Begraaf de strijdbijl en graaf de AXE op (3)

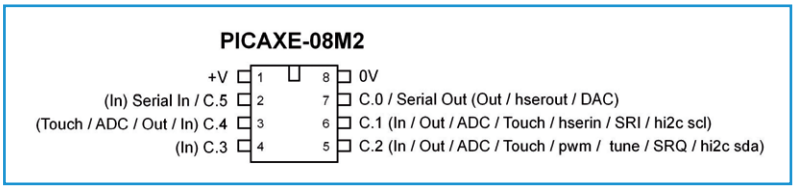
Wouter Spruit
(Nederland)

Over analoge inputs, PWM, servo's en het muzikale talent van de PICAXE

In het eerste deel van deze serie [1] hebben we gezien hoe een PICAXE-chip wordt geprogrammeerd en hoe eenvoudige schakelingen voor input en output zijn te bouwen (Elektor**.**POST project nr. 8). In de tweede aflevering (Elektor**.**POST project nr. 16) behandelden we het besturen van verschillende soorten schakelaars en hoe we geschikte waarden voor de componenten in een schakeling bepalen. Deze keer gaan we kijken hoe we waarden van analoge inputs kunnen lezen en verwerken en beschrijven we verschillende soorten uitvoer met schakelingen die zijn aangesloten op een pulsbreedtegemoduleerde (PWM) output, in combinatie met de al bekende PICAXE-commando's. We gaan ook voorwerpen nauwkeurig positioneren met behulp van door een PICAXE-chip bestuurde servo's. Tenslotte gaan we monofone ringtones afspelen!

Wat hadden we ook alweer nodig?

Net als in de vorige afleveringen gebruiken we bij alle experimenten een PICAXE-08M2-microcontroller (de aansluitgegevens zijn nogmaals weergegeven in **figuur 1**). De schakelingen worden gevoed met een 5 V-ATX-voeding. De voorbeeldschakelingen worden opgebouwd op breadboards, dus zonder solderen. De PICAXE-chips worden geprogrammeerd via een USB-naar-serieel-kabel met de methode die is beschreven in de vorige artikelen [1]. Hierbij wordt LinAXEpad-software versie 1.5.0 voor (Arch)-Linux gebruikt. Om verwarring te voorkomen verwijzen alle pennummers in de schema's naar de fysieke pennummers op de chip (bij alle voorbeelden: pen 5 is een PWM-output, pen 6 is analoge input 1). De pennummers in de listings verwijzen naar interne pennamen volgens figuur 1. Zie de vorige artikelen [1], of de PICAXE-handleiding op de website [2] voor voorbeelden van het programmeren van een PICAXE-chip. PICAXE-chips en randapparatuur zijn verkrijgbaar via de webwinkel van Revolution Education [3].



Figuur 1. PICAXE-08M2 aansluitgegevens

Analoge input

In alle eerdere voorbeelden gebruikten we een binaire (alles-of-niets) schakelaar om het uitgangscircuit te besturen. Maar vaak willen we meer controle over de besturing. De analoog-naar-digitaal-converters (ADC's) van de PICAXE kunnen een analoge spanning, meestal tussen 0 V en +5 V (de voedingsspanning), lezen en die omzetten naar een 8-bits (of, bij sommige PICAXE-chips, 10-bits) digitale waarde. Een potentiometer werkt als

een instelbare spanningsdeler: we kunnen die gebruiken als een analoog invoerapparaat. Verbind de loper (de middelste aansluiting) met een ADC-pen van de PICAXE. De ingangsspanning hangt dan af van de stand van de potmeter.
Als we de loper dicht naar de 0-Ω-kant draaien, moet de stroom door de ADC-pen worden begrensd om schade aan de componenten te voorkomen (zoals besproken in deel 2 [1]). Maar als we gewoon weerstanden toevoegen om de stroom te beperken, heeft dat ook invloed op de gemeten waarde. Een weerstand van 330 Ω tussen de loper en de ADC-pen begrenst de stroom tot:

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

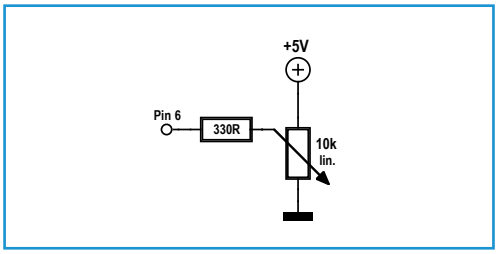
Dat is ruim beneden de maximale stroom van 0,02 A die de pen kan verwerken. In dit voorbeeld gebruiken we een lineaire potmeter van 10 kΩ. De maximale spanning over de ADC wordt dan:

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

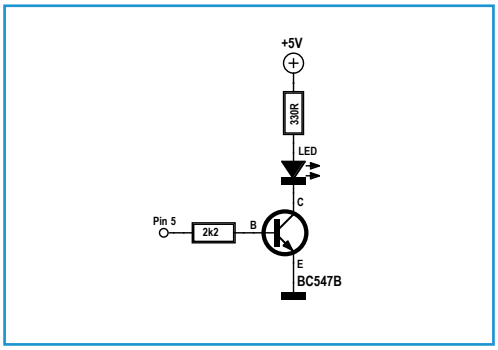
Dit relatief kleine effect kunnen we compenseren in de software. Op de PICAXE-forums [4] zijn discussies te vinden over het kiezen van de weerstand van componenten aan de ADC-input van de PICAXE. Men is op het forum tot de conclusie gekomen dat een waarde van 10 kΩ optimaal is voor een potentiometer aan de ADC-input van een PICAXE (of een vergelijkbare Microchip-PIC).

In het voorbeeld zien we hoe we een LED aan en uit kunnen schakelen met een potmeter. De stand van de loper werkt als een schakelaar. De schakeling is weergegeven in de schema's in **figuur 2** en **figuur 3** (input respectievelijk output). De gebruikte code is te vinden in **listing 1** en in **figuur 4** zien we hoe deze schakeling is op te bouwen op een breadboard.

In dit voorbeeld wordt de LED ingeschakeld als de ADC-waarde tussen 2,5 V en 5 V is; anders wordt de LED uitgeschakeld. Het spreekt vanzelf dat het in dit geval beter was geweest om gewoon een aan/uit-schakelaar te gebruiken, maar in het



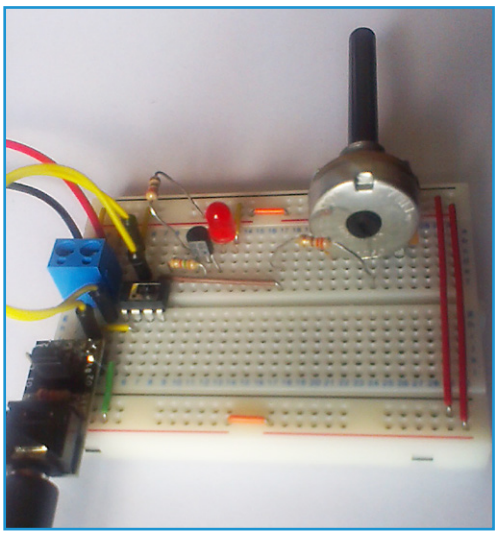
Figuur 2.
Schema voor het gebruik van de analoge input.



Figuur 3.
Schema voor PWM-output.

