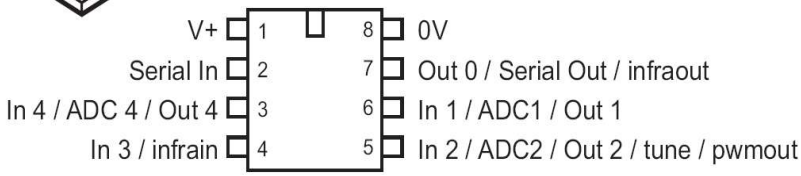
PICAXE-08M MIKROCONTROLLER PROGRAMMIER SYSTEM





Die 'PICAXE' sind ein einfach zu programmierendes Mikrocontroller-System, welche die Eigenschaften einer neuen Generation von Low-Cost, Flash-basierten Mikrocontrollern ausnutzen. Diese Mikrocontroller können immer wieder und wieder programmiert werden, ohne das ein zusätzliches teures Programmiergerät dafür gebraucht wird.

Die Stärke des PICAXE-08M Systems liegt in seiner Einfachheit. Kein Programmier- oder Löschgerät oder anderes komplizierte elektronische System ist notwendig – der Mikrocontroller wird programmiert (mit einem einfachen BASIC-Programm oder grafischen Flussdiagramm) über eine 3-Draht Verbindung mit dem Seriellen Anschluss des Computers. Die PICAXE Schaltung benutzt nur 3 Komponenten und kann einfach auf einem Steckbrett, Lochrasterplatte oder Leiterplatte realisiert werden.

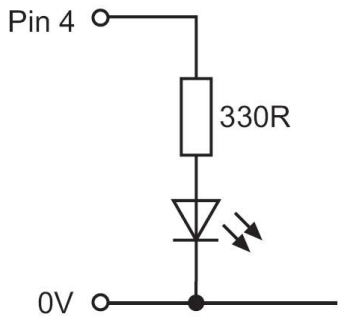
Die PICAXE-08M Mikrocontroller bieten 5 Ein-/Ausgänge.

- *Günstiger Preis, einfache Erstellung einer Schaltung*
- *5 Eingänge/Ausgänge (3 Eingänge wahlweise mit 10bit Analog-Digital Wandlung)*
- *Erweiterte Infrarot-Fähigkeiten (infrain2)*
- *Hardware-PWM Generation (pwmout)*
- *Musikfunktionen (4 vorprogrammierte Stücke und frei nutzbar)*
- *Schneller Download mit seriellem Kabel*
- *Freie, einfach zu handhabende Programm-Editor Software*
- *Einfach zu lernende BASIC Programmiersprache*
- *Wahlweise auch mit Flussdiagrammen programmierbar*
- *Freie Dokumentationen und Online Support Forum*

Das Starterpaket enthält die folgenden Teile:
- *PICAXE-08 Prototypen Leiterplatte*
- *Download-Kabel*
- *CDROM mit Programmen und Dokumentationen*
- *PICAXE-08M Mikrocontroller Chip*

Download des ersten Programmes

Dieses erste einfache Programm kann genutzt werden, um Ihr System zu testen. Es erfordert die Verbindung einer LED und einem 330Ohm Widerstand mit Output Pin 4 (Bein 3 des Controllers). Soll das ganze auf einer selbstgemachten oder Lochrasterleiterplatte erstellt werden, wird die LED mit dem Ausgangs-Pin des PICAXE und dem Widerstand nach Ground (GND) verbunden. Dabei ist darauf zu achten, dass die LED richtig herum eingesetzt wird.



Anmerkung: Der PICAXE Mikrocontroller kann mit einem Computer verbunden werden über das PICAXE-08 Prototypenbord oder über eine einfache selbstgemachte Leiterplatte (z.B. Lochraster- oder Streifenleiterplatte). Die Anschlüsse zur Programmierung sind weiter unten beschrieben.

1. Verbinden Sie das PICAXE-Programmierkabel mit dem seriellen Anschluss des Computers (normalerweise als COM1 oder COM2 bezeichnet).
2. Starten Sie den Programm-Editor.
3. Wählen Sie Ansicht > Einstellungen um den Optionen-Dialog zu erhalten (dieser Dialog kann beim Programmstart auch automatisch erscheinen).
4. Klicken Sie auf den 'Modus' Tab und wählen Sie PICAXE-08M
5. Klicken Sie auf den 'Serieller Port' Tab and wählen Sie den Anschluss am PC aus, an dem das Programmierkabel angeschlossen ist.
6. Klicken Sie 'OK'
7. Geben Sie das folgende Programm ein:

```
main: high 4
pause 1000
low 4
pause 1000
goto main
```

