CE

Programmeerbare besturing

Microcontrollers komen we overal tegen: in huishoudelijke apparatuur, in consumentenelektronica, in voertuigen, in meetapparatuur en zelfs in onbemande ruimteschepen. Ze doen overal dingen die door een programma worden opgedragen. Het is spannend om ook zelf eens een eenvoudig besturingsprogramma te schrijven.

De eerste stap is altijd een microcontroller of processor te zoeken die zo goed mogelijk bij de gewenste opgave past. Men heeft de keuze uit een onnoemelijk aantal types van verschillende fabrikanten en is keuze uit verschillende programmeertalen. Meestal wordt er gebruik gemaakt van Assembler of C, vaak ook van Basic of een andere hogere programmeertaal. Normaal gesproken heeft men voor het programmeren omvangrijke software en een programmeertoestel nodig. Naast de kosten moet ook de inwerktijd niet worden vergeten.

Bij de hier gebruikte microcontroller wordt het volledig anders aangepakt. Voor het programmeren zijn niet meer dan twee schakelaars nodig. De programmeerbare besturing (TPS) kent slechts een gering aantal instructies die eenvoudig te leren zijn en met behulp van de schakelaars in de controller worden geprogrammeerd. Het programma kan altijd zonder speciale hulpmiddelen worden gewijzigd.

Het systeem is bijzonder geschikt voor compacte toepassingen op het gebied van meten, besturen en regelen. Met dit systeem is al een groot aantal taken volledig op te lossen. Na de succesvolle programmering kunt u de microcontroller bovendien in uw eigen schakelingen inbouwen. Er wordt enige basiskennis op het gebied van elektronica verwacht.

Daarnaast is het systeem ook geschikt als basis voor een opleiding en voor de eerste kennismaking met de programmering van microcontrollers. U heeft met dit systeem eerder succes dan met andere systemen. De taalstructuur is vergelijkbaar met andere programmeertalen zodat u later eenvoudig met een andere taal verder kunt gaan.

Inhoudsopgave

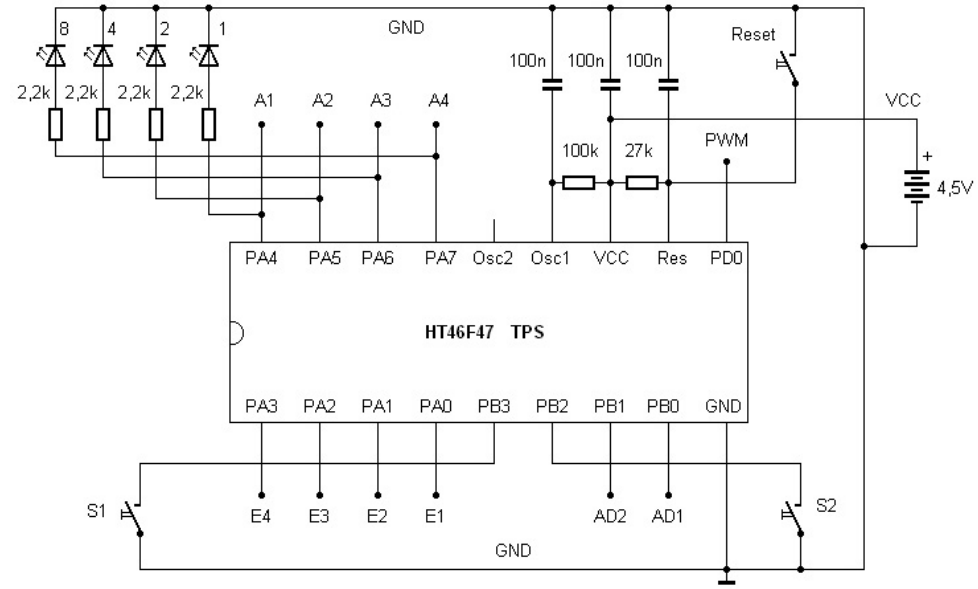
1 Inleiding

Het principe van de TPS controller is eenvoudig. Er zijn vier digitale ingangen E1 t/m E4 en vier digitale uitgangen A1 t/m A4. Daarnaast zijn er twee analoge ingangen AD1 en AD2 en een quasi-analoge PWM-uitgang. Op de reset-ingang kan een reset-toets worden aangesloten om het programma aan het begin te resetten. De controller wordt met behulp van drie AA-batterijen voorzien van een spanning van ca. 4,5 V en kan werken in een bereik van 2,2 V tot 5,5 V.

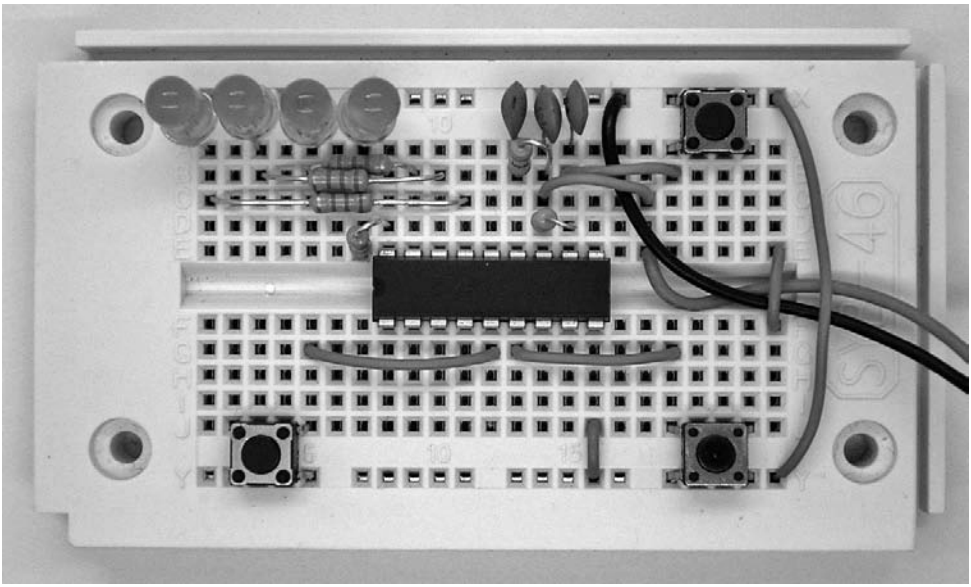
Technische specificaties:
Microcontroller: HT46F47
Pulsfrequentie: 2 MHz
Interne EEPROM: 128 Bytes
Voeding: 2,2 V tot 5,5 V
Stroomverbruik: 1 mA bij 4,5 V
4 uitgangen: belastbaar tot 10 mA
1 PWM-uitgang: belastbaar tot 10 mA
4 ingangen: rusttoestand 1
2 analoge ingangen: 0 V ... VCC
2 toetsaansluitingen: rusttoestand 1

Onderdelen van het leerpakket:
Insteekbord
Batterijhouder 3 * AA
Draad
HT46F47 met TPS firmware
3 schakeltoetsen
4 LED's (5 mm)
1 LED 5 mm, groen
1 LDR
3 schijfcondensatoren 100 nF
1 Elco 47 µF
5 weerstanden 2,2 kOhm
1 weerstand 10 kOhm
1 weerstand 27 kOhm
2 weerstanden 100 kOhm

Voor het programmeren zijn de beide toetsen S1 en S2 benodigd en een eenvoudige LED-display bestaande uit vier LED's aan de uitgangen A1 t/m A4. Er zijn in totaal 14 eenvoudige instructies met de bijbehorende gegevens of subinstructies. Instructies en gegevens bestaan uit 4-bit binaire getallen met een bereik van 0000 tot 1111 (decimaal 0 tot 15) gecodeerd en zijn direct zichtbaar op de LED-display. Het resp. getal wordt tijdens het programmeren geprogrammeerd door op de toets S1 te drukken. Met S2 wordt omgeschakeld tussen instructies en gegevens en wordt het adres van de opdrachtregel opgehoogd. De volledige programmastructuur is zo eenvoudig dat u deze met een beetje oefening uit het hoofd kunt leren.



Afb. 1: basisschakeling van het systeem



Afb. 2: standaardopbouw met schakeltoetsen

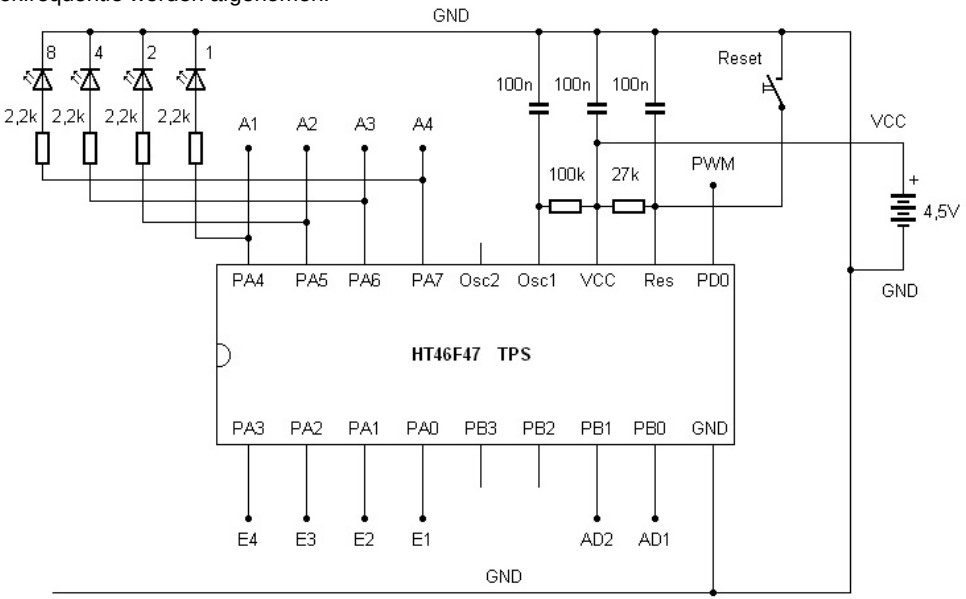
Bij levering zijn al enkele standaardprogramma's (default) in de TPS controller opgeslagen die direct kunnen worden gestart. Daarom is mogelijk om de controller stap voor stap in gebruik te nemen. Zorg dat u eerst vertrouwd raakt met de hardwarefuncties en begin daarna pas met eigen programma's.

Bij de eerste tests worden kleine programma's gestart die al kant en klaar in de controller zijn opgeslagen. De bijbehorende programmalistings geven een eerste indruk van de mogelijkheden. Deze worden slechts kort uitgelegd. De exacte uitleg van de verschillende instructies volgt in de onderstaande hoofdstukken.

Bouw voor het eerste experiment alleen de basisconfiguratie met de controller en de benodigde aanvullende componenten op het insteekboard op. In elk geval zijn nodig:

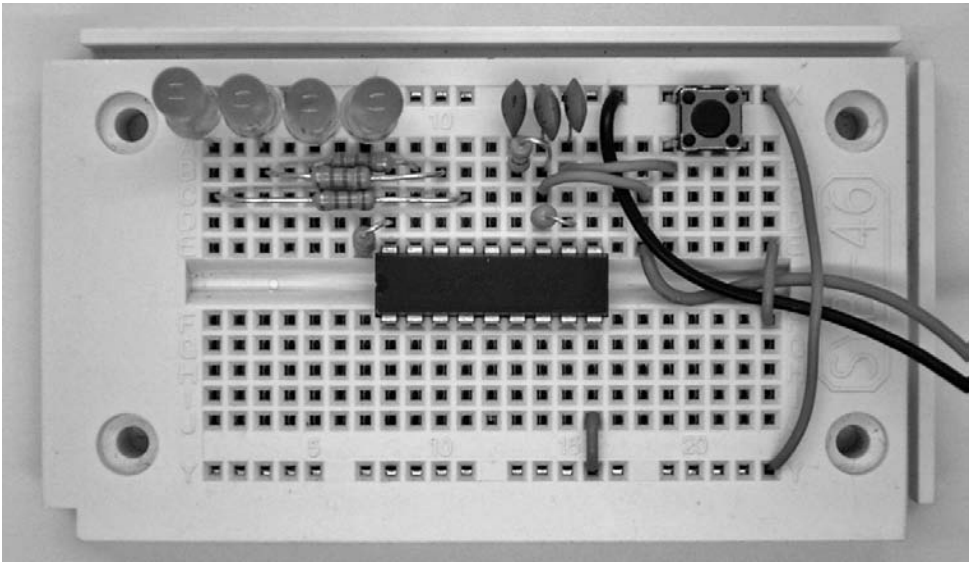
- Aansluiting voor de voeding aan GND (min) en VCC (plus)
- Een brugcondensator van 100 nF tussen VCC en GND
- Een resetweerstand achter VCC en een resetcondensator achter GND
- Een oscillatorweerstand van 100 kOhm achter VCC en een condensator achter GND

De microcontroller HT46F47 werkt met de ingebouwde RC-oscillator. Met de weerstand aan Osc1- wordt de klokfrequentie ingesteld. Met een weerstand van 100 kOhm wordt een frequentie ingesteld van ca. 2 MHz. Indien gewenst kan met een lagere of een hogere kloksnelheid worden gewerkt. De aangesloten condensator dient uitsluitend ter afscherming en heeft geen invloed op de klokfrequentie. De aansluiting Osc2 wordt niet gebruikt. Indien gewenst kan hier echter een extra weerstand aan VCC worden aangesloten en kan een pulssignaal met een kwart van de klokfrequentie worden afgenomen.



