voor wie wat meer met zijn computer wil doen dan recht voor zijn raap programma's draaien . . .

wij zoeken enthousiaste nieuwe leden. doel is computers (en niet alleen de newbrain) te gebruiken voor het besturen van allerlei apparaten en systemen

de belangrijste activiteit zal zijn het bij elkaar brengen van personen en groepen met gelijke belangstelling waarbij uitwisseling van ervaringen voorop staat

daartoe zullen de volgende activiteiten worden ontplooid:

- het (verder) ontwikkelen van interfaces en andere hardware met de bijbehorende software
- het verzamelen van informatie terzake
- het verwerken van die informatie
- het demonstreren met werkende apparaten en systemen
- het beschikbaar stellen van informatie onder meer in de vorm van publicaties

wie?

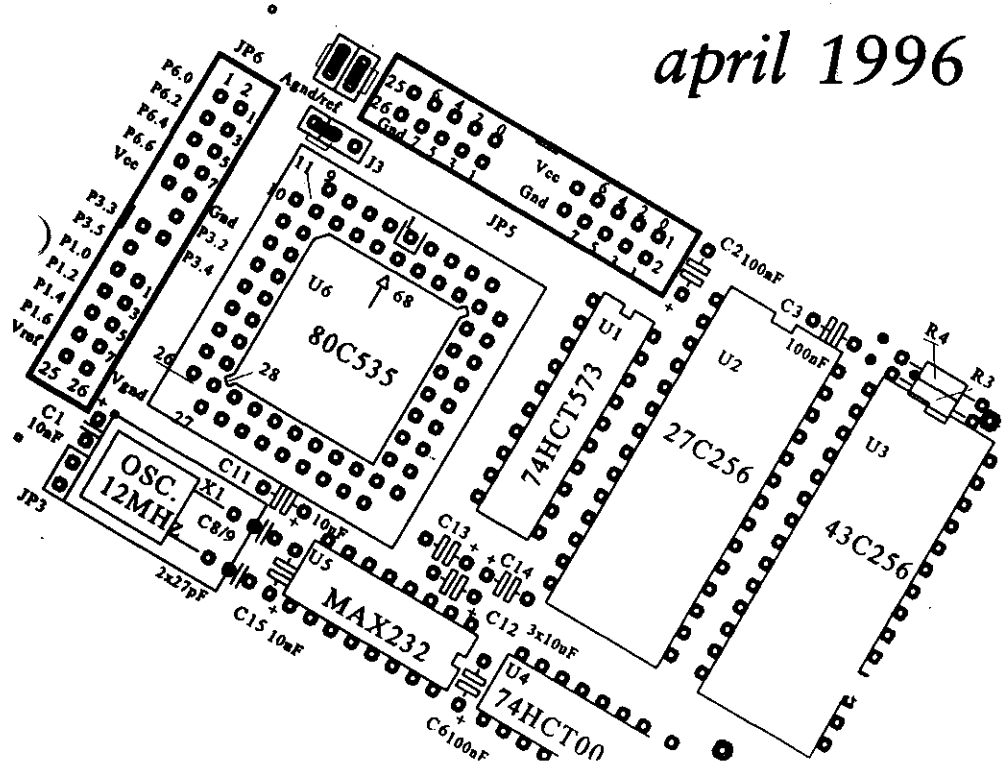
hcc newbrain-gebruikersgroep
postbus 94494
1090 GL amsterdam
telefoon (010) 4557698 / (020) 6924137

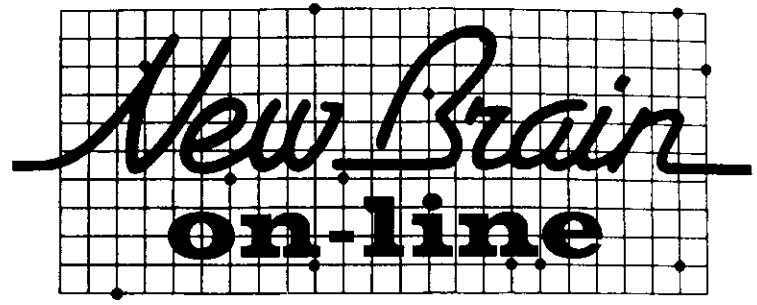
New Brain on-line

uitgave van de NewBrain-gebruikersgroep

20

april 1996





ten geleide

de toegenomen activiteiten binnen de gebruikersgroep hebben staan grotendeels in verband met het b+-besturingsbord, en deze newbrain on-line weerspiegelt dat