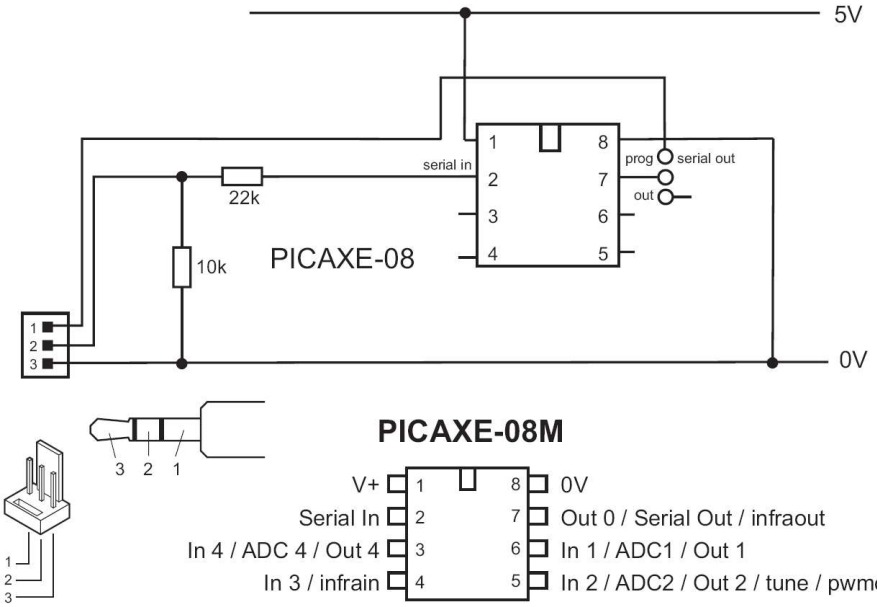
   (Bitte beachten Sie den Doppelpunkt (:) direkt hinter der Marke 'main' und die Leerzeichen zwischen den Befehlen und Zahlen)
8. Stellen Sie sicher, dass die PICAXE Schaltung mit dem seriellen Kabel verbunden ist und das die 4.5V Batterien angeschlossen sind. Weiterhin muss die LED und der 330Ohm Widerstand mit dem PICAXE-Ausgang verbunden sein.
9. Wenn das PICAXE-08 Prototypenbord genutzt wird, muss sich der Jumper in der 'PROG' Position befinden.
10. Wählen Sie im Menü des Programm-Editors PICAXE > Start.
    Ein Fortschrittsbalken sollte erscheinen der den Programm-Download anzeigt. Wenn der Download fertig ist, startet das Programm im PICAXE automatisch – die LED an Output 4 sollte eine Sekunde leuchten und dann wieder für eine Sekunde verlöschen.

Fehlerbehebung

Wenn eine Fehlermeldung erscheint, überprüfen Sie bitte den angezeigten Hinweis. Öfters vorkommende Fehler sind::
• *Serielles Programmierkabel zwischen Computer und PICAXE nicht oder nicht richtig angeschlossen.*
• *Der Download-Jumper befindet sich nicht in der 'PROG' Position.*
• *Falsche COM-Port Einstellung im Programm-Editor.*
• *Batterie nicht angeschlossen oder Spannung zu niedrig.*

Die PICAXE-08 Schaltung

Die PICAXE-08 Grundschaltung sieht wie folgt aus.

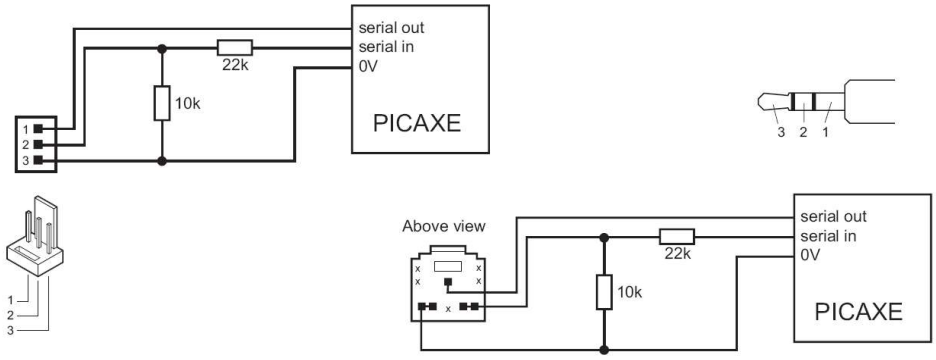


Der PICAXE-08M Mikrocontroller

Bitte beachten Sie, der PICAXE-08M Mikrocontroller ist kein leerer Mikrocontroller! Die PICAXE-Mikrocontroller sind vorprogrammiert mit einem Bootlader Programm, dass den direkten Kabel-Download überhaupt erst ermöglicht. Leere Mikrocontroller enthalten dieses Bootlader-Programm nicht und können deshalb in einem PICAXE-System nicht benutzt werden.
Bitte beachten Sie weiterhin, die Eingangs-/Ausgangs Pin Nummern sind nicht identisch mit den „Bein-Nummern" des Schaltkreises. So hat z.B. der Output 0 Pin die Bein-Nummer 7 (siehe Darstellung).
Diese Dokumentation enthält eine kurze Einführung in das PICAXE-08 System. Weitergehende Informationen finden Sie in den 'PICAXE Tutorial', 'BASIC Commands' und 'Electronic Interfacing' Hilfe Dateien.

Die PICAXE Computer Interface Schaltung

Das PICAXE System benutzt ein sehr einfaches Interface zum seriellen Anschluss des Computers. Obwohl dieses Interface nicht die vollen RS232 Spannungen ausschöpft, ist es wegen seiner Einfachheit sehr preisgünstig und hat seine Funktionstüchtigkeit bislang voll unter Beweis gestellt. Es arbeitet einwandfrei mit den allermeisten der modernen Computer.



Es wird sehr empfohlen, dass diese Interfaceschaltung auf allen Leiterplatten, die die PICAXE-Mikrocontroller verwenden mit integriert wird. Das erlaubt die jederzeitige Reprogrammierung des PICAXE ohne diesen aus der Schaltung zu entfernen.

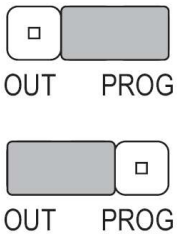
Anmerkung:

Die meisten Computer haben zwei serielle Ports, bezeichnet als COM1 und COM2.
Der Programm-Editor muss für den korrekten Port konfiguriert werden – wählen Sie **Ansicht > Einstellungen >Serieller Port** um die Einstellung für Ihren PC vorzunehmen.