Afb. 3: vier LED's aan de uitgangen

Gebruik de bovenste en de onderste spanningsrail op de insteekprintplaat voor de massa-aansluiting GND. Hierop wordt de zwarte kabel van de batterijhouder, dus de minpool, aangesloten. De plusaansluiting VCC wordt met de rode aansluitkabel van de batterijhouder verbonden. Verkeerde polariteit moet absoluut worden voorkomen omdat de controller daardoor onherstelbaar kan worden beschadigd. Bouw een kort stukje draad in als trekontlasting. Wanneer de voeding eenmaal is aangesloten moet deze altijd verbonden blijven. Verwijder een van de batterijen uit de houder om het systeem uit te schakelen.



Afb. 4: minimale schakeling met LED's

Bouw alvast de resetschakelaar in en sluit vier LED's aan met een voorschakelweerstand van 2,2 kOhm. Deze zijn benodigd voor een eerste test van de hardware. Let op de juiste volgorde: A1 wordt op de linker LED aangesloten en A4 op de rechter. Op deze manier ontstaat een binaire weergave met het hoogste bit (most significant bit MSB) links. Dat is vooral handig bij de latere programmering.

2 Wisselend knipperlicht

Plaats nu drie 1,5 V batterijen of drie NiMh accu's in de batterijhouder. Daarmee start u het eerste voorbeeldprogramma met een wisselend knipperlicht met de linker en de rechter LED. De knipperfrequentie is ca. 1 Hz. De programmalisting toont het eenvoudige programma met slechts vijf regels. Afwisselend worden de LED's 1 en 8 ingeschakeld. Tussen het inschakelen worden pauzeopdrachten uitgevoerd met een wachttijd van 0,5 s. De terugsprong naar het begin zorgt er voor dat het knipperen eindeloos wordt uitgevoerd. De verschillende instructies worden hieronder gedetailleerd uitgelegd. U kunt echter aan dit voorbeeld al de eenvoud van de programmering zien. In de firmware van de controller is een interpreter ingebouwd die de eenvoudige instructies herkend en uitvoert. De programma's worden daardoor veel compacter dan bij andere systemen.

Het voorbeeld neemt het adresbereik vanaf 20h (decimaal 32) in. Er kunnen meerdere programma's in het hogere adresbereik later ook vanuit uw eigen toepassingen worden gestart. De adressen kunnen alternatief ook met eigen programmacode worden overschreven. Indien nodig kan de controller ook weer in de originele toestand worden gezet waarbij de oorspronkelijke

voorbeeldprogramma's weer worden hersteld.

Adres	Instructie	Gegevens	Commentaar
20	1	1	LED 1
21	2	8	Wacht 500 ms
22	1	8	LED 8
23	2	8	Wacht 500 ms
24	3	4	Spring –4

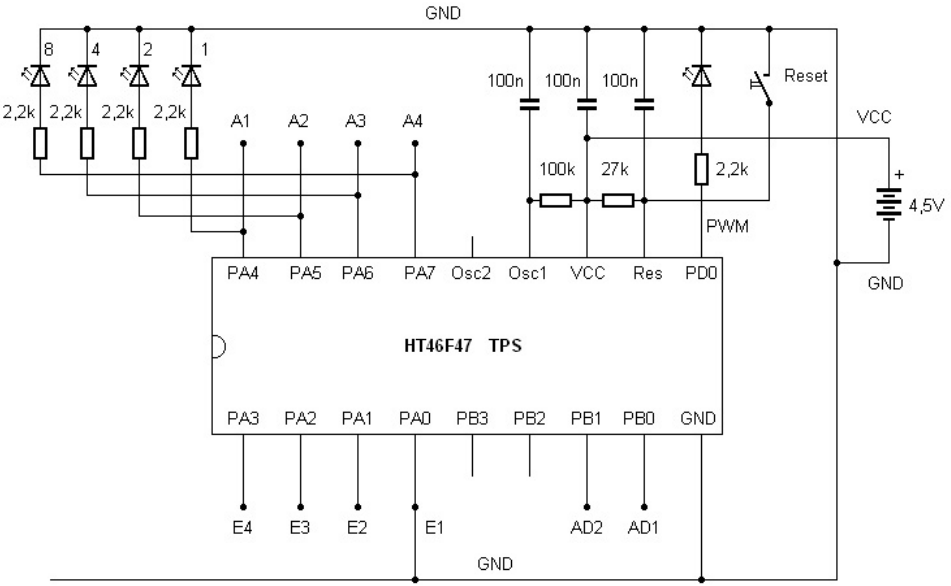
Listing 1: wisselend knipperlicht

Wanneer u niet het gewenste resultaat krijgt moet u eerst de juiste polariteit van de LED's controleren. Het is ook handig om de verschillende spanningen te meten. Maak bv. gebruik van een digitale multimeter met een bereik tot 20 V en laat de minpool aangesloten op GND. Op die manier worden alle spanningen tegen GND gemeten:

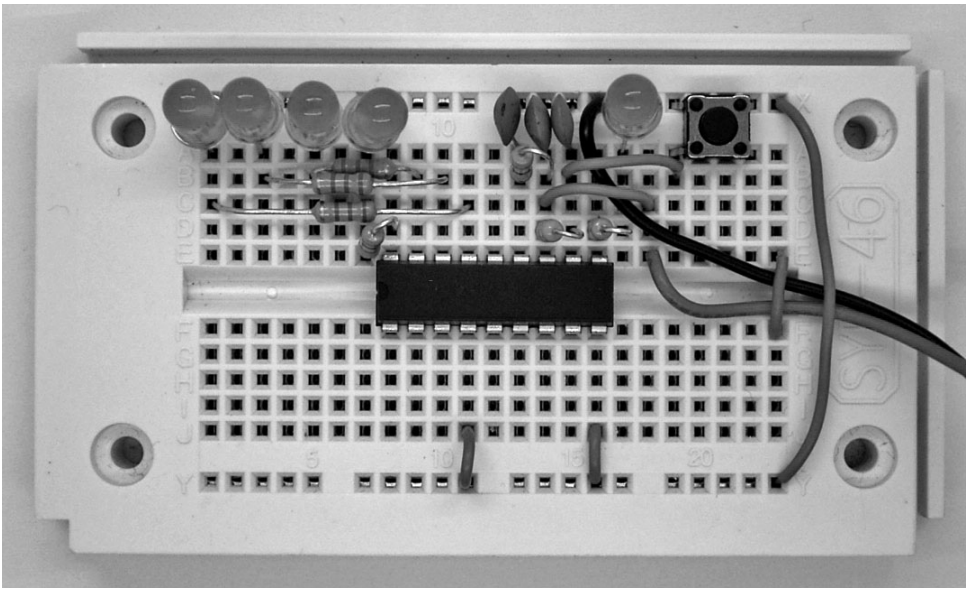
VCC: 4,5 V
Reset: 4,5 V
Osc1: 1,5 V
E1 t/m E4: 4,5 V
A1: wisselend
A2, A3: 0 V
A3: wisselend

3 Binaire teller en PWM-uitvoer

Alle digitale ingangen zijn met behulp van een interne weerstand (pullup weerstand) aan VCC ingesteld en hebben een rustspanning van dezelfde grootte als de voedingsspanning. U kunt echter elk van de ingangen met een draad of contact aan GND leggen. Bij de start leest het standaard programma de poort uit en evalueert de toestand. De verschillende aansluitingen kunnen aan GND worden gelegd zodat hier de nultoestand worden uitgelezen. Afhankelijk van het resultaat kunnen verschillende programma's worden opgeroepen.



Afb. 5: gebruik van de PWM-LED



Afb. 6: starten van de binaire teller

Schakel E1 aan GND. Hiermee start, na een reset, het tweede voorbeeldprogramma. Dit programma telt de toestand van de uitgang binair omhoog. De toestanden 0000 (decimaal 0) tot 1111 (decimaal 15) worden doorlopen. Het programma gebruikt de variabele A voor een eenvoudige optelling en weergave aan de digitale uitgangen en de PWM-uitgang. De instructies 7 en 5 bezitten subfuncties die als gegevens worden geschreven.

Adres	Instructie	Gegevens	Commentaar
25	7	1	A = A + 1
26	5	4	Port = A
27	5	9	PWM = A
28	2	6	Wacht 100 ms
29	3	4	Spring –4

Listing 2: binaire teller met LED en PWM uitvoer.

Het telprogramma kan worden gebruikt voor het lezen van binaire getallen die u moet beheersen wanneer u zelf gaat programmeren. Elk van de vier LED's stelt een bit voor In totaal kan daarmee een 4-bits getal worden weergegeven. De LED's worden in het schema aangeduid met de positiewaarden 8. 4, 2 en 1. Door het optellen van de verschillende waarden kunt u de decimale waarde bepalen. In hexadecimale notatie worden de getallen 10 tot 15 weergegeven met de letters A t/m F.

8	4	2	1	Decimaal	Hexadecimaal
0	0	0	0	0	0
0	0	0	1	1	1
0	0	1	0	2	2
0	0	1	1	3	3
0	1	0	0	4	4
0	1	0	1	5	5
0	1	1	0	6	6
0	1	1	1	7	7
1	0	0	0	8	8
1	0	0	1	9	9
1	0	1	0	10	A
1	0	1	1	11	B
1	1	0	0	12	C
1	1	0	1	13	D
1	1	1	0	14	E
1	1	1	1	15	F

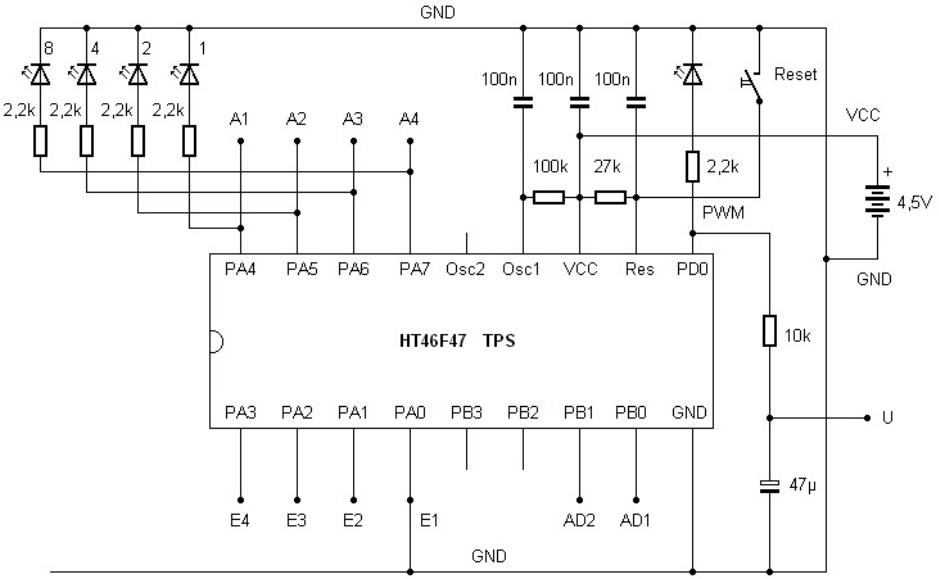
Met dit programma kan het knipperlicht ook met verschillende frequenties worden toegepast.