we beginnen met de overname van twee afleveringen van ton goossens' artikelenreeks in de softwarebus. deze keer de introductie van de microcontroller en de programmeertaal voor b+. jan wubben geeft een inleiding op het b+-bordje. van bas boetekees is er een overzicht van de stand van zaken bij de ontwikkeling van de nieuwe versie van het bordje

het gebruik in de praktijk wordt door herman van baarzel geïllustreerd met zijn hijskraan. daarbij ziet u ook een programmalisting in b+' eigen taal. de code is opvallend compact. maar met jan wubbens uitleg voor de niet zo geverseerde lezer is goed te volgen, hoe het werkt.we sluiten af met een suggestie van bas boetekees voor een andere toepassing van b+, en met de nagekomen tekeningen van evert drijvers pio-bordje

menno stevens

zaterdag 19 oktober 1996

newbraindag

sbbo
lammenschanspark 1
2311 JK leiden

NewBrain-
gebruikersgroep
postbus 94494
1090 GL amsterdam

de micro-controller

een serieuze poging om de dos-gebruikers enig inzicht in de werking van dit soort hardware te verschaffen

Het hierna volgende artikel is eerder (in twee afleveringen) geplaatst in de Softwarebus, het lijfblad van de DOS-gg. De schrijver ervan, Ton Goossens, is de geestelijke vader van het B+-project. Niet alleen daarom, maar ook voor de auteursrechten zijn wij hem dankbaar.

In de computerwereld spreken we niet van voortdurende verbazing, maar van technische ontwikkelingen. Eerlijk zeggen, een paar jaar geleden had u niet durven dromen, om vandaag met Windows een tekst te kunnen maken compleet met plaatjes en in ieder denkbaar lettertype. En dat ook nog eens te kunnen afdrukken alsof het gedrukt is. Een andere technische ontwikkeling, waar niet zoveel reclame voor wordt gemaakt als voor de Microsoft-produkten, is die van de microcontroller. Net als de computer bestaat de microcontroller al een aantal jaren. Eigenlijk zijn ze tezamen uit hetzelfde ei gekropen. De processor heeft zich in de richting van de pc ontwikkeld en de meeste controllers zijn in de besturingen terecht gekomen. Met name dit stukje van de computerelektronica zal de komende tijd sterk in de lift komen. Alle tekenen wijzen erop: bijna dagelijks nieuwe chips, meer en meer publikaties vooral in Amerikaanse en Duitse tijdschriften, steeds drukkere conversaties over het onderwerp op de internationale bulletinboards enzovoort. Tijd dus om u eens duidelijk uit te leggen, wat die microcontroller nou voor een ding is.

Een microcontroller is een chip met daarop een aantal componenten. Welke componenten is niet gedefinieerd. Een paar voorbeelden zullen het wat duidelijker maken. In de 8051, een controller van Intel, zitten onder andere

zaterdag 19 oktober 1996

newbraindag

sbbo
lammenschanspark 1
2311 JK leiden

NewBrain-
gebruikersgroep
postbus 94494
1090 GL amsterdam

de micro-controller

een serieuze poging om de dos-gebruikers enig inzicht in de werking van dit soort hardware te verschaffen

Het hierna volgende artikel is eerder (in twee afleveringen) geplaatst in de Softwarebus, het lijfblad van de DOS-gg. De schrijver ervan, Ton Goossens, is de geestelijke vader van het B+-project. Niet alleen daarom, maar ook voor de auteursrechten zijn wij hem dankbaar.

In de computerwereld spreken we niet van voortdurende verbazing, maar van technische ontwikkelingen. Eerlijk zeggen, een paar jaar geleden had u niet durven dromen, om vandaag met Windows een tekst te kunnen maken compleet met plaatjes en in ieder denkbaar lettertype. En dat ook nog eens te kunnen afdrukken alsof het gedrukt is. Een andere technische ontwikkeling, waar niet zoveel reclame voor wordt gemaakt als voor de Microsoft-produkten, is die van de microcontroller. Net als de computer bestaat de microcontroller al een aantal jaren. Eigenlijk zijn ze tezamen uit hetzelfde ei gekropen. De processor heeft zich in de richting van de pc ontwikkeld en de meeste controllers zijn in de besturingen terecht gekomen. Met name dit stukje van de computerelektronica zal de komende tijd sterk in de lift komen. Alle tekenen wijzen erop: bijna dagelijks nieuwe chips, meer en meer publikaties vooral in Amerikaanse en Duitse tijdschriften, steeds drukkere conversaties over het onderwerp op de internationale bulletinboards enzovoort. Tijd dus om u eens duidelijk uit te leggen, wat die microcontroller nou voor een ding is.