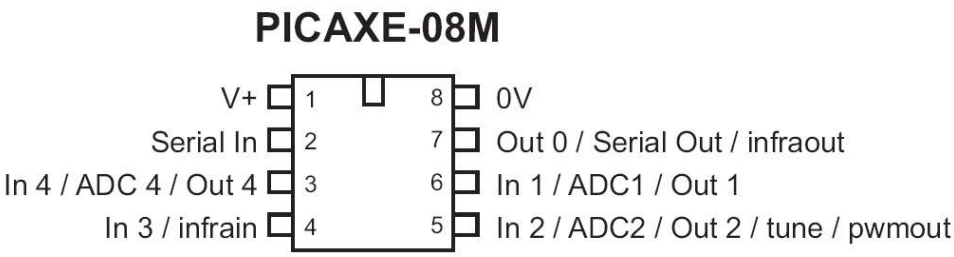
In letzter Zeit werden Computer angeboten, die statt mit seriellen RS232-Schnittstellen mit sogenannten USB-Schnittstellen ausgerüstet sind (z.B. Notebooks).
In diesem Fall wird die Verwendung eines USB-zu-Seriell-Adapters empfohlen, z.B. dem von Revolution Education sehr preisgünstig angebotenen und verifizierten USB010 Adapters. Dieser Adapter erstellt nach Installation unter Windows einen sogenannten virtuellen COMPort, der genau wie ein physikalisch vorhandener Anschluss im Programm-Editor eingetragen werden kann.

Jumper:

Das PICAXE-08 Prototypenbord hat einen Jumper um zwischen der seriellen Verbindung (erforderlich während Programmdownload) und der normalen Funktion des Output Pin 0 umzuschalten. Beide Funktionen nutzen das gleiche Bein des Mikrocontrollers, deshalb ist die richtige Jumperposition sehr wichtig.
Wird Output 0 im Programm nicht genutzt, so kann der Jumper ständig in Programmierposition verbleiben.



Ein- und Ausgänge



Der PICAXE-08M Mikrocontroller hat 5 Ein-/Ausgangs Pins. Diese Pins sind nicht vordefiniert, wie bei anderen PICAXE-Mikrocontrollern, der Benutzer kann festlegen ob ein Pin als Ein- oder Ausgang genutzt wird.
Pin 0 ist dabei jedoch immer ein Ausgang und Pin 3 muss immer als Eingang genutzt werden (wegen der internen Konstruktion des Mikrocontrollers). Die anderen 3 Pins können entweder als Eingang oder als Ausgang genutzt werden.
Falls gewünscht können Pin 1, 2 und 4 als Analogeingang (bis 10bit Auflösung) genutzt werden.
Die Hardware-PWM-Möglichkeiten und die Musikfunktionen des PICAXE-08M funktionieren nur mit dem Pin 2.

Wichtig – kommen Sie nicht durcheinander!

Die Input/Output Pin Nummern sind NICHT die gleichen, wie die externen ‘Bein-Nummern' des Mikrocontrollers. Die Pin-Nummern folgen der Port-Spezifikation und finden sich auch in der Programmierung wieder.
Um Verwirrung zu vermeiden sprechen die PICAXE-Dokumentationen von 'Beinen’ wo auf die externe physikalische Nummerierung des ICs Bezug genommen wird.

Bein	Beschreibung	Bemerkung
1	Positive Spannung, V+	3-5V Batterie (Pluspol) als Stromversorgung
2	Serial In	Verwendet für Programm Download
3	Pin 4	Input oder Output oder Analog Input
4	Pin 3	Nur Input
5	Pin 2	Input oder Output oder Analog Input
6	Pin 1	Input oder Output oder Analog Input
7	Pin 0 / Serial Out	Nur Output. Auch verwendet für Programm Download
8	Ground, 0V	Minuspol Stromversorgung (0V)

Anmerkung - Output Pin 0

Pin 0 (Bein 7) wird während des Programm Downloads benutzt, kann aber auch als normaler Ausgang verwendet werden nachdem der Download abgeschlossen ist.
Auf den Projektbords erlaubt ein Jumper die Zuordnung zum Download Sockel (PROG Position) oder zum Ausgang (OUT Position). Um Ihr Programm zu testen müssen Sie den Jumper immer in die korrekte Position setzen!
Wenn Sie eine eigene Leiterplatte entwerfen, können Sie einen ähnlichen Jumper oder kleinen Schalter vorsehen. Alternativ ist es ebenfalls möglich den Mikrocontroller sowohl mit der Programmierschaltung als auch mit dem Ausgang zu verbinden. Dabei ist zu beachten, dass während des Programmdownloads schnelle Impulse auf der Leitung auftreten. Das kann u.U. zur Gefahr werden, wenn gleichzeitig ein Motor o.ä. Gerät angeschlossen ist. Eine LED z.B. würde dann einfach leuchten und ist damit problemlos anschließbar.

Festlegen Eingang oder Ausgang

Beim Einschalten des PICAXE-08M, werden alle Pins als Eingangs Pins konfiguriert (außer Pin 0, der immer Ausgang ist). Es gibt 3 Methoden, diese Pins als Ausgang zu konfigurieren (wenn notwendig):

Methode 1 – Verwenden eines Befehls, der den Pin als Ausgang nutzt.

Das ist die einfachste Methode, die auch am meisten von den Einsteigern benutzt wird. Sobald ein Befehl verwendet wird, der ein Ausgangspin erfordert (wie z.B. high, low, toggle, serout oder sound), wird dieses Pin als Ausgangspin deklariert. Das Pin bleibt auch weiterhin als Ausgangspin gesetzt.
Damit ist der einfachste Weg ein simples 'low' am Programmanfang für alle Pins, welche als Ausgang arbeiten sollen. Das aktiviert die Ausgangstreiber im PICAXE und setzt gleichzeitig die Ausgänge auf Low (GND).

Methode 2 – Verwenden des Input und Output Befehls.

Der Befehl ‘output ?’ (das Fragezeichen steht für die Pin Nummer) kann ebenfalls genutzt werden, um die Datenrichtung des Pins als Ausgang festzulegen. Genauso legt ‘input ?’ das Pin als Eingang fest, obwohl dies eigentlich nicht notwendig ist, da die Pins nach einem Reset als Eingang deklariert sind.
Beachten Sie, dass der output-Befehl das Pin nicht auf einen definierten High oder Low Pegel setzt, so ist es oft einfacher den ‘low’ oder 'high' Befehl stattdessen zu nutzen.
Die 'input' und 'output' Befehle haben keine Wirkung auf Pin 0 (Output) und Pin 3 (Input), die sich nicht ändern lassen.