De volgende hogere uitgang heeft resp. de halve frequentie of de dubbele periode:
A1: 200 ms

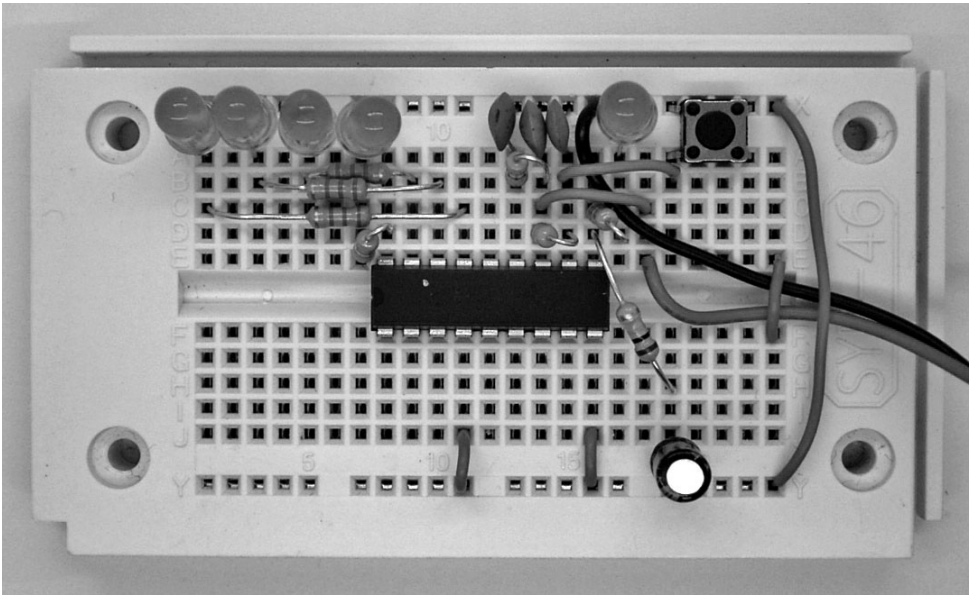
A2: 400 ms
A3: 800 ms
A4: 1.600 ms

Daarnaast worden de oplopende getalswaarden ook aan de PWM-uitgang (Puls Width Modulation: Pulsbreedte modulatie) uitgevoerd. Het PWM-signaal is een blokvormig signaal met een frequentie van ca. 16 kHz. De pulslengte wordt daarbij zo gestuurd dat de verhouding tussen puls en pauze de gemiddelde inschakelduur en daardoor de helderheid van de LED bepaald. De helderheid van de hier aangesloten LED wordt in 15 stappen tussen nul en de maximale helderheid aangestuurd.

Het PWM-signaal kan met behulp van een RC-laagdoorlaatfilter worden afgevlakt tot een gelijkspanning. De PWM-uitgang wordt op die manier een analoge uitgang. Met dit programma krijgt u een stapsgewijs oplopende gelijkspanning van 9 V tot 4,5 V. U kunt het spanningsverloop volgen met een multimeter of een oscilloscoop.



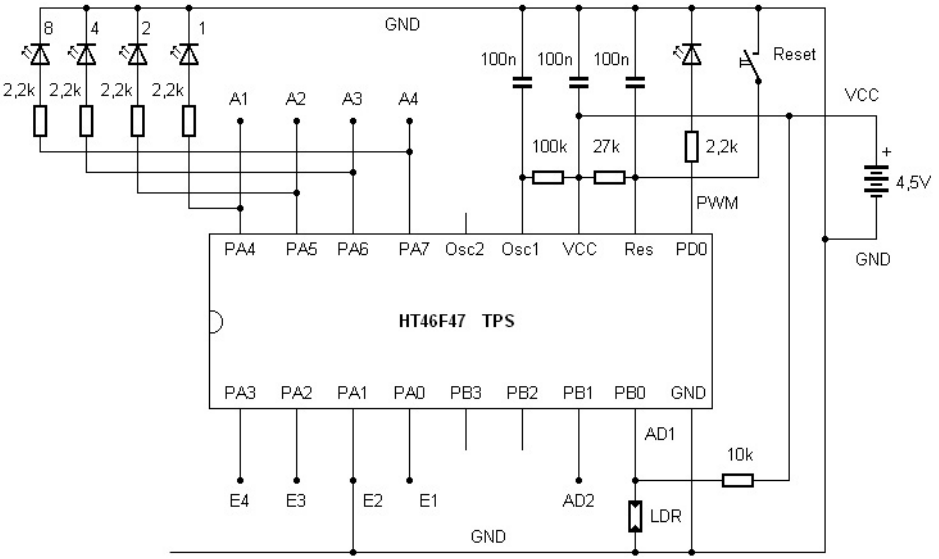
Afb. 7: laagdoorlaatfilter op de PWM-uitgang



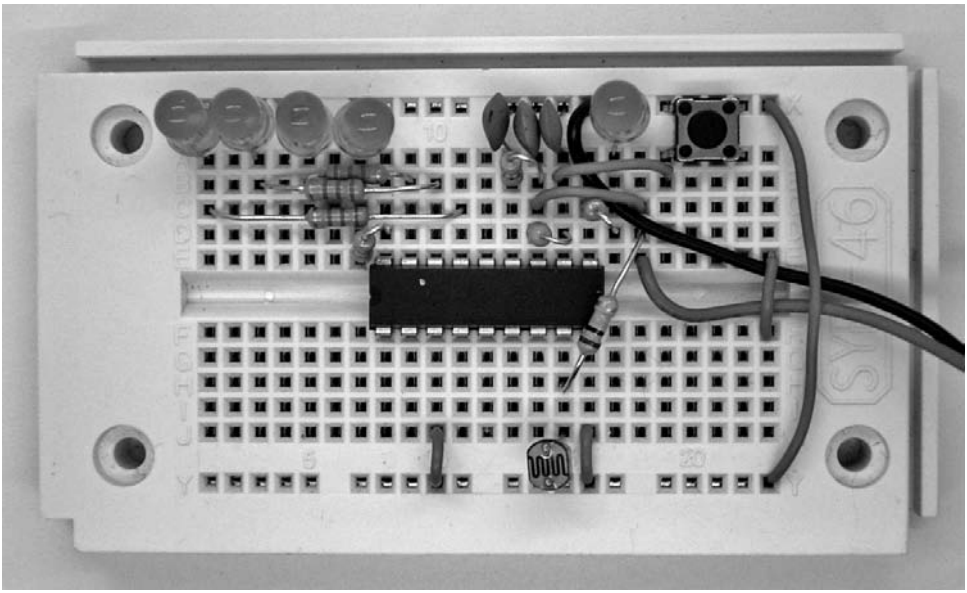
Afb. 8: afgevlakte PWM gelijkspanning

4 Analoog/digitaal omzetter

Met een verbinding E2 aan GND en een druk op de reset-toets start een klein voorbeeldprogramma met een analoog/digitaal omzetter (A/D-omzetter). De analoge spanning aan de analoge ingang AD1 wordt gemeten en omgezet in een digitale cijferwaarde. Omdat de TPS-controller met 4-bits waarden werkt, is het resultaat van de analoog/digitaal omzetting een getal in het bereik van 1 tot 15. Het resultaat 0 staat voor een ingangsspanning van 0 V, het resultaat 15 voor een spanning die overeenkomt met de voedingsspanning, dus bv. 4,5 V. De A/D-waarde wordt als binair getal weergegeven op de vier LED's en daarnaast op de PWM-uitgang gezet. Sluit een spanningsdeler, bestaande uit een vaste weerstand en een lichtafhankelijke weerstand (LDR) aan op de analoge ingang AD1.



Afb. 9: aansluiten van de lichtsensor



Afb. 10: de LDR aan de AD1-ingang

Het voorbeeldprogramma heeft vanwege de uitvoer aan de digitale uitgangen en de PWM-uitgang grote overeenkomst met het programma uit het voorgaande hoofdstuk. Op de eerste regel staat echter de instructie om een analoge waarde om te zetten.

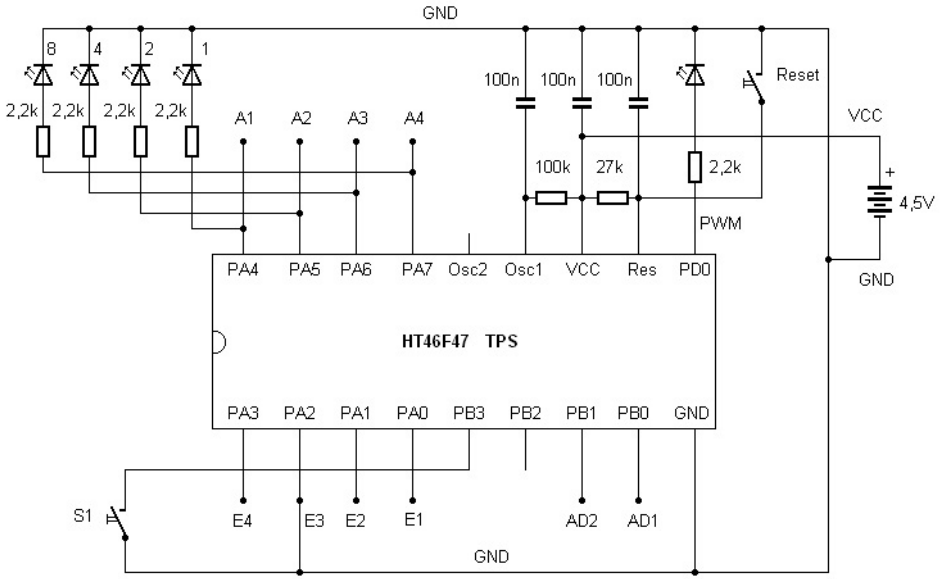
Adres	Instructie	Gegevens	Commentaar
2A	6	9	A = AD1
2B	5	4	Port = A
2C	5	9	PWM = A
2D	2	6	Wacht 100 ms
2E	3	4	Spring –4

Listing 3: AD-omzetter en PWM-uitgang

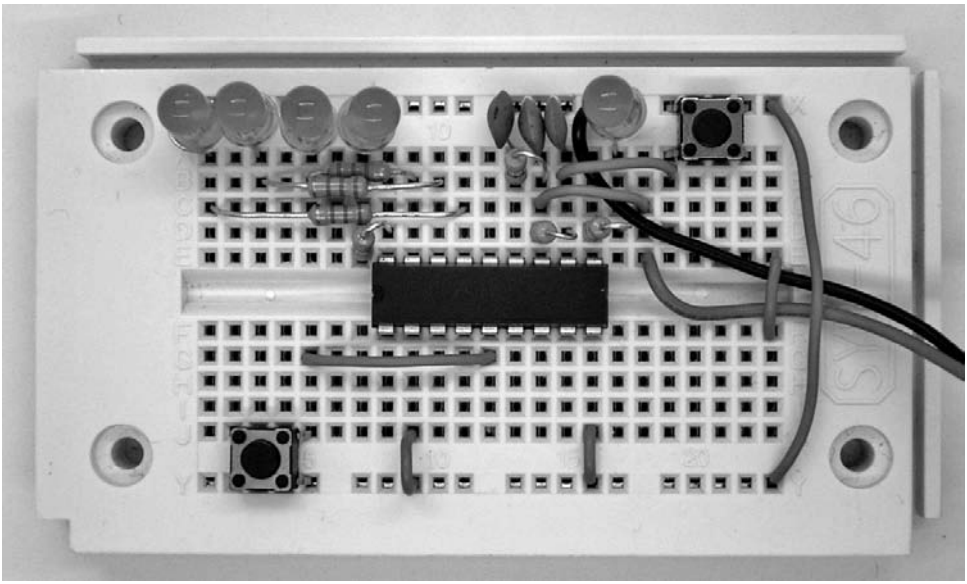
Test het programma met verschillende belichtingen van de sensor. Hoe meer licht er op de LDR valt, hoe lager de spanning aan AD1. Omgekeerd ontstaat er bij duisternis een maximale AD-waarde en daarmee een maximale helderheid van de LED aan de PWM-uitgang. Lees de binaire getallen af van de LED-display en probeer bv. de helderheid op precies de helft van het bereik in te stellen. De digitale waarde ligt dan bij 0111 of 1000. Bij enigszins schommelend kunstlicht kan het voorkomen dat het resultaat tussen twee niveaus heen en weer springt.