Een microcontroller is een chip met daarop een aantal componenten. Welke componenten is niet gedefinieerd. Een paar voorbeelden zullen het wat duidelijker maken. In de 8051, een controller van Intel, zitten onder andere

een processor, twee timer/counters, een interruptcontroller, een seriële poort, een baudrate generator en een aantal 8 bits bidirectionele poorten. In de 80188, de controlleruitvoering van de 8088 (uit uw XT) zitten behalve de reeds genoemde componenten uit de 8051 ook nog eens een flink stukje (statisch) geheugen en een (vrij primitieve) DMA-controller. We gaan al die zaken nog wat nader bekijken, dus niet stoppen met lezen, als er af en toe iets voorbij komt dat u nog niet kent.

Je zou dus eenvoudig kunnen zeggen dat een microcontroller (bijna) alle zaken bevat die nodig zijn om een complete computer te maken. Dat is in eerste aanleg ook de bedoeling. Een computer bedoeld voor een speciale taak, een *dedicated computer*. Het aardige van deze taak is (vrijwel altijd), dat je geen I/O nodig hebt. Even uitleggen. De specifieke taak voor de microcontroller zit in de controller 'ingebakken'. Er is dus geen floppy of harddisk nodig om uit te leggen, welk programma er op welke manier moet draaien. Dat scheelt natuurlijk enorm in ruimte, kosten enzovoort. Het nadeel is dan ook, dat deze controller niets anders kan dan de taak waarvoor hij is geprogrammeerd! Een microcontroller is dus een ding waarmee je op eenvoudige wijze kleine computerproblemen kunt oplossen.

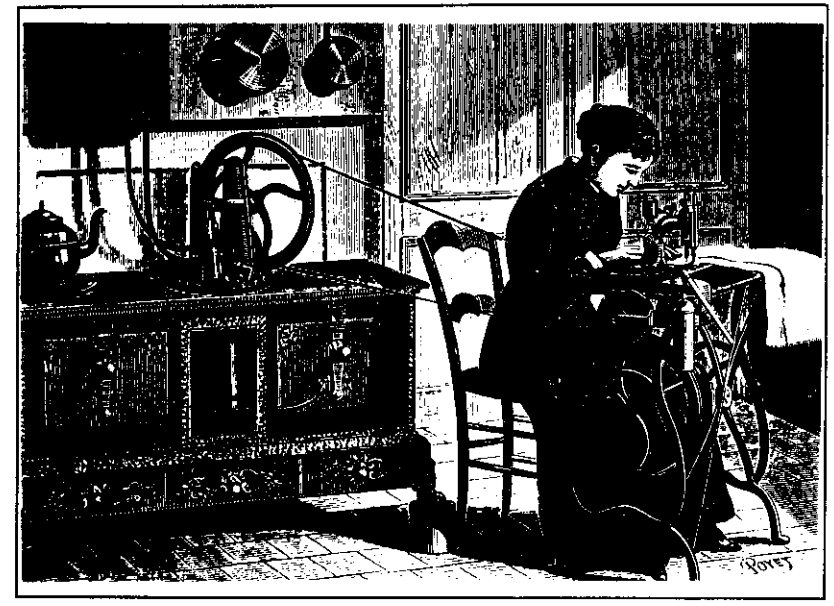
Stel je voor, je hebt een toetsenbord met daarop 120 toetsen. De gebruiker kan op ieder willekeurig moment een van de toetsen indrukken; de gebruiker kan ook een combinatie van toetsen indrukken. Gevraagd wordt: wat is de code van de ingedrukte toets(en). Nou, met een beetje logische chips, zoals poortjes, tellers en priority interrupt chips kun je dat probleem wel klaren. Maar, en hier komt de truc, met een microcontroller gaat het eenvoudiger. Door een processor te gebruiken wordt de hardware, dus het aantal gebruikte chips, eenvoudiger. Een nadeel van het gebruik van die processor is dat er nu een *programma* in het spel komt. De processor moet natuurlijk even weten wat hij daar moet doen en de verzameling van instructies aan die processor noemen we per definitie: een programma.