Methode 3 – (fortgeschritten) Verwenden des 'let dirs = ' Befehls

Der Befehl ‘let dirs = %000100111’ kann benutzt werden um die Richtung aller Pins gleichzeitig festzulegen. Das ist schneller als mehrfache input/output Befehle aber erfordert ein Verständnis der Bitdarstellung des Eingabe-/Ausgabeports (erklärt im 'Basic Commands manual'). Eine 0 für jedes Bit (Pin-Nummer) setzt das zugehörige Pin auf Eingang, eine 1 auf Ausgang. Der Wert der Bits 0,3,5,6,7 kann beliebig 0 oder 1 sein, da sie nicht änderbar sind und vom Mikrocontroller ignoriert werden.

Auswahl Pin 1, 2, 4 als Analogeingang.

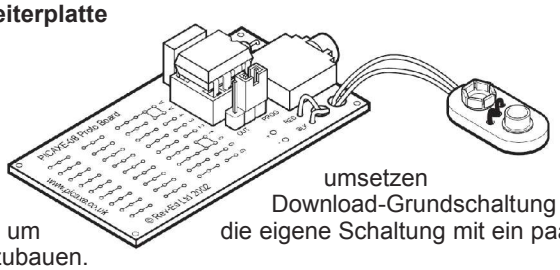
Die Benutzung des readadc oder readadc10 Befehls setzt automatisch das angegebene Pin als Analogeingang. Damit konfiguriert z.B. der Befehl ‘readadc 1,b2’ das Pin 1 als Analogeingang und speichert den gelesenen Wert in die Variable b2.

Aufbau der Schaltung

Die PICAXE-08 Schaltung kann sehr einfach auf einem Steckbrett oder auf einer Universalleiterplatte (Lochraster- oder Streifenleiterplatte) aufgebaut werden. Ebenfalls lässt sich eine richtige Leiterplatte entwerfen und ätzen.

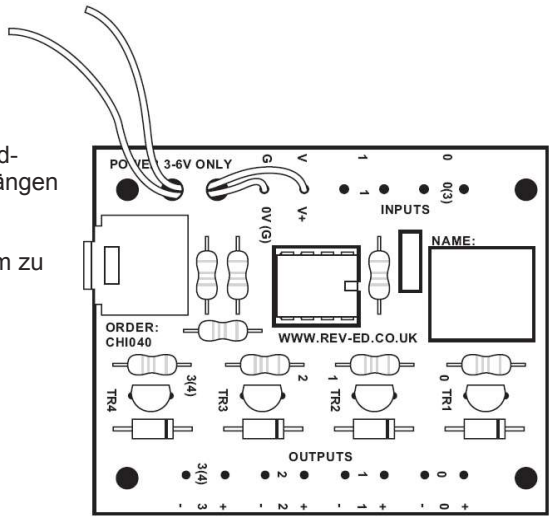
Um den schnellen Aufbau voranzutreiben gibt es folgende Prototypenleiterplatten:

AXE021 PICAXE-08 Prototypen- Leiterplatte



Diese Prototypen-Leiterplatte wurde entwurfen, damit der Nutzer sein Projekt mit dem PICAXE-08 Mikrocontroller in kürzester Zeit umsetzen kann. Die Leiterplatte enthält die Download-Grundschaltung mit einer kleinen 'Prototypen'-Fläche um die eigene Schaltung mit ein paar Ein- und Ausgabekomponenten aufzubauen.
Die Ein- und Ausgänge können auf dem Bord beliebig angeordnet werden.

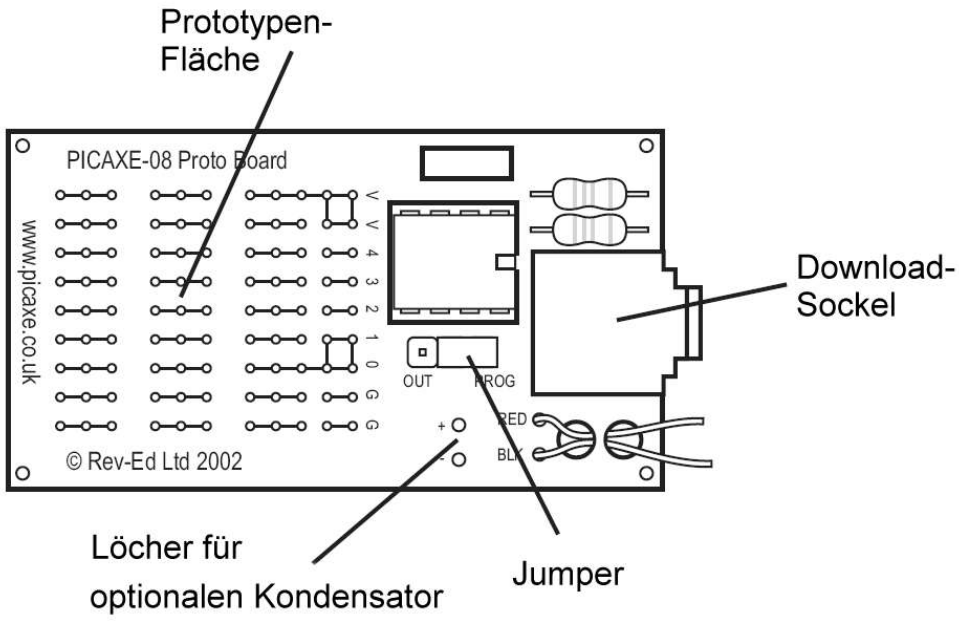
CHI040 Projekt-Bord



Das Projektbord enthält die Download-Grundschaltung mit bis zu vier Ausgängen und einem Eingang.
Die Ausgänge sind gepuffert mit Transistoren um einen höheren Strom zu schalten. Wenn gewünscht können unbenutzte Ausgänge ebenfalls als Eingänge verwendet werden.