5 Toevalsgenerator

Met een draadbrug E3 aan GND start u het voorbeeldprogramma voor een toevalsgenerator. Hier wordt de toestand van de toets S1 uitgelezen. De bijbehorende ingang is voorzien van een interne pullup weerstand die de spanning op het VCC niveau zet. De toets is aan massa aangesloten. De druk op de toets zet de ingang S1 op nul.



Afb. 11: starten van de toevalsgenerator



Afb. 12: draadbrug tussen E3 en GND

Het programma maakt gebruik van een voorwaardelijke sprongopdracht. Wanneer de ingangstoestand van S1, één is, wordt de volgende instructie overgeslagen. Wanneer u dan op de toets drukt, is de toestand nul waardoor de variabele A met één wordt opgehoogd. Dit leidt tot sneller ophogen van de uitgangstoestand. Wanneer u de toets loslat blijft de laatste tellerstand staan. Vanwege de hoge telsnelheid, heeft u geen invloed op het resultaat zodat de uitkomst toevallig is.

Adres	Instructie	Gegevens	Commentaar
30	5	4	Port = A
31	C	E	S1 = 1?
32	7	1	A = A + 1
33	3	3	Spring –3

Listing 4: toevalsgenerator

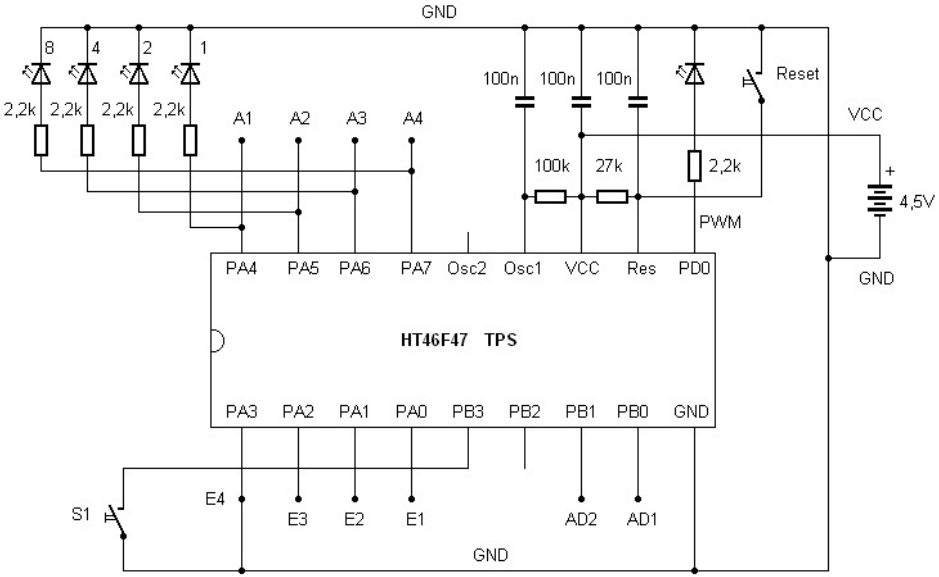
Druk kort op de toets om een nieuw toevallig resultaat te krijgen. Test de toevalsfunctie door een statistiek van de resultaten bij te houden. Na een voldoende groot aantal experimenten zou elk resultaat ongeveer even vaak voor moeten komen. Het programma is bijvoorbeeld geschikt als spel waarbij bv. het getal 1111 moet worden "gegooid".

Het programma is tegelijkertijd ook een teller met de maximaal mogelijke werksnelheid omdat er geen gebruik wordt gemaakt van een wachtinstructie. U kunt hiermee bijvoorbeeld de werksnelheid van de TPS controller onderzoeken. Zolang de toets is ingedrukt, staat er op de uitgang A1 een blokvormig signaal met een frequentie van ca. 133 Hz en een periode van 7,5 ms. De poort wijzigt dus na steeds ca. 3,75 s de toestand. Het programma voert vier instructies uit in een tellus. Voor elke instructie is dus ongeveer een milliseconde nodig. De laatste uitgang A4 geeft een frequentie weer van 16,6 Hz wat nog zichtbaar is als knipperen.

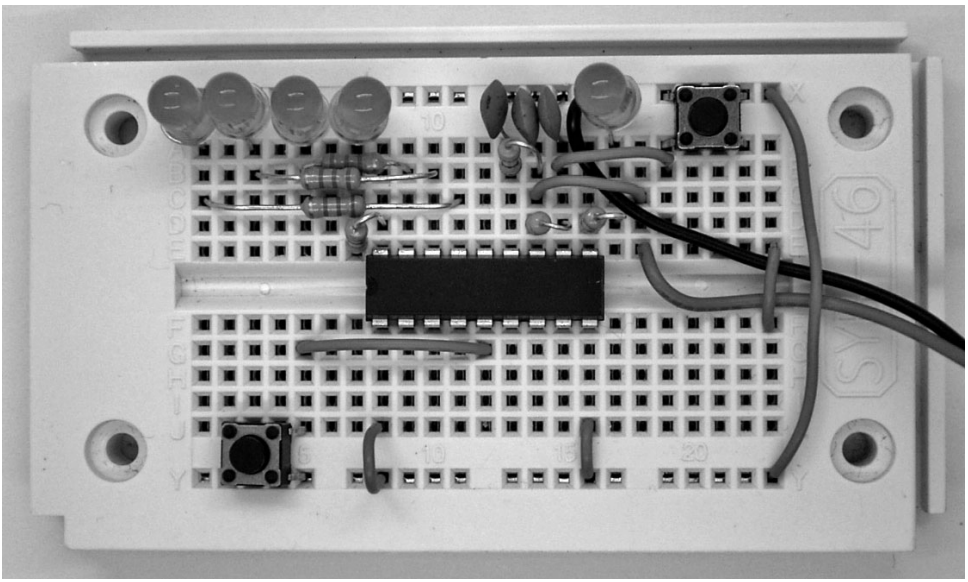
Wanneer er voor tijdkritische toepassingen eens een hogere werksnelheid benodigd is, kunt u de kloksnelheid van de controller verhogen door de weerstand aan Osc1 te verhogen. Met 100 kOhm ontstaat een klokfrequentie van 2 MHz. Vervang de weerstand door een weerstand van 27 kOhm. Hiermee heeft u een bijna 4 maal zo hoge kloksnelheid en een uitvoeringstijd per instructie van ca. 0,25 ms. Onder normale omstandigheden moet de controller echter met een weerstand van 100 kOhm aan Osc1 werken. Hiermee is een lager energieverbruik en veilig werken gewaarborgd, ook bij een lagere werkspanning tot 2,2 V.

6 Pulslengtemeting

Met E4 aan GND wordt na een reset, het voorbeeldprogramma voor het meten van een pulslengte gestart. Ook hierbij wordt de toestand van de ingang S1 bepaalt.



Afb. 13: E4 aan GND


Afb. 14: start van de pulslengtemeting

In de toestand S1 = 0, dus bij ingedrukte toets, loopt er een tijdmeting. Bij de wachttijd van 5 ms komt nog eens ca. 5 ms voor de uitvoering van de in totaal vijf instructies van de tellus. De tijdseenheid van de meting is daarom 10 ms.

Adres	Instructie	Gegevens	Commentaar
34	2	2	Wacht 5 ms
35	C	C	S1 = 0?
36	3	2	Spring –2
37	4	0	A = 0
38	2	2	Wacht 5 ms
39	7	1	A = A + 1
3A	5	4	Port = A
3B	C	E	S1 = 1?
3C	3	4	Spring –4
3D	3	9	Spring –9

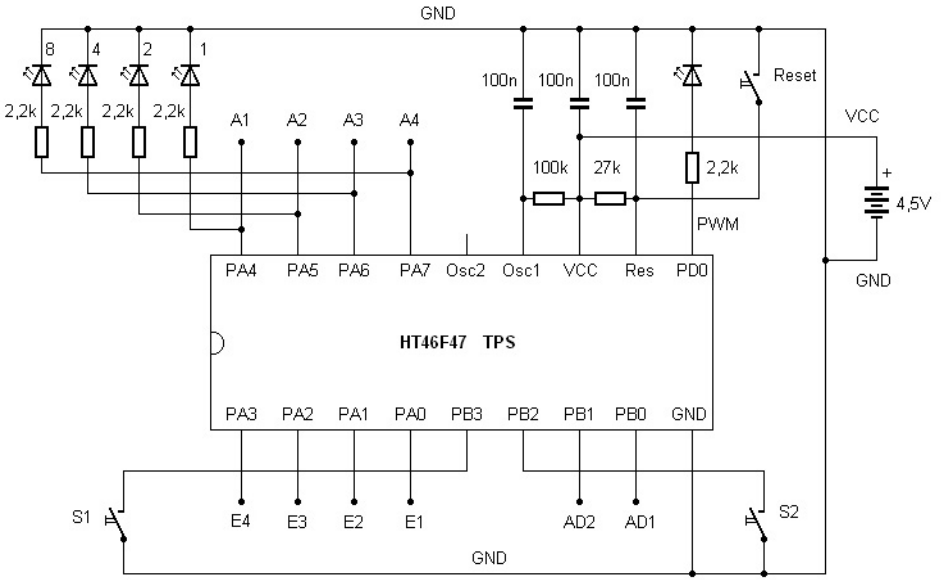
Listing 5: tijdmeting

Druk de toets S1 zo kort mogelijk in. U krijgt dan bv. het resultaat 1010, dus decimaal 10. Omdat de tijdeenheid van het programma 10 ms bedraagt, betekent dit resultaat 100 ms. Met enige oefening kunt u nog kortere tijden tot 50 ms bereiken.

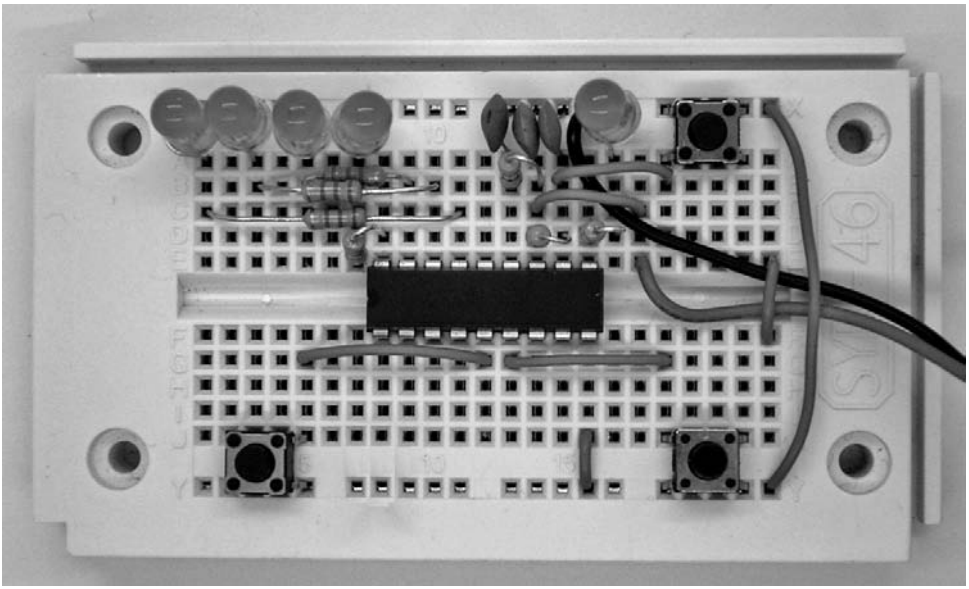
7 Programma's uitlezen

Voor de programmering wordt gebruik gemaakt van de toetsen S1 (gegevensinvoer, links) en S2 (programmeren, rechts). Daarnaast is de reset-toets vereist. Alleen met deze toetsen kunnen programma's worden uitgelezen en kunnen willekeurige programma's worden ingevoerd. Met enige oefening kunt u daarmee in korte tijd nieuwe programma's invoeren en bestaande programma's aanpassen.

U kunt de programmeermodus inschakelen door een reset: houd de programmeertoets S2 ingedrukt en laat S2 pas ca. een seconde na de reset los. U kunt nu met alleen de toets S2 door het actieve programma bladeren en de instructies en gegevens bekijken. Voor elk adres moet toets S2 tweemaal worden ingedrukt. Op deze manier schakelt u over tussen weergave van de instructie en de gegevens. Daarnaast wordt gedurende een korte tijd, het adres weergegeven.