Nu wordt de keuze tussen 'veel hardware' of 'een processor inzetten' dus duidelijker. Er wordt veel hardware gebruikt als het een produkt betreft met een kleine omzet, anders wordt gekozen voor de goedkope processor, compleet met de goedkope montage maar met een duur (ontwikkeling) programma.

In het toetsenbord van uw computer zit dus een microcontroller, compleet

met een programma. De processor in uw keyboard heeft voor dat programma zelfs geen aparte chip meer, die is ook geïntegreerd in de microcontroller. Dat geeft weer als voordeel, dat het programma gedurende de produktie van de processor in de chip kan worden aangebracht, dus een produktiestap wordt uitgespaard.

Zo zult u op bepaalde punten in uw pc mogelijk nog andere microcontrollers tegenkomen. Bijvoorbeeld misschien op de soundcard, zeker in de IDE-drive, misschien op de videocard, zeker in de keyboardcontroller en ga zo maar door.



Ook buiten uw pc zijn de microcontrollers te vinden. In uw tv zorgt de controller voor het ontvangen van de infraroodsignalen van de afstandsbediening, voor het inlezen van de teletekstpagina's, voor het automatisch opzoeken van de zenders enzovoort. Een beetje koffiepot heeft toch zeker een microcontroller die zorgt dat uw koffie op een redelijk voorspelbaar tijdstip in de ochtend klaar staat. Natuurlijk zit er een in de magnetron, in de vaatwasser, in de wasmachine en bij een hobbyist in Steenbergen ook in het

aquarium. In het *Bitje Randstad*, een publikatie van een aantal HCC-afdelingen in de Randstad, staat vanaf september een artikelenreeks over het maken van een huisbeveiligingscomputersysteempje, met microcontroller natuurlijk.

Wordt het niet eens tijd, dat u iets van die microcontrollers gaat begrijpen?

Een microcontroller is, zoals gezegd, gewoon een processor met daaromheen een aantal dingen, die je anders los zou moeten kopen. Zo simpel is dat. Om nu die microcontroller te kunnen gebruiken moet je er een programma voor schrijven. Als je een microcontroller gaat gebruiken, moet je een of andere programmeertaal gebruiken. Vrijwel altijd wordt daarvoor de taal *assembler* gebruikt. Dat is een taal die op het laagste niveau code aanmaakt voor de desbetreffende processor. En iedere fabrikant heeft zo zijn eigen processorcode ontwikkeld. Het werken met een microcontroller gaat dus nogal ingewikkeld. Aan de hand van een praktisch voorbeeld zullen we proberen u een idee van dat proces te geven.

Stel dat er een wasmachine moet worden bestuurd met een controller. De machine staat klaar. In de wasmachine zit een complete microcontroller met daarop aangesloten de motoren, de pompen, de kleppen en wat er zoal in zo'n machine zit. De programmeur gaat achter zijn computer zitten (meestal dezelfde pc als die van u) en maakt met een tekstverwerker een programma. Als dat er goed uitziet, laat hij dat programma omzetten door een assembler. Daaruit komt hetzelfde programma maar dan in code die de controller kan begrijpen. *Machinecode* dus. Die code gaat de programmeur in een *eprom* plaatsen (dat wordt ook wel *branden* genoemd), waarna de eprom met daarin het programma in de microcontrollerprint wordt geplaatst. De wasmachine wordt gestart en de foutjes in het programma, bijvoorbeeld tijden die te lang of te kort zijn, worden genoteerd en aan de programmeur gegeven, die zijn tekstverwerker weer voor de dag haalt, het originele programma aanpast, de assembler start, enzovoort.

Voor een leek - en dat zijn we toch (bijna) allemaal - is dat niet te doen. U wilt niet gaan programmeren in assembler, u wilt iets gaan besturen. Bijvoorbeeld uw huis gaan beveiligen. Daarbij moet er een alarm afgaan, als er iemand in huis rondloopt terwijl de beveiliging aanstaat. Daarvoor moet u de waarden van de naderingssensor uitlezen, misschien een telefoonnummer laten kiezen of iets anders ondernemen, maar niet gaan programmeren in assembler. Wat nu?

Stel je voor dat er in de microcontroller een programma zou zitten dat zo werkt: als er via de ingang een eenvoudig commando binnenkomt, dan vertaalt de computer dat en voert daarvoor een complexe bewerking uit. Een soort hulpprogramma dus. Daar zijn we mee aan de slag gegaan en het resultaat mag er zijn. Voor het programmeren van bovengenoemde beveiliging kunt u nu een aantal eenvoudige commando's geven en het resultaat direct bekijken.