Das PICAXE-08 Prototypen-Bord (AXE021)

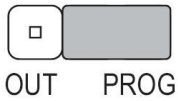
Diese Prototypen-Leiterplatte wurde entwurfen, damit der Nutzer sein Projekt mit dem PICAXE-08 Mikrocontroller in kürzester Zeit umsetzen kann. Die Leiterplatte enthält die Download-Grundschaltung mit einer kleinen 'Prototypen'-Fläche um die eigene Schaltung mit ein paar Ein- und Ausgabekomponenten aufzubauen.

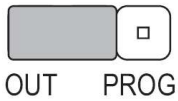


Die Prototypen-Fläche enthält Lötpads, die jeweils als Gruppe von 3 Pads verbunden sind (markiert durch Linien auf der Bestückungsseite). Jeder Ein-/Ausgang des Mikrocontrollers ist verbunden mit zwei Lötpads auf dem Prototypen-Feld, ebenfalls gibt es mehrere V+ und G (0V) Lötpads zur Versorgung.
Das erlaubt dem Nutzer sehr schnell einfache Schaltungen aufzubauen und in Betrieb zu nehmen ohne ein spezielles Bord dafür entwerfen zu müssen.

Die Versorgungsspannung kann über einen Batterieclip angeschlossen werden, der als Zugentlastung durch die größeren Löcher geführt wird.
Die PICAXE-08 Mikrocontroller arbeiten bei Batteriespannungen zwischen 3 und 5.5V (keine 6V oder 9V Batterie verwenden).
Falls Komponenten angeschlossen werden sollen, die Störungen erzeugen können (z.B. Motoren oder Hupen) wird empfohlen einen zusätzlichen Kondensator (z.B. 100uF 16V Elektrolyt) in die vorgesehenen Löcher zu bestücken um die Versorgungsspannung zu stützen.

Beachten Sie, dass der Jumper während eines Programm-Downloads bestückt sein muss (PROG Position) oder während des normalen Betriebes mit dem Output Pin 0 (OUT Position) verbunden, da beide Funktionen sich ein Mikrocontroller-Bein teilen.
Wird der Output 0 nicht genutzt, kann der Jumper ständig in der PROG Position verbleiben.



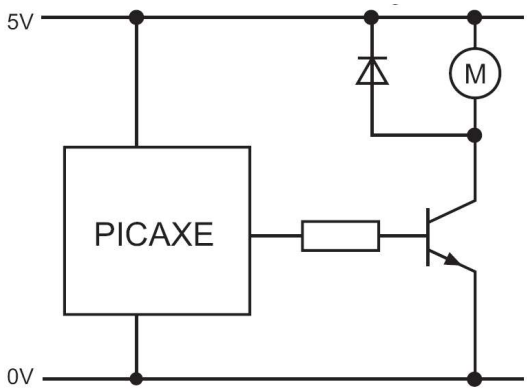


Allgemeine Eingangs-/Ausgangs-Schaltungen

(siehe auch Dokumentation 'Interfacing Electronics')

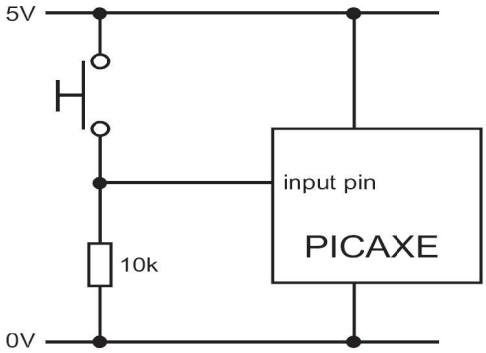
Digitale Ausgänge

Digitale Ausgänge können mit einem Transistor beschalten werden (z.B. BC548B) wie hier gezeigt, um den möglichen Strom zu erhöhen.



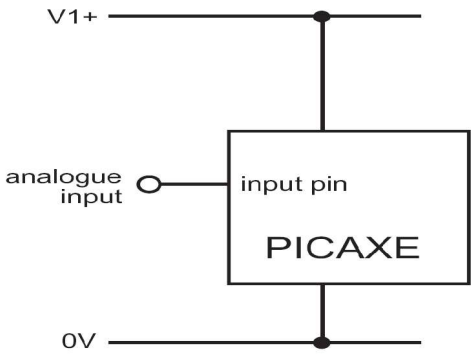
Digitale Eingänge

Digitale Eingänge können mit einem 10kOhm Pull-Down Widerstand wie folgt beschaltet werden:



Analoge Eingänge

Die Pins 1, 2, 4 können alternativ als Analogeingang genutzt werden. Nach Einschalten des PICAXE sind die Pins als Digital Input konfiguriert, nach Verwendung des Befehls ‘readadc’ oder 'readadc10' wird das angegebene Pin als Analogeingang rekonfiguriert.



BASIC Programmer Grundlagen

Die folgenden Programme sollen einen kurzen Einblick in ein paar grundlegende Programmiertechniken der PICAXE Mikrocontrollerfamilie geben.
Alle Programme können mit einer LED (mit 330Ohm Widerstand) an Output 4 und 0, einem Taster an Input 3, einem Piezo Schallgeber an Pin2 und einem Fotowiderstand (LDR, mit 1K Pullup) an Analogeingang 1. Diese Anordnung wird ebenfalls beim Cyberpet AXE101 genutzt.

Für weitere Details über jedes Programm lesen Sie die 'Basic Commands', 'Electronics Interfacing' und 'PICAXE Tutorial helpfiles' im Programm-Editor.

Ausgänge an- und ausschalten

Das folgende Programm schaltet Output 4 jede Sekunde an und aus. Das Programm zeigt die Benutzung der **high, low**, **wait**, **pause** und **goto** Befehle.
Nach dem Download lässt dieses Programm die LED an Output Pin 4 an- und ausschalten.

```
main:              ' erzeuge eine Marke mit dem Namen 'main'
     high 4        ' schalte Output 4 an
     wait 1        ' warte 1 Sekunde
     low 4         ' schalte Output 4 aus
     pause 1000    ' wart 1000ms (= 1 second)
     goto main     ' springe zurück zum Start
```

Die erste Zeile erzeugt eine Marke (Label) mit dem Namen 'main'. Eine Marke wird benutzt als Positionsmerker (Sprungadresse) im Programm. In diesem Programm benutzt die letzte Zeile die Marke 'main' um mit 'goto main' zurück an die erste Zeile zu springen. Das erzeugt eine Endlosschleife, die immer wieder durchlaufen wird.