Afb. 15: S1 en S2 voor de programmeermodus



Afb. 16: drie toetsen en LED-display

- Eerste toetsdruk S2
- Adres (laagste vier bits) weergeven, 300 ms
- Weergave uit, 300 ms
- Instructie weergeven
- Tweede toetsdruk S2
- Gegevens weergeven
- Derde toetsdruk S2
- Volgende adres weergeven, 300 ms
- etc.

Wanneer u bijvoorbeeld een bestaand programma met vijf stappen alleen wilt bekijken en niet wijzigen, komt u met 10x indrukken van S2 aan het einde. Omdat steeds het huidige adres even wordt weergegeven, kunt zich eenvoudig oriënteren U weet altijd of de huidige weergave een instructie of gegevens bevat.

Bij levering, bevinden zich de volgende instructies op de eerste vijf adressen. Het gaat hierbij om het begin van een keuzeprogramma voor het starten van de verschillende voorbeeldprogramma's.

Adres	Instructie	Gegevens	Commentaar
00	6	4	A = Din
01	5	1	B = A
02	4	E	A = 14
03	8	0	AdrHi = 0
04	C	3	A = B?

Listing 6: programmacode in fabriekstoestand

Een 5-bit instructie en de bijbehorende 4-bits gegeven, vormen samen en Byte, dus een 8-bits getal. Een halve Byte wordt ook wel "Nibble" genoemd. Het hoogste nibble bevat de instructie, het laagste nibble de bijbehorende gegevens. In de EEPROM van de controller heeft een capaciteit van 128 Bytes. Hierdoor kan een programma maximaal 128 instructies bevatten. Dat is voor de meeste toepassingen voldoende omdat de programmacode extreem compact is. Voor veel handige programma's zijn minder dan 10 instructies nodig.

Haal de verschillende instructies en gegevens op de display en vergelijk de inhoud van het geheugen. Druk daarna opnieuw op de reset-toets. Het oude programma start zonder wijzigingen.

8 Programma's invoeren

De toets S1 wordt gebruik om een instructie of de gegevens te wijzigen of opnieuw in te voeren. In principe kunnen alleen getalswaarden tussen 0 en 15 worden ingevoerd. Met de eerste druk op S wordt 0 ingesteld. Elke volgende druk verhoogt het getal met 1. De huidige positie wordt via de vier LED's weergegeven. Wanneer u bv. 4 wilt invoeren, drukt u in totaal vijfmaal op S1: 0, 1, 2, 3, 4. De binaire weergave luidt dan 0100.

Wanneer op deze manier de instructie, de gegevens of beide opnieuw zijn ingevoerd, zorgt de tweede druk op S2 ervoor, dat deze Byte in de EEPROM wordt opgeslagen. Om dat te verduidelijken, wordt de LED-display gedurende 600 ms uitgeschakeld voordat het volgende adres en daarna het volgende instructie wordt weergegeven. Deze korte pauze kunt u intuïtief zien als uitvoering van de programmeeropdracht. U kunt in uw achterhoofd de voorstelling aanhouden dat het systeem de energie voor de display bespaart en gebruikt voor de programmering in de

EEPROM. Iets vergelijkbaars kent u mogelijk van uw auto: wanneer u de motor start, gaan voor een moment het licht en de radio uit.

U kunt een bestaand programma ook slechts op één plek veranderen. Met S2 bladert u dan naar de gewenste plaats en u verandert met S1 de instructie of de gegevens, die vervolgens met S2 worden opgeslagen.

Voor een eerste test wordt een programma ingevoerd met slechts twee instructies. Hiermee worden drie LED's ingeschakeld en start een eindeloze lus.

Adres	Instructie	Gegevens	Commentaar
00	1	7	A1–4 = 0111
01	3	0	Springe 0

Listing 7: LED's inschakelen

In plaats van een uitgebreide listing kunt u ook kiezen voor een verkorte schrijfwijze. Hierbij worden de beide Bytes samengevat in een hexadecimale notatie: 17h, 30h. Hieronder wordt verder gebruik gemaakt van de hexadecimale schrijfwijze. De programma's worden daarom in de verkorte schrijfwijze zonder de HEX-markering geschreven: 17 30

Voor de invoer moet het volgende worden geschreven:
S2 + reset
2x S1
S2
8x S1
S2
4x S1
S2
1x S1
S2

Wanneer u per ongeluk een keer teveel op S1 heeft gedrukt, kan het juiste getal toch worden worden geschreven. U moet dan nog een keer naar 15 gaan want daarna volgt een overdracht naar de waarde 0.

Na volledige invoer wordt het nieuwe programma met de reset-toets gestart. U ziet dat hierbij drie LED's worden ingeschakeld. Verder gebeurt er niets. De controller reageert nu ook niet meer op de toestanden aan de ingangen E1 tot E4 omdat het oorspronkelijke programma gedeeltelijk is overschreven. Hierdoor kunnen ook de voorbeeldprogramma's niet meer worden gestart.

Omdat u alleen de eerste twee geheugenadressen heeft veranderd, kunt u het oorspronkelijke programma eenvoudig weer opstarten. Hiervoor hoeft u alleen maar de eerste twee instructies (64 51) volgens de listing uit het vorige hoofdstuk opnieuw in te voeren.

Test de oorspronkelijke werking van de voorbeeldprogramma's. U kunt het oefenprogramma het beste nog een keer invoeren. Na enige tijd raakt u vertrouwd met de invoer van de programma's.

9 Herstellen van de voorbeeldprogramma's

Wanneer u na enige tijd de oorspronkelijke toestand van de controller wilt herstellen, kunt u dit doen door de invoer van twee Bytes FF. Dit komt overeen met de toestand van een onbeschreven EEPROM. De firmware van TPS controller bevat een startfunctie die de eerste twee adressen controleert om een leeg geheugen vast te stellen. Wanneer hier twee bytes FF worden gelezen,

gaat de controller er van uit dat er nog geen programma is ingevoerd. In dat geval wordt de EEPROM automatisch gevuld met de voorbeeldprogramma's. Deze functie is eigenlijk bedoeld om de controller bij de eerste start te voorzien van de voorbeeldprogramma's in de EEPROM maar kan ook op elk moment worden gebruikt om de oorspronkelijke toestand te herstellen.

Adres	Instructie	Gegevens	Commentaar
00	F	F	-
01	F	F	-

Listing 8: herstellen van de oorspronkelijke toestand

Start de programmeermodus door een reset met ingedrukte S2 toets. Voer dan in totaal viermaal de waarde F in (decimaal 15) waarbij alle LED's A1 t/m A4 branden. Sluit de laatste invoer af met invoer van S2. Druk vervolgens op reset. De controller heeft nu iets langer dan normaal nodig om alle Bytes van de voorbeeldprogramma's opnieuw te programmeren. Hiermee is de oorspronkelijke toestand hersteld. Test bv. zonder draadbrug op de ingangen het wisselende knipperlicht van pagina 8.

10 TPS-basisinstructies

De programmeerbare besturing beschikt over een totaal van 14 instructies (1-14) Bij veel van deze instructies hoort een parameter in de vorm van een 4-bits getal 0000 tot 1111 (0-F), dus met een getalbereik tot 15 (decimaal). Andere instructies kennen subfuncties die worden ingevoerd in de vorm van parameters. Achter de instructiecode kunnen daarom maximaal 16 subinstructies; verborgen zitten. Zo staat bv. de instructie 7 voor "Bereken A = ...". De parameter geeft aan welke rekenfunctie moet worden uitgevoerd.

Hieronder worden instructies en gegevens samen in hexadecimale vorm geschreven als Byte. De instructie 1 wordt samen met parameter 4 dus de instructie 14h. De Hex markering wordt weggelaten omdat alle instructies en adressen principieel in hexadecimale notatie worden weergegeven.

De eerste drie instructies luiden:
10–1F: directe uitvoer aan de poort A1–A4, 0–15, binair 0000 tot 1111
20–2F: wachttijd 0–15
(1, 2, 5, 10, 20, 50, 100, 200, 500, 1.000, 2.000, 5.000, 10.000, 20.000, 30.000, 60.000 ms)
30–3F: Spring terug 0–15

instructie 1 dient voor de uitvoer van een constant getal aan de poort. Hiermee kunnen willekeurige bitmaskers worden uitgevoerd en bv. ook meerdere LED's gelijktijdig worden ingeschakeld.

De wachtopdracht 2 maakt gebruik van een parameter die de tijd in milliseconden en een onderverdeling in 1-2-5- stappen bevat. Ondanks het kleine getalbereik van 0 tot 15 kunnen op deze manier vertragingstijden tussen een milliseconde en een minuut worden uitgevoerd. Nog langere tijden moeten worden ingevoerd door een wachtopdracht meerdere keren, bv. in een tellus, uit te voeren.

De instructie 3, terugspringen, is bijzonder eenvoudig en is te gebruiken voor vele taken waarbij een reeks instructies eindeloos moet worden herhaald. De sprongafstand is beperkt tot 15. Omdat de sprongafstand relatief is ten opzichte van het huidige adres, kunnen delen van het programma met deze terugspring willekeurig naar andere adressen worden verschoven.

Voor het programma voor het wisselende knipperlicht zijn deze drie opdrachten voldoende. Het programma moet hier iets aangepast in het adresbereik vanaf 00 worden geschreven. Ook met

bitmasker voor de uitvoer en de wachttijden zijn veranderd.

Adres	Instructie	Gegevens	Commentaar
00	1	1	A1–4 = 0001
01	2	7	Wacht 200 ms
02	1	4	A1–4 = 0100
03	2	7	Wacht 200 ms
04	3	4	Spring –4

Listing 9: knipperprogramma

In de verkorte hexadecimale notatie ziet het programma er nu als volgt uit:
11 27 14 27 34

Op basis van de eerste drie instructies kan er al een groot aantal programma's worden geschreven, Analyseer en test de drie volgende programma's. Het doel is deze instructies intuïtief te kunnen gebruiken. Eenvoudige programmastructuren zoals deze kunt u na enige tijd zelfs uit het hoofd programmeren en direct invoeren. Een voorbeeld daarvan is het eenvoudige looplicht met vier uitvoerpatronen:

Adres	Instructie	Gegevens	Commentaar
00	1	1	LED’s 0001
01	2	8	Wacht 500 ms
02	1	2	LED’s 0010
03	2	8	Wacht 500 ms
04	1	4	LED’s 0100
05	2	8	Wacht 500 ms
06	1	8	LED’s 1000
07	2	8	Wacht 500 ms
08	3	8	Spring –8

11 28 12 28 14 28 18 28 38

Listing 10: looplicht 1

Breidt het programma nu uit met twee uitvoerpatronen zodat het licht steeds heen en weer loopt. Experimenteer ook met andere uitvoerpatronen en vertragingstijden.

Adres	Instructie	Gegevens	Commentaar
00	1	1	LED’s 0001
01	2	8	Wacht 500 ms
02	1	2	LED’s 0010
03	2	8	Wacht 500 ms

04	1	4	LED’s 0100
05	2	8	Wacht 500 ms
06	1	8	LED’s 1000
07	2	8	Wacht 500 ms
08	1	4	LED’s 0100
09	2	8	Wacht 500 ms
0A	1	2	LED’s 0010
0B	2	8	Wacht 500 ms
0C	3	C	Spring –12

11 28 12 28 14 28 18 28 14 28 12 28 3C

Listing 11: looplicht 2, heen en weer

Een tijdschakelaar kan met een wachtopdracht een vertraging bevatten van maximaal een minuut. Aan het einde staat een terugsprong met een sprongsfstand 0, dus een eindeloze lus zonder inhoud die dient als einde van het programma. Met de reset-toets wordt een nieuwe start uitgevoerd. U kunt het programma ook eens uitbreiden met een keukentimer voor drie minuten. Hierbij kunt u de resterende tijd weergeven door het aantal brandende LED's als lichtbalk weer te geven.

Adres	Instructie	Gegevens	Commentaar
00	1	F	LED’s 1111
01	2	F	Wacht 1 min
02	1	0	LED’s 0000
03	3	0	Ende

1F 2F 10 30

Listing 12: tijdschakelaar voor één minuut

11 Rekenen met variabelen

Tot nu toe werden voor de parameters voor de verschillende instructies alleen constante getalswaarden gebruikt. Dat is zinvol wanneer een programma altijd op dezelfde manier moet aflopen. Complexere programma's werken daarentegen met variabele gegevens. Dan kan er bijvoorbeeld een berekening zoals A = A + B worden uitgevoerd. Afhankelijk van de inhoud van de variabele A en B zal er steeds iets anders uitkomen. Met het resultaat kunnen bv. de LED's aan de uitgangen worden aangestuurd.