Nou, er is door een paar hobbyisten zo'n programma gemaakt voor die microcontroller. De werking is zo simpel, dat u zich nu kunt blijven concentreren op de werking van uw programma, zonder dat u zich om het programmeren zorgen hoeft te maken. Het gaat zo: met een willekeurige tekstverwerker op uw pc maakt u het programmaatje. Daarna stuurt u dat stukje tekst via een communicatieprogramma (bijvoorbeeld *Telix* of *Procomm*) op naar de microcontroller. In die controller wordt uw tekst omgezet naar machinecode.

Daarna kunt u uw programma starten en kijken of het resultaat is wat u er van verwacht. Als dat niet zo is, en het zou wel een wonder zijn als het de eerste keer precies deed wat u wilt, verander dan de tekst op uw pc, stuur het even op en test het opnieuw. Als het goed werkt, haal dan de koppeling met de pc weg en u heeft een compleet zelfstandig werkende microcontroller. Een paar afdelingen van de vereniging hebben een microcontroller aangeschaft en de leden kunnen daar tijdens de bijeenkomsten mee spelen.

In het voorgaande is eigenlijk alleen maar reclame gemaakt voor het onderwerp. Nu zullen we ingaan op de werking van de verschillende componenten van de microcontroller en eens kijken of de reclame wel op zijn plaats is.

Een paar lezers hebben contact opgenomen om wat meer informatie. Nou, er zijn ondertussen een paar afdelingen waar enige activiteiten met betrekking tot de microcontroller plaats vinden. Daar verwijs ik de vragenstellers ook naar. Dus, als u vragen heeft ten aanzien van de 'werkende' microprocessor, bijvoorbeeld hoe groot is het allemaal of, laat nou eens zien hoe dat allemaal werkt, ga dan eens langs bij de bijeenkomst die uw afdeling houdt en informeer daar eens naar de microcontroller. Trouwens, alle bellers bedankt. Dat even terzijde.

de instructies

We zouden in deze aflevering iets gaan doen aan de instructies. Dat wordt dus een simpel verhaal. Voor we daaraan beginnen nog even de gang van zaken bij het gebruik van een microcontroller met ingebouwde compiler (en voor zover ik weet, is de HCC de enige leverancier van deze uiterst eenvoudige oplossing). U wilt iets besturen, bijvoorbeeld een motortje dat de luxaflex laat zakken als de zon schijnt en ze daarna weer ophijst. Begin met het aansluiten van het motortje aan de microcontroller. Sluit daarna uw computer aan, aan de controller (dat gaat heel eenvoudig, een serieel snoertje met drie draadjes en twee 9-polige D-connectortjes). Als u de voedingsspanning aansluit aan de controller (alles tussen de 7 en de 17 volt is ok), zal de controller zich melden op uw computer.

U kunt nu de aangesloten motor direct besturen via het commando MP, daar komen we later op terug, of, u schrijft een programma voor de besturing van de motor. U schrijft het programma op uw computer in geeft niet welke teksteditor. Bijvoorbeeld de *Norton Editor* of *QuickEdit (QE)*. Als uw programma klaar is, stuurt u het naar de microcontroller, die het ontvangen programma omzet in machinetaal (compileren) en wacht op uw commando om het programma te starten (GO).

Als blijkt dat uw programma doet wat u wilt, dat het moet doen, kunt u de computer loskoppelen. Die heeft u niet meer nodig. De microcontroller gaat nu helemaal zelfstandig te werk. Het programma dat u gemaakt heeft voor uw toepassing komt in de microcontroller in een stukje RAM terecht. Aan dat stukje RAM (32 kilobytes) zit echter een batterijtje zodat uw programma ook in leven blijft, als u met de microcontroller op pad zou gaan, of zelfs als de stroom zou uitvallen. De nieuwe versie van de microcontroller is zo zuinig met stroom, dat die op een klein batterijtje helemaal kan blijven werken gedurende vele uren!

Duidelijk? U heeft alleen even een computer nodig om een programmaatje te maken en dat op te sturen naar uw microcontroller; daarna kan de computer weer voor andere spelletjes zoals Windows en Pacman worden gebruikt. Zullen we dan maar eens naar de programmeertaal kijken?

de programmeertaalcommando's

U kunt hier twee soorten commando's onderscheiden. Ten eerste de monitorcommando's en ten tweede de programmeertaalcommando's. Over de monitorcommando's gaan we het de volgende keer hebben in *De monitorcommando's*. De monitor bij een microcontroller is wel te vergelijken met de DOS op uw computer. U tikt een commando in, en het wordt uitgevoerd of u krijgt een foutmelding. Een monitorcommando dat u bijvoorbeeld zeker nodig zult hebben is natuurlijk: GO.