Beachten Sie, dass der 'high 4' Befehl automatisch das Pin 4 als Output Pin konfiguriert. Eine Marke kann ein beliebiges Wort sein (Schlüsselwörter wie 'high' ausgenommen), müssen aber mit einem Buchstaben beginnen. Bei der Definition muss die Marke mit einem Doppelpunkt (:) abgeschlossen werden. Der Doppelpunkt 'sagt' dem Computer, dass das Wort als eine neue Marke zu interpretieren ist.
Normalerweise setzt man ein paar Leerzeichen (oder Tabs) an den Zeilenanfang jeder folgenden Zeile, beginnend ab der Marke. Das erhöht die Lesbarkeit und Verständlichkeit der Programme.
Kommentare können hinter einem Apostroph (') hinzugefügt werden, ebenfalls um die Lesbarkeit zu erhöhen.

Die Befehle **wait** und **pause** erzeugen Zeitverzögerungen. Dabei ist der Befehl **wait** für größere Verzögerungen (Auflösung in Sekunden) gedacht, **pause** kann auch kürzere Verzögerungen erzeugen (gemessen in Millisekunden).

Auswerten der Eingänge

Das folgende Programm aktiviert kurzzeitig Output 4 (LED) wenn ein Taster an Input 3 betätigt wird.

```
main:                          ' erzeuge eine Marke 'main'
     if pin3 = 1 then flash    ' Springe wenn Input ist ein
     goto main                 ' anderenfalls gehe zum Start

flash:                         ' erzeuge eine Marke 'flash'
     high 4                    ' schalte Output 4 an
     pause 500                 ' warte 0,5 Sekunden
     low 4                     ' schalte Output 4 aus
     goto main                 ' springe zurück zum Start
```

In diesem Programm bilden die ersten 3 Zeilen eine Endlosschleife. Wenn der Eingang ausgeschalten ist, durchläuft das Programm immer wieder und wieder diese Zeilen.

Wenn der Eingang eingeschalten wird, springt das Programm zur Marke **'flash**'. Daraufhin wird der Ausgang für eine halbe Sekunde eingeschalten und kehrt zur Hauptschleife zurück.
Bitte beachten Sie die Syntax in der 'if…then' Zeile – 'pin3' ist ein Wort (ohne Leerzeichen). Weiterhin darf sich hinter dem Befehl 'then' nur eine Marke befinden, es sind keine anderen Befehle erlaubt.

Benutzung von Symbolen

Manchmal kann es schwierig sein sich zu erinnern, welche Pins mit welchen Geräten verbunden sind.
Der 'symbol' Befehl kann am Anfang des Programms benutzt werden, um Ein- und Ausgänge umzubenennen.

```
symbol red = 4                 ' output4 heißt ab jetzt 'red'
symbol green = 0               ' output0 heißt ab jetzt 'green'
main:                          ' erzeuge eine Marke 'main'
     high red                  ' rote LED an
     wait 2                    ' warte 2 Sekunden
     low red                   ' rote LED aus
     high green                ' grüne LED an
     wait 1                    ' warte 1 Sekunde
     low green                 ' grüne LED aus
     goto main                 ' springe zurück zum Start
```

For…Next Schleifen

Es ist oftmals nützlich einen Programmabschnitt eine bestimmte Anzahl zu durchlaufen, zum Beispiel wenn ein Licht 25 mal aufblitzen soll. In diesem Fall kann eine **for…next** Schleife benutzt werden.

```
symbol red = 4              ' output4 heißt ab jetzt 'red'
symbol counter = b0         ' definiert einen Zähler mit Variable b0
main:                       ' erzeuge eine Marke 'main'
     for counter = 1 to 25  ' startet eine for…next Schleife
          high red          ' rote LED an
          pause 500         ' warte 0,5 Sekunden
          low red           ' rote LED aus
          pause 500         ' warte 0,5 Sekunden
     next counter           ' nächster Schleifendurchlauf
end                         ' Programmende
```

In diesem Programm wird der Abschnitt zwischen 'for' und 'next' 25 mal ausgeführt. Die Anzahl der Wiederholungen wird im Register b0 gespeichert. Die Variable 'counter' ist ein Symbol für das Register b0 (Vereinbarung in Zeile 2).
Es gibt 14 verfügbare Variablen, 'b0' bis 'b13' die benutzt werden können. Variablen sind Speicherbereich die für das einfache Ablegen von Zahlen genutzt werden können.