Listing 1: Analoge Input

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



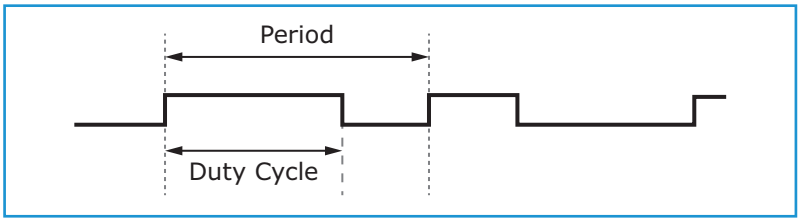
Figuur 4.
PWM-output opgebouwd op een breadboard.

volgende voorbeeld gaan we dezelfde schakeling gebruiken om de helderheid van de LED te regelen. Soms gedragen de ADC-inputs van de PICAXE zich vreemd, vooral als ze worden aangesloten op een impedantie van 20 kΩ of meer. Meer informatie hierover is te vinden op de forums [4] en natuurlijk op Internet.

PWM-basiskennis

Hoewel de PICAXE geen analoge outputs heeft, heeft hij wel één (of meer) PWM-outputs. Met een PWM-output kunnen we verschillende nuttige dingen doen, ook iets dat dicht in de buurt komt van een echte analoge output. Een PWM-output genereert continu een blokgolfsignaal. In tegenstelling tot andere PICAXE-commando's draait het PWM-output-commando op de achtergrond (parallel aan de rest van de uitvoering van het programma). Twee belangrijke eigenschappen van een PWM-signaal zijn de frequentie en de duty-cycle (zie **figuur 5**). De frequentie is het aantal herhalingen per tijdseenheid (meestal per seconde); de periode is de tijdsduur van één cyclus en de duty-cycle is het percentage van de tijd dat de uitgang actief (hoog) is in elke periode. Er bestaat software zoals de Python-bibliotheek RPi.GPIO voor de Raspberry PI [5], waarmee de gebruiker de PWM-frequentie en –duty-cycle rechtstreeks kan besturen, maar bij de PICAXE is dat wat ingewikkelder. Er bestaan commando's voor dit doel, maar het nieuwste en belangrijkste is '`pwmout`'.

Omdat dit commando best ingewikkeld is, gaan we uit van de beschrijving van `pwmout` in [2], in plaats van alles in detail hier uit te leggen. De frequentie van het PWM-signaal is gebaseerd op de klokfrequentie van de PICAXE. De meeste PICAXE-chips kunnen werken met verschillende kloksnelheden, maar het `pwmout`-commando werkt alleen met bepaalde frequenties. Meestal is de standaard klokfrequentie 4 MHz en die is geschikt voor `pwmout`. De eenvoudigste aanroepen van het `pwmout`-commando zijn '`pwmout pin,period,duty cycles`' om het signaal in te schakelen en '`pwmout pin,OFF`' om het weer uit te schakelen. De variabelen `period` en `duty cycles` worden gebruikt om de PWM-frequentie (f_{PWM}), -periode



Figuur 5.
Structuur van een PWM-signaal.

(T_{PWM}) en -duty-cycle (D_{PWM}) in te stellen. Hiervoor gelden de volgende relaties:

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

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

Hierin is `resonator speed` 1/4.000.000 bij een klokfrequentie van 4 MHz.

Let op: het `pwmout`-commando gebruikt dus een tijdsperiode in plaats van een percentage voor het specificeren van de duty-cycle! Gelukkig maakt de PWMout-wizard het ons gemakkelijk. De LinAxePad-software bevat verschillende wizards. Voor PWM is die te vinden onder PICAXE → Wizards → PWMout. In deze wizard kunnen we een PWM-frequentie in Hz en een duty-cycle als percentage (zoals we dat gewend zijn) invoeren. De uitvoer is een `pwmout`-commando compleet met variabelen om een PWM-signaal te produceren, dat de ingevoerde waarden benadert (voor zover de resolutie van het `pwmout`-commando dat mogelijk maakt).
Met PWM kunnen we bijvoorbeeld LED's dimmen of de snelheid van een gelijkspanningsmotor regelen. Een LED houdt zich niet aan de Wet van Ohm (zie het vorige artikel). Om de lichtsterkte te regelen zonder een stroombron (een voeding die zijn uitgangsspanning zó regelt dat de uitgangsstroom constant blijft), kunnen we PWM gebruiken om de LED snel aan en uit te schakelen. Bij een duty-cycle van 50% is de LED maar 50% van de tijd ingeschakeld. De schijnbare lichtsterkte van de LED verandert overeenkomstig. Omdat de frequentie van het PWM-signaal heel hoog is, zien we alleen het gemiddelde resultaat.
De PWM-frequentie voor het dimmen van een LED moet dus hoog genoeg zijn om onzichtbaar te zijn. Maar let wel op de reso-