Eerst komen nu de commando's van de programmeertaal aan de orde. Enige algemeenheden over de schrijfwijze van uw programma:

- De programmeertaal is *case sensitive*. In het Nederlands betekent dat: gebruik grote en kleine letters zoals het is gedefinieerd. Als u V1 moet schrijven en u schrijft v1, dan krijgt u een foutmelding.

- De programmeertaal accepteert meerdere instructies op een regel. U kunt op een regel net zoveel instructies schrijven als u wilt.

- De programmeertaal gebruikt de spaties of tabs in of tussen de instructies niet. Als u schrijft:

  V1 = 11 V2=22V3= 33

  in plaats van:

  V1 = 11 V2 = 22 V3 = 33

  is dat in orde. Het tweede staat alleen een beetje slordig en de eerste regel is niet leesbaar. Gebruik tab's om commando's te scheiden.

Het monitorcommando om instructies in te voeren is ID (Input Data). Op de volgende regel moet u een X typen, daarna uw instructies en tenslotte weer een X. Al deze zaken worden nog uitvoerig behandeld. U krijgt dit voorschotje, om alvast met de instructies te kunnen oefenen.

Er is nog een monitorcommando dat u zou kunnen gebruiken om de afloop van uw programma te testen: het commando MP (Modify Parameter). Type

op de prompt MP. Het programma vraagt dan welke parameter u wilt bekijken of wijzigen. Probeer het maar, het programma stuurt vanzelf verder.

de variabelen

In iedere taal hebben we zogenaamde *variabelen* nodig. Variabelen zijn geheugenplaatsen waar u een getal kunt bewaren. Het commando V bewerkt variabelen. De variabele in onze programmeertaal is 8 bits groot. U kunt dus getallen van 00 tot FF (of anders: van 0 tot 255) opslaan. U werkt met de variabelen in het zogenaamde *hexadecimale* formaat. Het lijkt erger dan het is. Voor de gemiddelde computergebruiker die weet wat een kilo memory is, is het hexadecimale stelsel snel uit te leggen. Trouwens, als u per se wilt, kunt u best decimaal werken! In het laatste stukje staat een korte uitleg over hexadecimaal en wat daarbij hoort.

Een variabele heeft een naam. In onze programmeertaal heten alle variabelen V. U kunt in totaal 127 verschillende variabelen gebruiken, van V0 tot V7F. Dat laatste is een hexadecimaal (hex-) getal.

Alle variabelen hebben bij de start van uw programma de waarde 0. U hoeft de variabelen die u nodig heeft, dus niet op 0 te zetten bij het begin van uw programma.

Met de variabelen kunt u allerlei dingen gaan doen. U kunt er bijvoorbeeld een getal instoppen: V1 = 11. Ook kunt u een getal optellen bij de variabele: V2 + 5. Stel dat V2 het getal 2 bevatte, dan is de inhoud na V2 + 5 dus gelijk aan 7.

V2 + 5 betekent, verhoog de inhoud van V2 met 5. Meestal wordt voor deze optelling een andere notatie gebruikt, zoiets als: V2 = V2 + 5. Volgens ons is dat te uitgebreid. Een nadeel van de notatie, zoals die in deze programmeertaal wordt gebruikt, is dat de opgave V6 = V5 + 2 in twee keer geschreven moet worden! Zo ook: V2 - 6. Daar staat, verlaag de inhoud van V2 met 6.

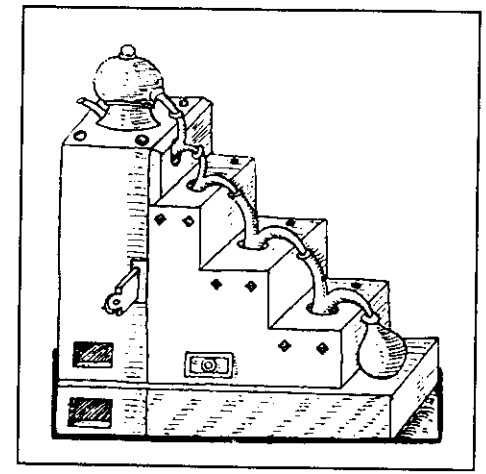
Zoals u wellicht weet, worden in uw computer de letters ook voorgesteld als cijfers. De A (de grote A) heeft als waarde 41. En de kleine a heeft als waarde 61. U mag in onze programmeertaal ook de letters gebruiken als cijfers! V3 = a is dus hetzelfde als V3 = 61. Let bij het gebruik van letters op de aanhalingstekens rond de letters.