De besturing beschikt over de variabelen A, B, C en D. De belangrijkste variabele is A. Deze wordt ook aangeduid als accumulator of kortweg accu. A wordt gebruikt bij alle rekenoperaties en bevat het resultaat van de berekening. Daarnaast wordt A gebruikt voor het transport van de gegevens. B wordt hoofdzakelijk gebruikt voor rekenoperaties. C en D kunnen worden gebruikt als tussengeheugen en worden later nog gebruikt als tellers in tellussen.

Bovendien zijn er twee analoge ingangen (AD1 en AD2) en een PWM-uitgang. De verwerkte gegevens zijn begrensd tot vier bits en alleen toegankelijk via de variabele A (A = AD1, PWM = A). De accu A kan ook direct met een getal worden geladen (instructies 40–4F). Om B, C of D te vullen, moet eerst A worden geladen en de inhoud vervolgens aan de andere variabele worden toegewezen (instructies 51–53). Met A en B kunnen enkele stappen van een berekening (instructies 71-7A) worden uitgevoerd.

De instructies 40–4F wijzen A een nieuwe waarde toe. De instructies 51–5A wijzen de inhoud van A toe aan een doelvariabele, bv. B of de PWM-uitgang. In deze groep zijn ook instructies die een apart bit van de uitgangspoort kunnen zetten.

Met de instructies 61–6A worden de gegevens precies in de omgekeerde richting verwerkt, gegevens van een bron worden aan A toegeschreven De instructies 71–7A tenslotte voeren enkele rekenoperaties uit waarbij het resultaat altijd aan A wordt toegewezen. De uitgangspoort Dout omvat de vier uitgangen A1 t/m A4 die gezamenlijk of als losse bits Dout.0 t/m/ Dout.3 kunnen worden aangesproken. De ingangen E1 t/m/ E4 worden op dezelfde manier als ingangspoort Din aangesproken.

40–4F: A = 0–15
51–5A: Doe l1–9 = A
51: B = A
52: C = A
53: D = A
54: Dout = A
55: Dout.0 = A.0
56: Dout.1 = A.0
57: Dout.2 = A.0
58: Dout.3 = A.0
59: PCM = A
61–6A: A = Bron 1–10
61: A = B
62: A = C
63: A = D
64: A = Din

65: A = Din.0
66: A = Din.1
67: A = Din.2
68: A = Din.3
69: A = AD1
6A: A = AD2

71 –7A: A = uitvoer 1–10
71: A = A + 1
72: A = A – 1
73: A = A + B
74: A = A – B
75: A = A * B
76: A = A / B
77: A = A And B
78: A = A Or B
79: A = A Xor B
7A: A = Not A

Een voorbeeld voor het gebruik van de variabele A vindt u bij de programmavoorbeelden in hoofdstuk 3. Het programma is hier op adres nul gezet en iets uitgebreid. Er is een gedefinieerd begin bijgekomen met de waarde 0 in de variabele A. Op adres 01 staat een rekenopdracht, hier een verhoging met 1. De inhoud van de variabele A wordt vervolgens aan de PWM-uitgang en de uitgangspoort toegewezen.

Adres	Instructie	Gegevens	Commentaar
00	4	0	A = 0
01	7	1	A = A + 1
02	5	4	Port = A
03	5	9	PWM = A
04	2	6	Wacht 100 ms
05	3	4	Spring –4

40 71 54 59 26 34

Listing 13: verhogen met 1

Een ander voorbeeld is al in hoofdstuk 4 besproken. De gegevens komen daarbij van de analoge ingang AD1 en worden aan de uitgangspoort en de PWM-uitgang toegewezen. Het aangepaste programma bevat nog een extra rekenstap, namelijk de inversie (omkering) van de inhoud van de variabele A. Daardoor wordt namelijk de waarde 0000 omgezet in de nieuwe waarde 1111, d.w.z. 0 wordt 15 en omgekeerd. Een oplopende ingangsspanning wordt op deze manier omgezet in een dalende PWM-uitvoer.

Adres	Instructie	Gegevens	Commentaar
00	6	9	A = AD1
01	5	4	Port = A

02	7	A	A = Not A
03	5	9	PWM = A
04	2	6	Wacht 100 ms
05	3	5	Spring –5

69 54 7A 59 26 35

Listing 14: inverteren

12 Sprongen en vertakkingen

Tot nu toe werd er alleen gebruik gemaakt van de eenvoudige terugsprong (instructie 3) die maximaal 15 adressen achteruit kon springen. Nu komt daar de absolute sprong (jump) bij. Omdat het doeladres voor de sprong met maar 4 bits kan worden opgegeven, is er een extra instructie beschikbaar om het high-nibble van het adres te bepalen. Hiermee ontstaat een adresruimte van 0–255. Dat is meer dan nodig is want de EEPROM van de controller omvat slechts 128 Bytes, dus het bereik 00 tot 7F (decimaal 0 tot 127). Het geheugen is daardoor in acht pagina's 0 t/m 7 (Page 0–7), ingedeeld. Vóór een absolute spong moet de pagina van het doeladres worden opgegeven.

Twee tellussen met de variabele C en D voeren eveneens absolute sprongen uit waarbij ook vooraf de pagina van het doeladres moet worden opgegeven. De voorwaardelijke sprongen werken als "skip"-instructies (weglaten, overslaan). Wanneer aan de resp. voorwaarde is voldaan, wordt een adres overgeslagen. Daar zou bv. een sprongopdracht of een rekenopdracht kunnen staan. Als voorwaarde is een vergelijking tussen A en B en direct uitlezen van een bit aan de ingangspoort beschikbaar.

Daarnaast is er nog een oproep voor een subroutine (Call) en het bijbehorende sprongopdracht (Return). Er zijn weliswaar meerdere subroutines toegestaan maar vanuit een subroutine mag geen andere subroutine worden opgeroepen omdat de interpreter altijd maar één terugsprongadres kan onthouden.

80–8F: Adr-high = 0–15

90–0F: directe sprong (Jump) naar Adr-high, Adr-low

(0–15) A0–AF: tellus C-maal Adr-high, Adr-low

B0–BF: tellus D-maal Adr-high, Adr-low

(0–15)

C1–CF: voorwaardelijke sprong: wanneer (voorwaarde 1–15) dan overslaan
C1: if A > B then Adr = Adr + 1
C2: if A < B then Adr = Adr + 1
C3: if A = B then Adr = Adr + 1
C4: if Din.0 = 1 then Adr = Adr + 1

C5: if Din.1 = 1 then Adr = Adr + 1
C6: if Din.2 = 1 then Adr = Adr + 1
C7: if Din.3 = 1 then Adr = Adr + 1
C8: if Din.0 = 0 then Adr = Adr + 1
C9: if Din.1 = 0 then Adr = Adr + 1
CA: if Din.2 = 0 then Adr = Adr + 1
CB: if Din.3 = 0 then Adr = Adr + 1
CC: if S1 = 0 then Adr = Adr + 1
CD: if S2 = 0 then Adr = Adr + 1
CE: if S1 = 1 then Adr = Adr + 1
CF: if S2 = 1 then Adr = Adr + 1

D0–DF: oproep subroutine (Call) Adr-high, Adr-

low (0-15) E0–EF: terugsprong vanuit een subroutine

(Return)

Een voorbeeld voor de toepassing van voorwaardelijke sprongen vindt u in het voorbeeldprogramma in hoofdstuk 6. Het is is hier enigszins aangepast op adres 0 gezet. Omdat het bovenste deel van het adres (Adr-hi) in rusttoestand 0 is, de controller dus op pagina 0 begint, mag de instructie 80 hier niet worden gebruikt. Hiermee wordt gemeten en weergegeven hoelang de toets wordt ingedrukt. Alle wachtopdrachten zijn uit het programma verwijdert zodat het nu met een hogere resolutie (tijd) werkt.

Adres	Instructie	Gegevens	Commentaar
00	C	C	S1 = 0?
01	3	1	Spring –1
02	4	0	A = 0
03	7	1	A = A + 1
04	5	4	Port = A
05	C	E	S1 = 1?
06	3	3	Spring –3
07	3	7	Spring –7

CC 31 40 71 54 CE 33 37

Listing 15: reactie op de toets S1

De sprongopdracht CC op adres 00 leest de toestand van de schakelaar S1 uit. In rusttoestand is S1 = 1. De voorwaarde is dus niet waar en de instructie op adres 01 wordt niet overgeslagen. Daar staat een relatieve sprongopdracht naar het begin. Het programma herhaalt de instructies op de adressen 00 en 01 totdat de toets wordt ingedrukt. Dan wordt aan de voorwaarde voldaan en wordt het adres 01 overgeslagen. Daarmee begint de eigenlijke meting. De accu wordt gewist en vervolgens steeds met 1 opgehoogd en aan de LED's uitgevoerd. Op adres 05 staat een andere voorwaardelijke sprongopdracht CE. Hier is de voorwaarde voor het overslaan van een instructie S1=1. Omdat de toets in eerste instantie nog was ingedrukt, is niet aan de voorwaarde voldaan. De instructie op adres 06 wordt dus uitgevoerd en er wordt teruggesprongen naar 03. Pas wanneer de toets wordt losgelaten loopt het programma door naar adres 07 en daardoor via de terugsprong naar het begin.

Voer het programma in en test het. De reactietijd is nu aanzienlijk sneller. De tijdeenheid bedraagt nu ca. 5 ms.

Het oorspronkelijke voorbeeldprogramma bevindt zich nog in het geheugen vanaf adres 34h omdat alleen de onderste adressen overschreven zijn. Schrijf een klein programma dat alleen een sprong naar dit adres bevat. Hier moet eerst pagina 3 worden aangewezen. De dan volgende absolute sprong met het opgegeven adres wijst dan naar het werkelijke adres 34.

Adres	Instructie	Gegevens	Commentaar
00	8	3	Pagina 3
01	9	4	Adres = 34

83 94

Listing 16: absolute sprong naar het programma tijdmeting

Hiermee wordt het oorspronkelijke programma weer opgeroepen. Test dit ook eens voor de andere Voorbeelden. In de appendix vindt u een compleet overzicht van alle bruikbare programma's.

13 Overzicht van de instructies

Alle instructies in een oogopslag - dat vereenvoudigt het werken met de controller. De onderstaande tabel bevat alle instructie's in een overzichtelijk vorm.

	1	2	3	4	5	6	7	8	9	A	B	C	D	E
	Port=	Wait	Jump	A=	... = A	A = ...	A = ...	Page	Jump	C*	D*	Skip if ...	Call	Ret
0	0	1 ms	0	0				0	0	0	0		0	
1	1	2 ms	1	1	B = A	A = B	A =	1	1	1	1	A>B	1	
2	2	5 ms	2	2	C = A	A = C	A = A–	2	2	2	2	A<B	2	
3	3	10 ms	3	3	D = A	A = D	A =	3	3	3	3	A = B	3	
4	4	20 ms	4	4	Dout =	A = Din	A = A–	4	4	4	4	Din.0 = 1	4	
5	5	50 ms	5	5	Dout.0 = A.0	A = Din.0	A = A*B	5	5	5	5	Din.1 = 1	5	
6	6	100 ms	6	6	Dout.1 = A.0	A = Din.1	A = A/B	6	6	6	6	Din.2 = 1	6	
7	7	200 ms	7	7	Dout.2 = A.0	A = Din.2	A = A And	7	7	7	7	Din.3 = 1	7	
8	8	500 ms	8	8	Dout.3 = A.0	A = Din.3	A = A Or		8	8	8	Din.0 = 0	8	
9	9	1 s	9	9	PWM = A	A = AD1	A = A		9	9	9	Din.1 = 0	9	

	1	2	3	4	5	6	7	8	9	A	B	C	D	E
A	10	2 s	10	10		A = AD2	A = Not A		A	A	A	Din.2 = 0	A	
B	11	5 s	11	11					B	B	B	Din.3 = 0	B	
C	12	10 s	12	12					C	C	C	S1 = 0	C	
D	13	20 s	13	13					D	D	D	S2 = 0	D	
E	14	30 s	14	14					E	E	E	S1 = 1	E	
F	15	60 s	15	15					F	F	F	S2 = 1	F	

14 Tellussen

Een opdracht moet bv. exact vijfmaal worden uitgevoerd. Hiervoor wordt gebruik gemaakt van een tellus. In dit geval wordt een sprongopdracht exact vijf maal uitgevoerd, daarna niet meer. De telvariabele heet C. De getalswaarde 5 moet eerst aan A worden toegewezen en vervolgens aan C. De instructie A2 voert een absolute sprong naar 02 uit en vermindert tegelijkertijd de inhoud van de variabele C met 1. Wanneer C de waarde 0 heeft bereikt, wordt de sprong niet meer uitgevoerd. Het absolute sprongadres bevindt zich op de opgegeven pagina. Bij een programma op pagina 0 mag de pagina-instructie 80 ook worden weggelaten. Dit is echter absoluut noodzakelijk wanneer er een naar andere pagina moet worden gesprongen.