lutie van het uitgangssignaal: als we een frequentie kiezen die dichtbij de maximum frequentie van de PICAXE ligt, hebben we minder bits beschikbaar om de duty-cycle te veranderen.

In dit voorbeeld gebruiken we een LED om het principe van het regelen van het uitgangssignaal met PWM te laten zien, maar we kunnen op dezelfde manier de snelheid van een gelijkspanningsmotor regelen. De PICAXE kan niet genoeg stroom leveren voor een motor, dus we moeten een transistor gebruiken om de motor aan te sturen, zoals in het vorige PICAXE-artikel is beschreven (zie Elektor.POST project nr. 16 [1]). We hebben het toen gehad over de schakelsnelheid van de transistor. Dat is iets om rekening mee te houden als we PWM gebruiken. Als de transistor niet snel genoeg kan schakelen om de PWM-frequentie bij te houden, wordt het PWM-uitgangssignaal vervormd en daardoor onbruikbaar. Dat is op te lossen door de PWM-frequentie te verlagen of door een snellere transistor te kiezen. Darlington-paren zijn nog langzamer dan losse transistors, zoals we ook in het vorige deel hebben uitgelegd.

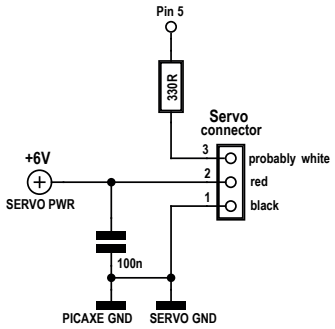
Het schema is hetzelfde als in het vorige voorbeeld (zie figuur 2 en 3), maar dit keer is de software wat ingewikkelder (zie **listing 2**). Het rekenwerk dient voor het aanpassen van het bereik van de inputwaarden (0...255) aan het bereik van de output-waarden (0...100). Het volgende deel maakt gebruik van dit principe.
Het nieuwe programma gebruikt het 8-bits ADC-ingangssignaal van de potmeter om de duty-cycle van het PWM-uitgangssignaal voor de LED te regelen. Nu kunnen we de lichtsterkte van de LED besturen met de potentiometer!

Servo's bewegen

Een servomotor is een mechanische actuator die is bedoeld om bijvoorbeeld een positie te regelen. Een typische servo heeft drie aansluitdraden: zwart voor de massa, rood voor de voedingsspanning en (meestal) wit voor de data. Een servo probeert een stand in te nemen die wordt aangegeven door het signaal op de datalijn. Als het signaal wegvalt, houdt hij zijn positie niet meer vast. Voor het regelen van de positie verwacht een servo een blokgolfsignaal met een periodetijd van 20 ms (frequentie = 1/0,020 = 50 Hz); de duty-cycle (pulsduur van 0,75 ms tot 2,25 ms) bepaalt

Listing 2: PWM Output

```
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
```



Figuur 6.
Schema voor het aansturen van een servomotor.