Ook logische operaties zijn toegestaan. Dat zijn bewerkingen waarbij de enkele bits van de variabelen worden gebruikt. Er zijn verschillende goede publikaties over dit soort van rekenwerk. In de afdeling Den Haag Zoetermeer Delft is sinds enige tijd een boekje over binair rekenen, waarin al deze geheimen worden ontrafeld.

V2 & 3	staat voor *and* de inhoud van V2 met 3
V2 | 3	staat voor *or* de inhoud van V2 met 3
V2 ^ 3	staat voor *xor* de inhoud van V2 met 3
V2 > 3	staat voor *shift* de inhoud van V2 met 3 naar links
V2 < 3	staat voor *shift* de inhoud van V2 met 3 naar rechts
V2 : 3	staat voor *test* V2 op de waarde 3 (geeft: waar of niet waar)

In het bovenstaande mag u de waarde 3 eventueel vervangen door een andere variabele. Dus: V2 + V5 enzovoort.

V++ en V--

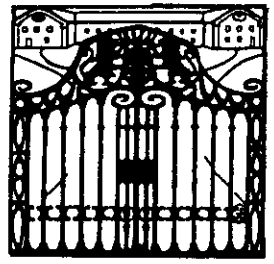


De instructie Vnn + 1 wordt door de compiler anders vertaald dan de instructie Vnn + 2. In het eerste geval wordt gebruik gemaakt van de computerinstructie: *increment* (verhoog) en in het tweede geval wordt een optelling gedaan.

Increment compileert naar 3 bytes en gebruikt 2 microseconden, terwijl de optelling 8 μsec gebruikt. Het nadeel van

het increment is, dat het resultaat niet decimaal gemaakt kan worden. Daar komen we nog op terug. Ook testen op nul is er niet bij. Om dit probleem te omzeilen zijn er een aparte increment- en decrementinstructie gekomen: Vnn ++ voor Vnn + 1, en Vnn -- voor Vnn -1

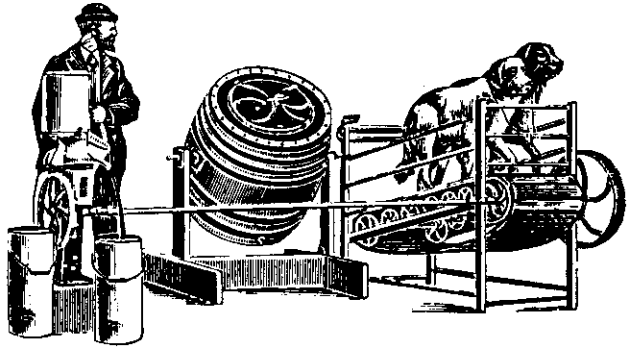
Deze instructies produceren een increment of decrement (verlaag) code en zijn dus niet te gebruiken om op te testen! Hierdoor wordt Vnn + 1 een optelinstructie en kunt u die gebruiken om op te testen. In plaats van V2 + 1 kunt u nu vaak schrijven V2 ++, het is gewoon een kortere schrijfwijze die tot tijdwinst kan leiden.



De volgende keer gaan we spanningen meten en poorten besturen. Probeer voor uzelf een voorstelling te maken van de besturing die u nog eens wilt gaan maken. De theorie over het programmeren wordt dan meteen een stuk duidelijker.

Succes, en deze keer zeg ik het er maar bij: als u vragen heeft, bel gerust even.

Ton Goossens



b+

een inleiding

Voor de meeste vaste lezers van NewBrain on-line zal de inhoud van dit nummer een hoop onbekende termen bevatten. Vandaar, wat is B+? B+ is een *besturingscomputer* plus een ingebouwde *compiler*; een compleet besturingssysteem dus.

De hardware bestaat uit een *processor*, een *eprom* en *ram*, met enkele extra chips, die nodig zijn om het geheel met elkaar te verbinden. Verder is er een spanningsregelaar aangebracht, zodat de computer kan werken op elke simpele netvoeding, die een spanning levert tussen 7 en 16 volt. Ingebouwd in de eprom zit een monitorprogramma, tezamen met een communicatieprogramma.