Verwenden von Variablen und Erzeugung von Tönen

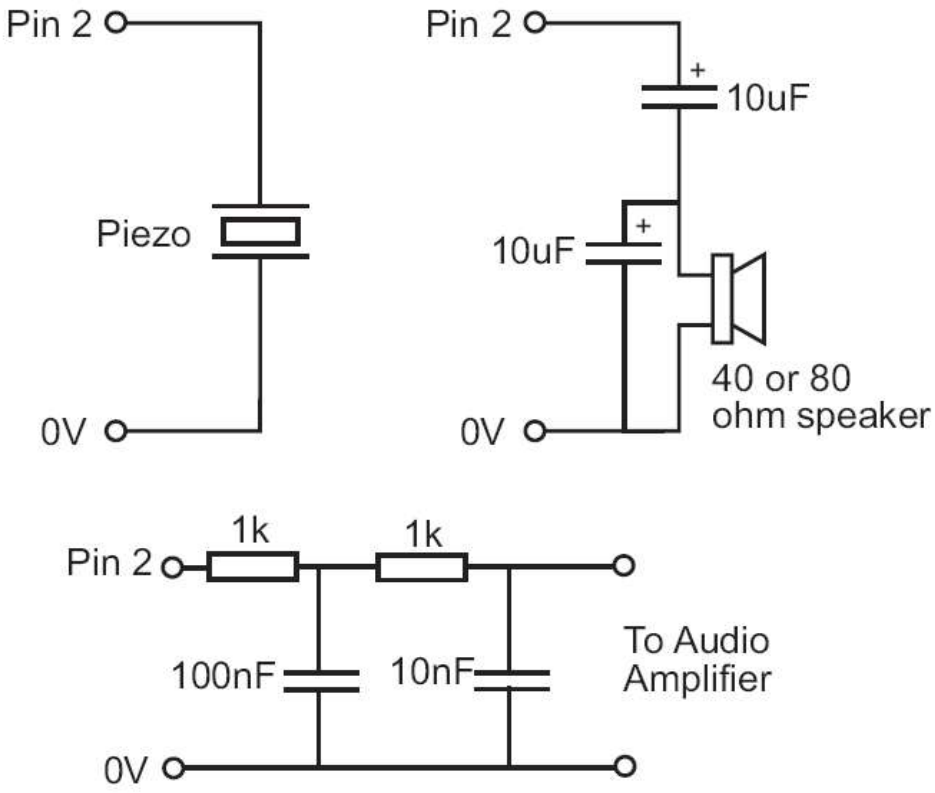
Die Variablen können auch genutzt werden um numerische Werte während des Programmablaufs zu speichern. Das folgende Programm erzeugt eine Folge von verschiedenen Tönen am Ausgang 2 (bei angeschlossenem Piezo-Schallgeber).

```
main:                       ' erzeuge eine Marke 'main'
     let b0 = b0 + 1        ' addiere eins zu b0
     sound 2,(b0,50)        ' erzeuge einen Ton mit der Tonhöhe b0
     pause 10               ' warte 0,01 Sekunden
     goto main              ' springe zurück zum Start
```

Die Variable b0 ist eine Byte-Variable. Das heißt es werden die Werte von 0-255 unterstützt. Weiterhin entsteht beim fortlaufenden Addieren ein Überlauf nach der höchsten Zahl: (…253-254-255-0-1-2…)
Das ist sehr wichtig zu wissen, wenn mathematische Operationen mit Byte-Variablen durchgeführt werden.
Der Piezo-Schallgeber arbeitet übrigens am besten mit Tonhöhen-Werten zwischen 50 und 150, Werte über 150 geben im allgemeinen keinen brauchbaren Ton.
Weitere Informationen zu den PICAXE-Variablen finden Sie in der 'BASIC Commands' Hilfe-Datei.

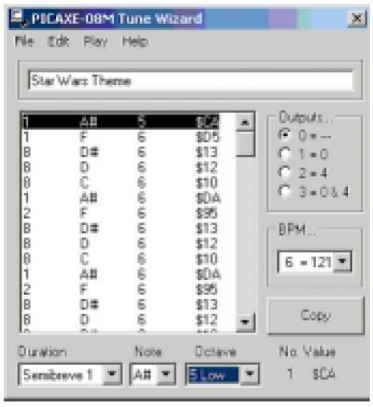
Nutzung der Musik-Fähigkeiten des PICAXE-08M

Der PICAXE-08M hat vier eingebaute Melodien (Happy Birthday, Rudolf Red Nosed Reindeer, Stille Nacht, Jingle Bells), die über den einfachen Befehl 'play' abgespielt werden können. Dazu wird an Pin 2 ein Piezo-Schallgeber angeschlossen. Auch lässt sich ein kleiner Lautsprecher über einen Serienkondensator wie im Bild gezeigt anschließen. Ein Verstärker kann man bei Bedarf über die gezeigte RC-Kombination anschließen.



Der 'Tune' Befehl erlaubt das Abspielen von kompletten eigenen Melodien. Um die Erstellung des Programms dafür zu vereinfachen gibt es im Programm-Editor einen speziellen Melodien-Wizard.

Der Melodien-Wizard (ab Programm-Editor Version 4.1.0) ermöglicht auch den Import von monofonen Klingeltönen im Nokia-RTTTL Format. Verschiedene Melodien sind im Internet zum Download verfügbar (Urheberrecht beachten!).



Lesen der Analogeingänge

Der PICAXE-08M hat 3 Analogeingänge. Mit dem readadc-Befehl (oder readadc10) kann der Wert eines Eingangs in eine Variable kopiert werden. Die Variable value (Lichtpegel) (Wert zwischen 0 und 160) kann dann abgefragt werden.
Das folgende Programm schaltet eine LED ein, wenn der Wert größer als 150 ist und eine andere LED, wenn der Wert kleiner als 100 wird. Liegt der gelesene Wert zwischen 100 und 150 sind beide LEDs aus.

```
main:                       ' erzeuge eine Marke 'main'
     readadc 1,b0           ' liest Analogeingang1 in Variable b0
     if b0 > 150 then red   ' Wenn b0 > 150 dann gehe nach red
     if b0 < 100 then green ' Wenn b0 < 100 dann gehe nach green
     low 4                  ' anderenfalls schalte Output 4 aus
     low 0                  ' schalte Output 0 aus
     goto main              ' springe zurück zum Start
red:                        ' Marke red
     high 4                 ' schalte Output 4 ein
     low 0                  ' schalte Output 0 aus
     goto main              ' springe zurück zum Start
green:                      ' Marke green
     high 0                 ' schalte Output 0 ein
     low 4                  ' schalte Output 4 aus
     goto main              ' springe zurück zum Start
```