Listing 3: Servobesturing

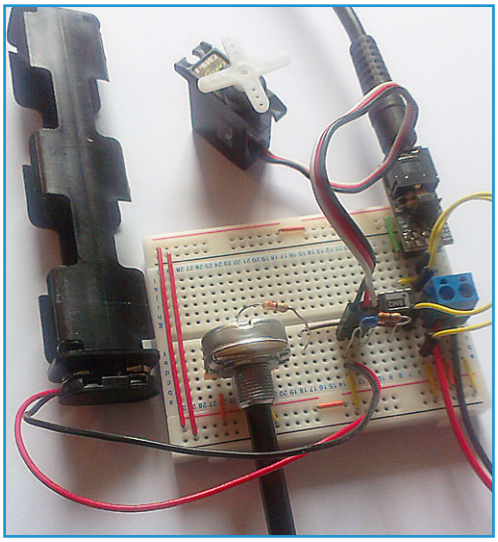
```
init:
servo 2,100   ;init servo
main:
do
  readadc 1,b1  ;get analog input
  if b1 > 250 then  ;clamp value
    b1 = 250
  endif
  w1=b1*3  ;scale value
  w1=w1/5
  b2=b2+75
  servopos 2,b2 ;update servo position
loop
```

de hoek. We hebben geen `pwmout`-commando nodig om dit signaal op te wekken, want daar heeft de PICAXE twee andere commando's voor. Het eerste commando, '`servo pin,pulse`' initialiseert een PWM-output voor gebruik als servo-output. Het tweede commando, '`servopos pin,pulse`', stelt de servopositie in.

Als we de servo aansluiten, verbinden we de datalijn via een stroombegrenzingsweerstand (bijvoorbeeld 330 Ω) met de PWM-uitgang van de PICAXE, maar de servo heeft een aparte voeding nodig. Daar zijn verschillende redenen voor. Een servo kan flink wat stroom trekken en veel storing op de voedingslijn veroorzaken. Daar kan de PICAXE last van hebben als de voeding niet gescheiden is! Bovendien hebben servo's soms een hogere voedingsspanning nodig dan de rest van de elektronica. De 0 V van de PICAXE-voeding en van de servovoeding moeten onderling worden verbonden om te zorgen voor een gemeenschappelijk massaniveau. Tussen servo-V+ en -V0 sluiten we een condensator aan om de voedingsspanning te bufferen. Voor het voeden van de servo gebruiken we in het voorbeeld vier 'AA'-batterijen van 1,5 V (geen accu's, die hebben een lagere nominale spanning van ongeveer 1,2 V). De servo wordt dus gevoed met 6 V.

De waarde van de variabele `pulse` moet liggen tussen 75 en 225, wat overeenkomt met de pulsduur van 0,75 tot 2,25 ms die de servo verwacht.

Een schema voor het aansluiten van een servo is te zien in **figuur 6**. Opnieuw is de analoge input-schakeling hetzelfde als in de vorige voorbeelden (zie figuur 2). De code voor het besturen van de servo met de potmeter aan de analoge input is te vinden in **listing 3**. De opbouw kan er uit zien als in **figuur 7**. De PICAXE-registers zijn 8- of 16-bits integers. We moeten om die begrenzing heen werken om de ADC-input (0...255) om te zetten naar het uitvoerbereik (0...100). In dit geval is het output-waardenbereik 75...225, dat betekent dat we het bereik van de analoge input moeten schalen naar 225 – 75 = 150. Omdat 150/250 = 3/5 kunnen we de waarde schalen door de ADC-waarde te vermenigvuldigen met 3 en daarna te delen door 5. We moeten daarvoor rekenen in 16-bits getallen, omdat 3 × 255 te groot is om weer te geven binnen 8 bits. Tenslotte tellen we nog de minimum waarde van 75 er bij op om de geschaalde output-waarde `b1` te berekenen, in het juiste bereik voor het besturen van de servo, afhankelijk van de waarde aan de loper van de potmeter. De variabele `b2` wordt in de code gebruikt om de laagste 8 bits van w1 aan te duiden, omdat w1 een 16-bits register is dat bestaat uit de twee 8-bits registers b2 en b3, zoals beschreven in het vorige deel [1].



Figuur 7.
Opbouw van de servomotorbesturing op een breadboard.