Adres	Instructie	Gegevens	Commentaar
00	4	5	A = 5
01	5	2	C = A
02	1	5	Port = 0101
03	2	8	500 ms
04	1	A	Port = 1010
05	2	8	500 ms
06	8	0	Pagina 0
07	A	2	C-maal 02
08	3	0	Einde

45 52 15 28 1A 28 80 A2 30

Listing 17: een tellus

Test het programma. De LED's geven bij elke doorloop het patroon 0101 en 1010 weer. Echter, dit deel van het programma wordt kennelijk niet vijfmaal maar precies zesmaal doorlopen. Weliswaar wordt de sprongopdracht op adres 07 daadwerkelijk exact vijfmaal uitgevoerd, om echter de eerste keer op dit adres te komen, wordt er al een knipperopdracht uitgevoerd. Daarom knippert het programma in totaal zes maal.

Wijzig de tijdvariabele naar de waarde 4 en test het programma opnieuw. Nu knipperen de LED's exact vijf maal.

De tellus kan echter ook andersom worden gebruik, zodat niet achteruit- maar vooruit wordt gesprongen. Dit keer wordt het programma daadwerkelijk vijf maal uitgevoerd wanneer C aan het begin de waarde 5 krijgt toegewezen. Het overgeslagen adres 04 bevat dan een relatieve sprong naar zichzelf en daarmee een eindeloze lus die dient als einde van het programma.

Adres	Instructie	Gegevens	Commentaar
00	4	5	A = 5
01	5	2	C = A
02	8	0	AdrHi = 0
03	A	5	C-maal 05
04	3	0	Einde
05	1	5	Port = 0101
06	2	8	Wacht 500 ms
07	1	A	Port = 1010
08	2	8	Wacht 500 ms
09	3	6	Spring –6

45 52 80 A5 30 15 28 1A 28 36

Listing 18: vijf maal knipperen

15 Vergelijkingen

Er moeten twee getallen worden vergeleken. Afhankelijk van het resultaat wordt er een sprong uitgevoerd. De beide getallen moeten in A en B zijn opgeslagen. In het volgende voorbeeld wordt het getal 5 toegewezen aan variabele B. A krijgt de waarde van de analoge ingang AD1. Hier kan bijvoorbeeld, zoals in hoofdstuk 4, een lichtsensor zijn aangesloten. Het programma moet nu de volgende acties uitvoeren.
Wanneer AD1 > 5
dan: alle LED's aan
anders: alle LED's uit

Het eindresultaat is een schemerschakelaar. Omdat de LDR aan GND is aangesloten, zorgt meer licht voor een lagere spanning aan AD1. De LED's gaan uit zodra een bepaalde helderheid wordt overschreden waardoor de spanning onder een bepaalde waarde komt. De grenswaarde ligt op 6 want het eindresultaat van de meting moet groter dan 5 zijn.

Adres	Instructie	Gegevens	Commentaar
00	4	5	A = 5
01	5	1	B = A
02	8	0	AdrHi = 0

Adres	Instructie	Gegevens	Commentaar
03	6	9	A = AD1
04	C	1	Skip if A>B
05	9	8	Adr 08
06	1	F	LED's 1111
07	3	4	Adr 03
08	1	0	LED's 0000
09	3	6	Adr 03

45 51 80 69 C1 98 1F 34 10 36

Listing 19: eenvoudige schemerschakelaar.

Test het programma door de lichtsensor met de hand meer of minder af te dekken. U zult vaststellen dat aan de basisfunctie is voldaan. Helaas treed er meestal een minder fraai neveneffect op: precies op de grens tussen aan en uit gaan de LED's ongecontroleerd knipperen. Vooral bij kunstlicht schommelt de helderheid snel rondom een gemiddelde. Dit knipperen wordt weliswaar door het programma correct geïnterpreteerd maar het resultaat is niet zoals men het van een schemerschakelaar verwacht. In hoofdstuk 18 wordt een verbeterde schemerschakelaar beschreven.

16 AND, OR en XOR

Twee binaire toestanden kunnen tot een nieuwe toestand worden vergeleken. Een voorbeeld daarvan is de AND-functie (en): Wanneer bit 1 de toestand 1 heeft AND bit 2 de toestand 1, wordt de uitgangstoestand ook 1. Ook binaire getallen met meerdere bits kunnen op deze manier worden vergeleken. De vergelijking "10 AND 3 = 2" wordt begrijpelijk wanneer de in binaire getallen wordt genoteerd:

1010 AND 0011 = 0010

Het volgende programma vergelijkt de ingangstoestanden met de constante 3. De AND-functie heeft daarbij als resultaat dat de beide laagste bits worden gemaskerd (uitgefilterd). In rusttoestand heeft de ingangspoort de toestand 1111. De AND-vergelijking met 0011 geeft van de toestand 0011 aan de LED's. Wanneer echter op één van ingangen E1 of E2 aan GND wordt geschakeld, wordt de 0-toestand ook aan de uitgangen zichtbaar. Een verandering aan E3 en E4 heeft geen gevolgen.

Adres	Instructie	Gegevens	Commentaar
00	6	4	A = Din
01	5	1	B = A
02	4	3	A = 3
03	7	7	A = A And B
04	5	4	Port = A
05	3	5	Spring –5

64 51 43 77 54 35

Listing 20: toepassing van de AND-functie

Verander het programma en test ook de andere logische vergelijkingen. De OR-functie (of) kan worden gebruikt om bepaalde ingangstoestanden altijd op 1 te zetten 64 51 43 78 54 35
1010 OR 0011 = 1011

Met de XOR-Funktion (Exclusive OR, 79) kunnen individuele bits worden geïnverteerd:
64 51 43 79 54 35

1010 XOR 0011 = 1001

17 Subroutines

Wanneer delen van een programma meerdere keren worden gebruikt, worden die in een subroutine geschreven. Hiermee wordt vaak geheugenruimte bespaard maar vaak ook veel werk bij het invoeren. Het volgende voorbeeld, laat het gebruik van een subroutine zien die op twee plaatsen vanuit het hoofdprogramma wordt opgeroepen. De subroutine bevat in dit geval alleen een instructie (A = A-1) en de instructie terugspringen. Daarom wordt hier geen geheugenruimte bespaard maar het voorbeeld dient alleen ter demonstratie van de CALL en de RET-instructie.

Hoofdprogramma

Adres	Instructie	Gegevens	Commentaar
00	8	0	AdrHi = 0
01	D	8	Call 08
02	5	4	UItvoer
03	2	9	Wacht 1 s
04	D	8	Call 08
05	5	4	Uitvoer
06	2	8	Wacht 0,5 s
07	3	7	Spring –7

Subroutine:

Adres	Instructie	Gegevens	Commentaar
08	7	2	A = A-1
09	E	0	Ret

80 D8 54 29 D8 54 28 37 72 E0

Listing 21: oproepen van subroutines

Het resultaat van het programma is een aflopende binaire teller met ongelijke vertragingen. Test ook andere instructies in subroutines.

Onder de standaard beschikbare voorbeeldprogramma's zijn meerdere handige subroutines voor algemene toepassingen. De listings hiervan zijn volledig in de appendix opgenomen. Voor eigen gebruik is alleen het startadres nodig:

50: subroutine: geluid lang
52: subroutine: geluid kort
53: subroutine: geluid willekeurig, lengte in A
60: subroutine: wachten tot indrukken van toets aan S1
68: subroutine: wachten tot indrukken van toets aan S2
70: subroutine: getallen invoeren met S1 en S2

De subroutine vanaf adres 60 wordt nu gebruikt om een teller op te bouwen die via de toets S1 wordt aangestuurd. De teller begint op 0. Het hoofdprogramma is relatief kort omdat de complexe taak van het uitlezen van de toets in de subroutine wordt ondergebracht.

Adres	Instructie	Gegevens	Commentaar
00	4	0	A = 0
01	5	4	Uitvoer
02	7	1	A = A+1
03	8	6	Pagina 6
04	D	0	Call 60, toets S1
05	3	4	Spring –4

40 54 71 86 D0 34

Listing 22: via S1 gestuurde teller

Test het programma. Wanneer u tien maal op S1 drukt moet het resultaat 1010 zijn. Wijzig het programma zo dat de subroutine vanaf adres 68 wordt gebruikt. Nu reageert de teller op S2.

18 Schemerschakelaar

Een schemerschakelaar moet het licht inschakelen wanneer het omgevingslicht onder een bepaalde grenswaarde komt. Wanneer het lichter wordt, moet het licht weer worden uitgeschakeld. Er moet voor worden gezorgd dat het licht op de grens van licht en donker niet knippert. Dit is mogelijk met behulp van een hysterese, d.w.z. een zekere afstand tussen de inschakel- en de uitschakelhelderheid. Het hier beschreven programma werkt volgens de volgende regels:

- Wanneer de spanning aan AD1 niet hoger wordt dan 5, wordt het licht uitgeschakeld.
- Wanneer de spanning aan AD1 niet lager wordt dan 9, wordt het licht ingeschakeld.

Hiermee ontstaat een middenbereik waarin de toestand niet wordt gewijzigd. Dit bereik voorkomt dat de LED's gaan knipperen.

0–5: LED's uit
6-8: LED's onveranderd
9-15: LED's aan

Adres	Instructie	Gegevens	Commentaar
00	1	0	LED's 0000
01	4	5	A = 5

02	5	1	B = A
03	6	9	A = AD1
04	C	1	Skip if A>B
05	1	0	LED's 0000
06	4	9	A = 9
07	5	1	B = A
08	6	9	A = AD1
09	C	2	Skip if A<B
0A	1	F	LED's 1111
0B	3	A	Spring –10

10 45 51 69 C1 10 49 51 69 C2 1F 3A

Listing 23: schemerschakelaar met hysterese

19 LED-dimmer

Het doel van dit voorbeeldprogramma is een regelbare LED-lamp. De helderheid van een LED aan de PWM-uitgang moet via toetsen kunnen worden ingesteld. Hierbij kan met steeds kort op een toets drukken om het volgende helderheidsniveau in te stellen of men drukt langer op de toets waarbij de helderheid ook continue veranderd.

In de kern van het programma worden de bekende skip- instructies toegepast. Wanneer de resp. toets niet wordt ingedrukt, wordt de bijbehorende instructie voor het ophogen of verlagen van de inhoud van de accu overgeslagen. Het probleem is echter dat er normaal gesproken ook een overdracht van 15 naar 0 of van 0 naar 15 kan voorkomen. Het is iets ingewikkelder om deze overdracht te voorkomen. Hiervoor moet namelijk steeds worden uitgelezen of de onderste grens (0) of de bovenste grens (15) al is bereikt. Omdat bij een vergelijking altijd de accu wordt gebruikt, moet de inhoud daarvan tussendoor steeds worden opgeslagen. Hiervoor wordt gebruik gemaakt van variabele C.