Der PICAXE-08M Mikrocontroller hat 3 analoge Eingangskanäle (Pin 1, 2, 4) im Gegensatz zum PICAXE-08, der nur einen analogen Eingang besitzt (nur 8bit).
Um beim PICAXE-08M die Eingänge mit der Auflösung von 10bit auszulesen, muss der Befehl readadc10 benutzt werden. Als Parameter ist eine Wortvariable (16bit) zu übergeben.

```
main:                       ' erzeuge eine Marke 'main'
     readadc10 1,w0         ' liest 10bit Wert von Analogeingang1 in
                            ' Variable w0
     debug w0               ' überträgt Wert zum Computer
     pause 100              ' kurze Pause (100ms)
     goto main              ' springe zurück zum Start
```

Beim Arbeiten mit einem AnalogSensor ist es oft notwendig, einen Schwellwert zu bestimmen (z.B. die Werte 100 und 150 im obigen Programm). Der 'debug'-Befehl bietet eine einfache Möglichkeit, die Werte in 'Echtzeit' am PC anzusehen, so dass die Schwellwerte zumeist experimentell durch Ausprobieren bestimmt werden können.
Nach dem übertragen des Programms erscheint ein 'Debug-Fenster', in dem der Wert der Variablen angezeigt wird.
Eine Alternative bietet der Befehl 'sertxd'. Dabei wird der Wert als Textzeile zum Computer zurück übertragen und kann mit dem Terminalprogramm des Programm-Editors (oder einem beliebigen anderen Terminalprogramm) dargestellt werden.

```
main:                       ' erzeuge eine Marke 'main'
     readadc10 1,w0         ' liest 10bit Wert von Analogeingang1 in
                            ' Variable w0
     sertxd ("Value: ",#w0,13,10)' überträgt Text zum Computer
     pause 100              ' kurze Pause (100ms)
     goto main              ' springe zurück zum Start
```

Das '#' vor der Variable bewirkt die Übertragung der Variablen als Textwert, das '13,10' ist einfach das Zeichen für eine neue Zeile und wird vom Terminalprogramm interpretiert.

Bitte beachten Sie, dass es bei 'debug' und 'sertxd' durch die extrem einfache und preisgünstige Realisierung des internen Taktgenerators mitunter zu Problemen bei der Darstellung auf manchen Computersystemen kommen kann.

Die PICAXE-08M Befehle

Die Liste zeigt eine Zusammenfassung der verfügbaren Befehle, die von den PICAXE-08M unterstützt werden. Für weitere Details wird auf die Online-Hilfe unter 'Hilfe > BASIC Commands' verwiesen.

DIGITALE AUSGABE

HIGH	Setzt ein Ausgangs-Pin auf High (an).
LOW	Setzt ein Ausgangs-Pin auf Low (aus).
TOGGLE	Wechselt den Status eines Ausgangs-Pins.
OUTPUT	Konfiguriert ein Pin als Output.
INPUT	Konfiguriert ein Pin als Input.
REVERSE	Kehrt die Input/Output Richtung eines Pins um.
PULSOUT	Gibt einen Puls mit einer bestimmten Länge aus.

ANALOGE AUSGABE

PWM	PWM Erzeugung (veraltet, besser pwmout verwenden).
PWMOUT	PWM Erzeugung im Hintergrund (Hardware basiert)
SOUND	Gibt einen Ton aus.
PLAY	Spielt vordefinierte Melodie.
TUNE	Spielt einen oder mehrere Töne.

DIGITALE EINGABE

IF… THEN	Springt zu einer neuen Programmzeile, abhängig von Input Pin.
PULSIN	Misst Pulslänge an einem Input Pin.

ANALOG EINGABE

READADC	Liest Analogwert in eine Variable.
READADC10	Liest Analogwert (10bit-Auflösung) in eine Variable.

PROGRAMMABLAUF

FOR.. NEXT	Bildet eine FOR-NEXT Schleife
BRANCH	Springt an eine Adresse spezifiziert durch einen Offset
GOTO	Springt zu einer Adresse
GOSUB	Springt zu einem Unterprogramm an einer Adresse.
RETURN	Rückkehr aus einem Unterprogramm.
IF.. THEN	Vergleich und bedingter Sprung.

VARIABLE MANIPULATION

{LET}	Zuweisung von Werten an Variablen (math. Operationen).
LOOKUP	Findet Daten spezifiziert durch Offset und speichert sie in Variable.
LOOKDOWN	Findet Daten in einer Liste und speichert Offset in Variable
RANDOM	Erzeugt eine Pseudo-Zufallszahl

SERIELLE EIN-/AUSGABE

SEROUT	Gibt serielle Daten an einen Output Pin. Bis zu 2400 baud.
SERIN	Liest serielle Daten an einem Input pin. Bis zu 2400 baud.
SERTXD	Serielle Ausgabe über Programmierport (4800 Baud)

INTERNAL EEPROM ACCESS

EEPROM	Initialisiert Daten im EEPROM (vor Programmdownload)
READ	Liest Daten aus EEPROM in Variable
WRITE	Schreibt Daten in EEPROM Bereich

INTERNER RAM ZUGRIFF

PEEK	Liest Daten aus RAM in Variable
POKE	Schreibt Daten in RAM Bereich

STROMSPARMODIS

NAP	Stromsparmodus für kurze Zeit (bis zu 2,3 Sekunden)
SLEEP	Stromsparmodus für längere Zeiten (bis zu 65535 Sekunden)
END	Prozessor-Halt bis Reset

INFRAROT INTERFACE

INFRAIN2	Empfangen SONY Infrarot-RC-Codes
INFRAOUT	Senden SONY Infrarot-RC-Codes

1WIRE

READOWSN	Lesen Dallas 1 wire Seriennummer

TEMPERATUR

READTEMP	Lesen Dallas DS18B20 1wire Temperatursignal
READTEMP12	Lesen Dallas DS18B20 1wire Temperatursignal (12bit Auflösung)

VERSCHIEDENES

PAUSE	Pause (max. 65535 Millisekunden)
WAIT	Pause (max. 65 Sekunden, Sekundenauflösung)
SERVO	Ansteuerung RC-Modellbauservo
SETINT	Deklaration Interruptroutine