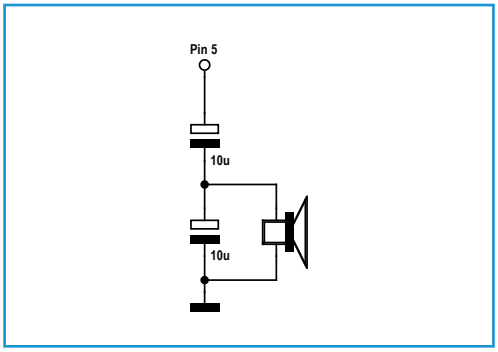
Nauwkeurig besturen van servostanden is heel nuttig bij het bouwen van robots, maar robotica-projecten hebben waarschijnlijk veel meer servo's nodig dan er PWM-outputs op één PICAXE-chip zitten. Dit probleem kan bijvoorbeeld worden opgelost met externe servobesturingschips. De PICAXE kan die chips dan aansturen en de servobesturingskaart zou de positie van meer dan 20 servo's per kaart kunnen bewaren en regelen. De AXE031 kan bijvoorbeeld maximaal 21 kanalen besturen [3]. In het volgende artikel zullen we bespreken hoe we de PICAXE kunnen laten 'praten' met randapparatuur.

De zingende PICAXE: genereren van melodieën en klanken

Een andere ingebouwde functionaliteit van de PICAXE die met PWM werkt is het

'tune'-commando. Het syntax van het commando verschilt per versie van de chip, omdat het aantal PWM-outputs verschillend is. We gebruiken hier de versie die geschikt is voor 8-pens PICAXE-chips. Het tune-commando speelt een monofone melodie van piepjes af, zoals ringtones op ouderwetse mobiele telefoons. Het commando heeft ook functionaliteit om andere outputs te schakelen (bedoeld om LED's te laten knipperen in de maat van de muziek). Om tune te gebruiken is natuurlijk een PWM-output nodig en in het geval van de PICAXE-08M2is er maar één beschikbaar. Er is ook een commando 'play' dat één van de voorgeprogrammeerde liedjes speelt. Deel 2 van de PICAXE-handleiding [6] bevat uitgebreide documentatie over het tune-commando en ook voorbeeldschakelingen om het te gebruiken met een piëzo-elektrische buzzer, een luidspreker en met een externe audioversterker. De auteur gebruikte bij het voorbeeld met tune een oude koptelefoon. Bij het schema voor het aansluiten van een speaker in de handleiding wordt aangeraden een luidspreker van 40...80 Ω te gebruiken. Gebruik bij speakers met een lagere impedantie een serieweerstand, zodat de totale belasting tenminste 40 Ω is.

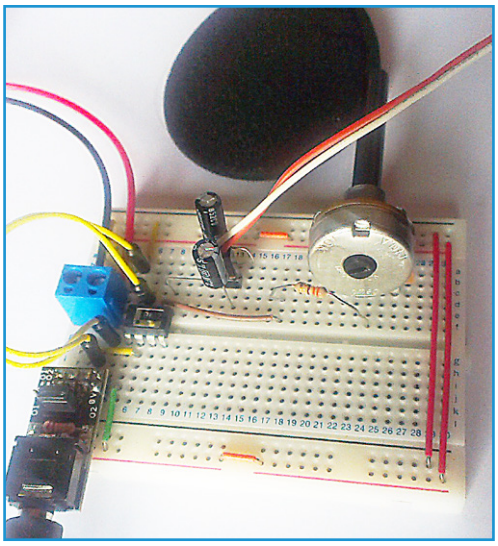
Er is ook een wizard beschikbaar (onder PICAXE → Wizards → Ring Tone Tunes) voor het aanmaken van melodieën voor het tune-commando (en voor het importeren van ringtones in RTTTL-formaat, bijvoorbeeld van de site [7]). Met dit commando kunnen we snel en gemakkelijk liedjes programmeren, maar we kunnen ook het pwmout-commando gebruiken om met de hand liedjes en geluidseffecten te maken net als in ouderwetse computerspelletjes. Hint: verander de PWM-parameters snel in een for-lus met verschillende STEP-parameters. For-lussen kunnen ook genest worden. Het schema voor het aansluiten van een luidspreker (uit deel 2 van de PICAXE-handleiding [6]) is te zien in **figuur 8**. **Listing 4** bevat de broncode voor het spelen van één liedje. Voer dit heel zorgvuldig in. Alle code moet op één regel staan om het programma goed te laten werken. De opbouw van dit experiment is te zien in **figuur 9**. Het programma in de listing speelt een eenvoudige demonstratiemelodie. Natuurlijk kunnen we ook experimenteren met eigen liedjes en geluiden in het tune- (of pwmout)-commando. Voor dit voorbeeld hebben we de input-schakeling niet nodig. Verwijder hem nog niet, we gaan hem binnenkort weer gebruiken.