Adres	Instructie	Gegevens	Commentaar
00	8	0	AdrHi = 0
01	5	9	PWM = A
02	2	7	200 ms
03	5	2	C = A
04	4	F	A = 15
05	5	1	B = A
06	6	2	A = C
07	C	2	Skip if A<B
08	9	B	Spring 0B
09	C	F	Skip if S2 = 1
0A	7	1	A = A + 1
0B	5	2	C = A

0C	4	0	A = 0
0D	5	1	B = A
0E	6	2	A = C
0F	C	1	Skip if A>B
10	9	0	Spring 00
11	C	E	Skip if S1 = 1
12	7	2	A = A – 1
15	9	0	Spring 00

80 59 27 52 4F 51 62 C2 9B CF 71 52 40 51 62 C1 90 CE 72 90

Listing 24: helderheidsregeling

20 Cijferslot

Het hier beschreven cijferslot schakelt de PWM-uitgang in wanneer de gebruiker de juiste getallen heeft ingevoerd. De invoer van de getallen moet exact volgens het geprogrammeerde patroon via de toetsen S1 en S2 gebeuren. Het volgende programma laat zien hoe de verschillende getallen via S1 worden ingevoerd. Net als bij het programmeren zorgt de eerste druk op de toets voor het resultaat 0000. Elke volgende druk op S1 verhoogt de uitvoer met 1. Door indrukken van S2 wordt de invoer beëindigd. In dit geval eindigt het programma in een eindeloze lus.

Adres	Instructie	Gegevens	Commentaar
00	C	C	S1 = 0?
01	3	1	Spring –1
02	4	0	A = 0
03	5	4	Dout = A
04	2	3	10 ms
05	C	E	S1 = 1?
06	3	2	Adr 04
07	C	F	S2 = 1?
08	3	0	Einde
09	C	C	S1 = 0?
0A	3	3	Adr 07
0B	7	1	A = A + 1
0C	2	3	10 ms
0D	C	C	S1 = 1?
0E	3	1	Adr 0D
0F	3	C	Adr 03

CC 31 40 54 23 CE 32 CF 30 CC 33 71 23 CC 31 3C

Listing 25: invoeren van een getal

De invoer van een getal is ook beschikbaar als kant en klare subroutine vanaf adres 70. In plaats van de eindeloze lus in regel 08 staat hier een RET-instructie. De subroutine wordt met het resultaat van de invoer in A verlaten.

Het volgende cijferslot roept de getalinvoer driemaal op en vergelijkt het resultaat met de voorgedefinieerde getallen. In dit voorbeeld luidt de correcte invoer 3, 5, 2. Daarna wordt de PWM-uitgang met de waarde 15 volledig opengezet. Elke verkeerde invoer leidt daarentegen tot een eindeloze lus die alleen met een reset kan worden verlaten.

De PWM-uitgang wordt in dit voorbeeld behandeld als een normale digitale poort. Dat is noodzakelijk omdat alle vier de uitgangen A1 t/m A4 nodig zijn voor de invoer van de cijfers. Na elke volledige invoer worden de vier LED's gewist om een eventuele toeschouwer zo min mogelijk informatie te geven over de geheime cijfercombinatie.

Adres	Instructie	Gegevens	Commentaar
00	8	7	Pagina 7
01	4	3	A = 3
02	5	1	B = A
03	D	0	call 70
04	C	3	Skip if A=B
05	3	0	Einde
06	1	0	LED’s uit
07	4	5	A = 5
08	5	1	B = A
09	D	0	call 70
0A	C	3	Skip if A=B
0B	3	0	Einde
0C	1	0	LED’s uit
0D	4	2	A = 2
0E	5	1	B = A
0F	D	0	call 70
10	C	3	Skip if A=B
11	3	0	Einde
12	1	0	LED’s uit
13	4	F	A = 15
14	5	9	PWM=A
15	3	0	Einde

87 43 51 D0 C3 30 10 45 51 D0 C3 30 10 42 51 D0 C3 30 10 4F 59 30

Listing 26: het cijferslot

21 Appendix

Listings van de voorbeeldprogramma's

Adres	Instructie	Gegevens	Commentaar
00	6	4	A = Din
01	5	1	B = A
02	4	E	A = 1110
03	8	0	Pagina 0
04	C	3	A = B?
05	9	8	Spring 08
06	8	2	Pagina 2
07	9	5	Spring 25, “Omhoogtellen”
08	4	D	A = 1101
09	8	0	Pagina 0
0A	C	3	A =B ?
0B	9	E	Spring 0E
0C	8	2	Pagina 2
0D	9	A	Spring 2A, »AD/PWM«
0E	4	B	A = 1011
0F	8	1	Pagina 1

64 51 4E 80 C3 98 82 95 4D 80 C3 9E 82 9A 4B 81

Pagina 0: Kiezen en starten van de programmeervoorbeelden

Adres	Instructie	Gegevens	Commentaar
10	C	3	A = B?
11	9	4	Spring 14
12	8	3	Pagina 3
13	9	0	Spring 30, “Toeval”
14	4	7	A = 0111
15	8	1	Pagina 1
16	C	3	A = B?
17	9	A	Spring 1A
18	8	3	Pagina 3

Adres	Instructie	Gegevens	Commentaar
19	9	4	Spring 34, “Stopwatch S1”
1A	4	3	A = 0011
1B	8	2	Pagina 2
1C	C	3	A = B?
1D	9	0	Spring 20 “Wisselend knipperlicht”
1E	8	4	Pagina 4
1F	9	0	Spring 40 “Stopwatch S1/S2”

C3 94 83 90 47 81 C3 9A 83 94 43 82 C3 90 84 90
Pagina 1: Kiezen en starten van de voorbeeldprogramma's

Adres	Instructie	Gegevens	Commentaar
20	1	1	0001 “Wisselend knipperlicht”
21	2	8	Wacht 500 ms
22	1	8	1000
23	2	8	Wacht 500 ms
24	3	4	Spring –4
25	7	1	A = A + 1 “Omhoog tellen”
26	5	4	Port = A
27	5	9	PWM = A
28	2	6	Warte 100 ms
29	3	4	Spring –4
2A	6	9	A = AD1 »AD/PWM«
2B	5	4	Port = A
2C	5	9	PWM = A
2D	2	6	Wacht 100 ms
2E	3	4	Spring –4
2F	F	F	-

11 28 18 28 34 71 54 59 26 34 69 54 59 26 34 FF
Pagina 2: voorbeeldprogramma's: wisselend knipperlicht, optellen, AD/PWM

Adres	Instructie	Gegevens	Commentaar
30	5	4	Port = A “Toeval”
31	C	E	S1 = 1?
32	7	1	A = A + 1
33	3	3	Spring –3
34	2	2	Wacht 5 ms “Stopwatch S1”
35	C	C	S1 = 0?
36	3	2	Spring – 2
37	4	0	A = 0
38	2	2	Wacht 5 ms
39	7	1	A = A + 1
3A	5	4	Port = A
2B	C	E	S1 = 1?
3C	3	4	Spring –4
3D	3	9	Spring –9
3E	F	F	-
3F	F	F	-

54 CE 71 33 22 CC 32 40 22 71 54 CE 34 39 FF FF

Pagina 3: voorbeeldprogramma's: toevalsgenerator stopwatch S1

Adres	Instructie	Gegevens	Commentaar
40	8	6	Pagina 6 “Stopwatch start/stop”
41	D	0	Oproep “Wacht S1”
42	4	0	A = 0
43	7	1	A = A + 1
44	5	4	Port = A
45	2	3	Wacht 10 ms
46	C	D	S2 = 0?
47	3	4	Spring –4
48	D	8	Oproep “Wacht S2”
49	4	0	A = 0
4A	5	4	Port = A
4B	3	B	Spring –11

Adres	Instructie	Gegevens	Commentaar
4C	F	F	-
4D	F	F	-
4E	F	F	-
4F	F	F	-

86 D0 40 71 54 23 CD 34 D8 40 54 3B FF FF FF FF
Pagina 4: voorbeeldprogramma's stopwatch start/stop

Adres	Instructie	Gegevens	Commentaar
50	4	F	A = 15 “Geluid lang”
51	9	3	Adr 03
52	4	5	A = 5 “Geluid kort”
53	5	3	D = A “Geluid variable”
54	1	9	A4 = 1
55	1	1	A4 = 0
56	2	1	2 ms
57	1	9	A4 = 1
58	1	1	A4 = 0
59	2	1	2 ms
5A	1	9	A4 = 1
5B	1	1	A4 = 0
5C	2	0	1 ms
5D	B	4	Dmal 04
5E	1	0	Dout 0
5F	E	0	Return

4F 93 45 53 19 11 21 19 11 21 19 11 20 B4 10 E0
Pagina 5: subroutine geluidsuitvoer

Adres	Instructie	Gegevens	Commentaar
60	2	3	Wacht 10 ms “Wacht S1”
61	C	E	S1 = 1?
62	3	2	Spring –2
63	2	3	Wacht 10 ms
64	C	C	S1 = 0?
65	3	1	Spring –1
66	E	0	Return
67	F	F	-
68	2	3	Wacht 10 ms “Wacht S2”
69	C	F	S2 = 1?
6A	3	2	Spring –2
6B	2	3	Wacht 10 ms
6C	C	D	S2 = 0?
6D	3	1	Spring –1
6E	E	0	Return
6F	F	F	-

23 CE 32 23 CC 31 E0 FF 23 CF 32 23 CD 31 E0 FF
Pagina 6: subroutine wacht S1 en wacht S2

Adres	Instructie	Gegevens	Commentaar
70	C	C	S1 = 0? “Toetsinvoer”
71	3	1	Spring –1
72	4	0	A = 0
73	5	4	Port = A
74	2	3	Wacht 10 ms
75	C	E	S1 = 1?
76	3	2	Spring –2
77	C	F	S2 = 1?
78	E	0	Return
79	C	C	S1 = 0?
7A	3	3	Spring –3
7B	7	1	A = A + 1

Adres	Instructie	Gegevens	Commentaar
7C	2	3	Wacht 10 ms
7D	C	C	S1 = 1?
7E	3	1	Spring – 1
7F	3	C	Spring –12

CC 31 40 54 23 CE 32 CF E0 CC 33 71 23 CC 31 3C

Pagina 7: subroutine toetsinvoer

Instructietabel

	1	2	3	4	5	6	7	8	9	A	B	C	D	E
	Port =	Wait	Jump	A =	... = A	A = ...	A = ...	Page	Jump	C*	D*	Skip if ...	Call	Ret
0	0	1 ms	0	0				0	0	0	0		0	
1	1	2 ms	1	1	B = A	A = B	A = A+1	1	1	1	1	A>B	1	
2	2	5 ms	2	2	C = A	A = C	A = A–1	2	2	2	2	A<B	2	
3	3	10 ms	3	3	D = A	A = D	A = A+B	3	3	3	3	A = B	3	
4	4	20 ms	4	4	Dout =	A = Din	A = A–B	4	4	4	4	Din.0 =	4	
5	5	50 ms	5	5	Dout.0 = A.0	A = Din.0	A = A*B	5	5	5	5	Din.1 = 1	5	
6	6	100 ms	6	6	Dout.1 = A.0	A = Din.1	A = A/B	6	6	6	6	Din.2 = 1	6	
7	7	200 ms	7	7	Dout.2 = A.0	A = Din.2	A = A And B	7	7	7	7	Din.3 = 1	7	
8	8	500 ms	8	8	Dout.3 = A.0	A = Din.3	A = A Or B		8	8	8	Din.0 = 0	8	
9	9	1 s	9	9	PWM =	A =	A = A Xor		9	9	9	Din.1 =	9	
A	10	2 s	10	10		A =	A = Not A		A	A	A	Din.2 =	A	
B	11	5 s	11	11					B	B	B	Din.3 =	B	
C	12	10 s	12	12					C	C	C	S1 = 0	C	
D	13	20 s	13	13					D	D	D	S2 = 0	D	
E	14	30 s	14	14					E	E	E	S1 = 1	E	
F	15	60 s	15	15					F	F	F	S2 = 1	F	

Colofon

www.elo-web.de
Auteur: Burkhard Kainka

ISBN 978-3-645-10104-2

Geproduceerd in opdracht van Conrad Electronic SE, Klaus-Conrad-Str. 1, D-92240 Hirschau .

Elektrische en elektronische apparatuur mag niet met het huishoudelijk afval worden afgevoerd! Verwijder het product aan het einde van de levensduur volgens de geldende wettelijke voorschriften. Voor retournering zijn inzamellocaties ingericht waar u elektrische en elektronische apparatuur kosteloos kunt afgeven. U kunt bij uw gemeente het adres van dergelijke inzamellocaties navragen.

CE

Dit product voldoet aan de geldende CE-richtlijnen bij gebruik volgens de meegeleverde handleiding. De beschrijving hoort bij het product en moet worden meegegeven wanneer u het product doorgeeft.