Figuur 8.
Schema voor het genereren van geluid.

Listing 4: Genereren van een melodie (één regel)

```
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)
```



Figuur 9.
Opbouw van de luidspreker (oude koptelefoon) besturing op een breadboard.

PWM-output omzetten naar analoog

De PICAXE heeft alleen digitale outputs. Maar het is wel mogelijk om een analoge spanning te genereren als we PWM gebruiken in combinatie met een laagdoorlaatfilter. In de vorige delen [1] hebben we het

principe van de spanningsdeler besproken. Als we een spanning aansluiten op een aantal weerstanden in serie, is de spanning over elk van de individuele weerstanden evenredig met de weerstand. Als we een PWM-(blok)golf aansluiten op een laagdoorlaatfilter gebeurt er min of meer hetzelfde. We vervangen de tweede weerstand in de spanningsdeler door een condensator. De spanning over de condensator zal dus evenredig zijn met de duty-cycle van het PWM-signaal. Maar dat werkt alleen als de PWM-frequentie veel hoger is dan de afsnijfrequentie van het laagdoorlaatfilter. De formule voor het berekenen van de afsnijfrequentie is:

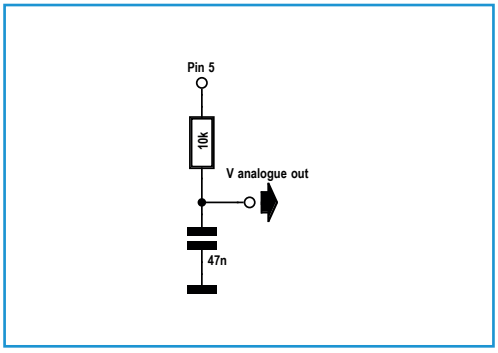
$$f_C = \frac{1}{2\pi\ RC}$$

Een grotere RC-waarde geeft een gladdere gelijkspanning aan de output, maar maakt de schakeling ook langzamer: Als het PWM-frequentie verandert, duurt het langer voordat het analoge uitgangssignaal verandert. Om de componenten te kiezen berekenen we eerst de weerstandswaarde om de stroom te beperken en dan kiezen we een waarde voor C met behulp van de filterformule:

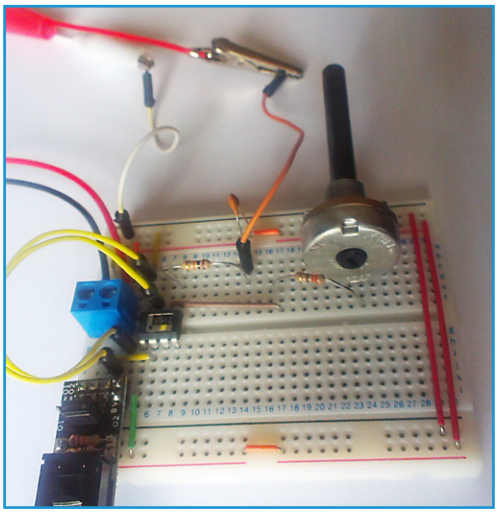
$$C = \frac{factor}{2\pi\ R\ f}$$

Hierin is 'factor' een vermenigvuldigingsfactor tussen de afsnijfrequentie van het filter en de PWM-frequentie. Er is een website [8] waar we de effecten van de PWM-frequentie en de RC-waarde kunnen visualiseren met behulp van een simulatie. Zo kunnen we laten zien hoe het analoge uitgangssignaal verandert bij verschillende parameters in het PWM-signaal en in de schakeling. Bij het berekenen van de RC-waarde is het verstandig tenminste een factor 100 te kiezen voor een uitgangssignaal met weinig rimpelspanning.

We gebruiken niet de hoogst beschikbare PWM-frequentie, omdat de PICAXE bij hogere PWM-frequenties de duty-cycle niet meer zo nauwkeurig kan regelen. We kiezen liever voor een frequentie van 4 MHz / 100 = 40 kHz om de duty-cycle te besturen in 100 stappen. Dit is heel handig, want nu is de parameter `duty cycles`



Figuur 10.
Schema voor analoge output.



Figuur 11.
Analoge output opgebouwd op een breadboard.

die we opgeven in het `pwmout`-commando precies gelijk aan het duty-cycle-percentage van het output-signaal! Bij een PWM-frequentie van 40 kHz en een waarde van 10 kΩ voor R (als stroombegrenzing) vinden we voor C:

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

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

De meest nabije standaard condensatorwaarde is dan 47 nF. In dit geval wordt het PWM-signaal niet veranderd, dus we kunnen een (veel) grotere waarde voor C kiezen om een gladder uitgangssignaal te krijgen zonder bijeffecten.

Dit uitgangscircuit is te zien in **figuur 10**; de input-schakeling blijft hetzelfde (figuur 2). We gebruiken dezelfde code als

in listing 2, want de PWM-frequentie is al 40 kHz, en de juiste code voor het schalen is ook al aanwezig. De opbouw op het breadboard is te zien in **figuur 11**.
Het analoge ingangssignaal wordt gebruikt om de analoge uitgangsspanning te regelen met behulp van de duty-cycle van het PWM-signaal. De 'gladheid' van het uitgangssignaal kan met een oscilloscoop zichtbaar gemaakt worden. Als het uitgangssignaal vloeiend genoeg is, lukt het ook met een gewone voltmeter. We zouden het uitgangssignaal rechtstreeks naar de analoge input van de PICAXE kunnen voeren, om het te meten met de ADC van de PICAXE, maar op dit moment hebben we nog geen manier om dat resultaat in beeld te brengen.

Samenvatting

Naast het programmeren van een PICAXE-chip en het aansturen van eenvoudige elektronica kunnen we nu ook analoge inputs gebruiken en we hebben meer controle gekregen over de outputs. Met de PWM-output kunnen we een analoog outputsignaal genereren, een gedimde output ontwerpen en zelfs op ouderwetse manier geluid genereren. Verder hebben we geleerd de positie van een servo nauwkeurig te besturen. Maar bij het laatste voorbeeld kwamen we een probleem tegen: We kunnen nog geen data weergeven. Dat zou heel mooi zijn als onderdeel van de gebruikersinterface van geavanceerde PICAXE-programma's, of voor het visualiseren van data van de (analoge) input. Maar geen paniek! In de volgende aflevering over de PICAXE in Elektor.POST laten we zien hoe de PICAXE kan communiceren met verschillende randapparatuur. Dat is een geweldige uitbreiding van de mogelijkheden van PICAXE-projecten. Eén daarvan is een OLED-display.

(130262)

Weblinks

[1] "Begraaf de strijdbijl en graaf de AXE op", (1) en (2), Elektor.POST Projecten 8 en 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