Het hart van de machine wordt gevormd door een *microcontroller* van Intel, uit de familie van de 8052-set. Hier wordt gebruikt de 80(C)535, die eigenlijk een sterk uitgebreide 8052 is; laten we zeggen de AT onder de microcontrollers.

Doorgaans worden microcontrollers geprogrameerd in *assembler*. Dat is een moeilijke taal, die bovendien nog enkele andere bezwaren telt: het programma moet op een andere machine geschreven worden, waarna telkens als het programma 'af' is, het met speciale apparatuur in een eprom gebrand moet worden. Als dan bij het testen blijkt, dat het niet, of niet goed loopt, moet het programma worden bijgewerkt en opnieuw ingebrand. De oude eprom's moeten overigens, voordat zij worden hergebruikt, eerst worden gewist. Al met al een tijdrovende en ingewikkelde en ook wel kostbare zaak.

Bij B+ moet ook het programma op een andere computer worden geschreven. Maar vervolgens kan de *source* via een RS232-poort B+ ingeblazen

worden. De ingebouwde compiler zet de source om in *machinetaal* en de zaak kan worden getest.

Rest nog de opmerking, dat de ontwerpers en ontwikkelaars van B+ de verdere ontwikkeling en vervaardiging van B+-hardware aan de technici van de NewBrain-gebruikersgroep hebben overgedragen. U hoort nog van ons.

Jan Wubben

enige technische gegevens

hardware

processor	80535 of 80C535
programmageheugen	27256 of 27C256 (32 kB)
werkgeheugen	43256 of 43C256 (32 kB), of 2 maal 43C256 voor 64 kB werkgeheugen
clock cristal	12 Mc
resulterend in	1 µsec processor-cycles
seriële verbinding	9600 Bps (kan ook 4800 Bps)
RS232-poort	8 bits, 1 stopbit, geen pariteit 3-draadsverbinding: receive, transmit en ground geen gebruik interne timer
aantal poorten	7, waarvan 5 beschikbaar: P1, P3, P4, P5 en P6
aantal analoge kanalen	8
nauwkeurigheid A/D-conv.	5 mV voor 10 bits en 20 mV voor 8 bits
meettijd	15 µsec (plus enige overhead)
aantal timers/counters	3
hoogste meetfrekwentie	1 µsec
voeding	7-16 V, 200 mA (spanningsregelaar op print)
accu back-up voor ram	2,4 tot 4,8 volt
formaat print	12 x 7 cm

software

compiler speciaal voor B+
ingebouwde monitor met eigen commando's
ingebouwd communicatieprogramma voor de RS232-poort

versie 2

de nieuwe versie van het B+-bord

Leden van de NewBrain-gebruikersgroep zijn al jaren bezig met het besturen van applicaties via databussen. De gebruikersgroep heeft onder andere een buitenwereldkaart en een pio-kaart. In december 1995 zijn wij benaderd door Ton Goossens om het B+-bord te gaan produceren. Als bestuur hebben we daar *ja* tegen gezegd, ofschoon we ons niet gerealiseerd hebben, hoeveel werk we daarmee op onze schouders namen.

De eerste opdracht was een nieuwe lay-out te maken voor het bordje en deze zo klein mogelijk te houden. Dat is gelukt, want de afmeting is 60 x 115 mm geworden (was 70 x 120 mm). Dit heeft als consequentie ten opzichte van het oude bord, dat poorten nu in lijn samengevoegd zijn, maar voor andere uitbreidingen, zoals geheugen, is een rij van tweemaal 15 pinnen toegevoegd. Ook de RS232-aansluiting is gewijzigd. Bezat het oude bord een D-connector, het nieuwe heeft een 2 x 5-pins header, waarbij via een bandkabel deze rechtstreeks op een 9-pins seriële poort kan worden aangesloten. De verdraaiing van 2 naar 3 en 3 naar 2 zit nu op het bord. Het regelmatig solderen is hopelijk voorbij.

Voor de toekomstige gebruikers van de B+-print is het belangrijk, dat zij over gegevens van de nieuwe print kunnen beschikken. Op de eerste plaats is de lay-out van de print belangrijk. Hij is hier weergegeven inclusief de plaats van de componenten, en voor de spanningsgevoelige onderdelen (elco's en voedingen) is de pluszijde aangegeven. Verder vindt u ook het schema, zodat u weet hoe de componenten met elkaar verbonden zijn.

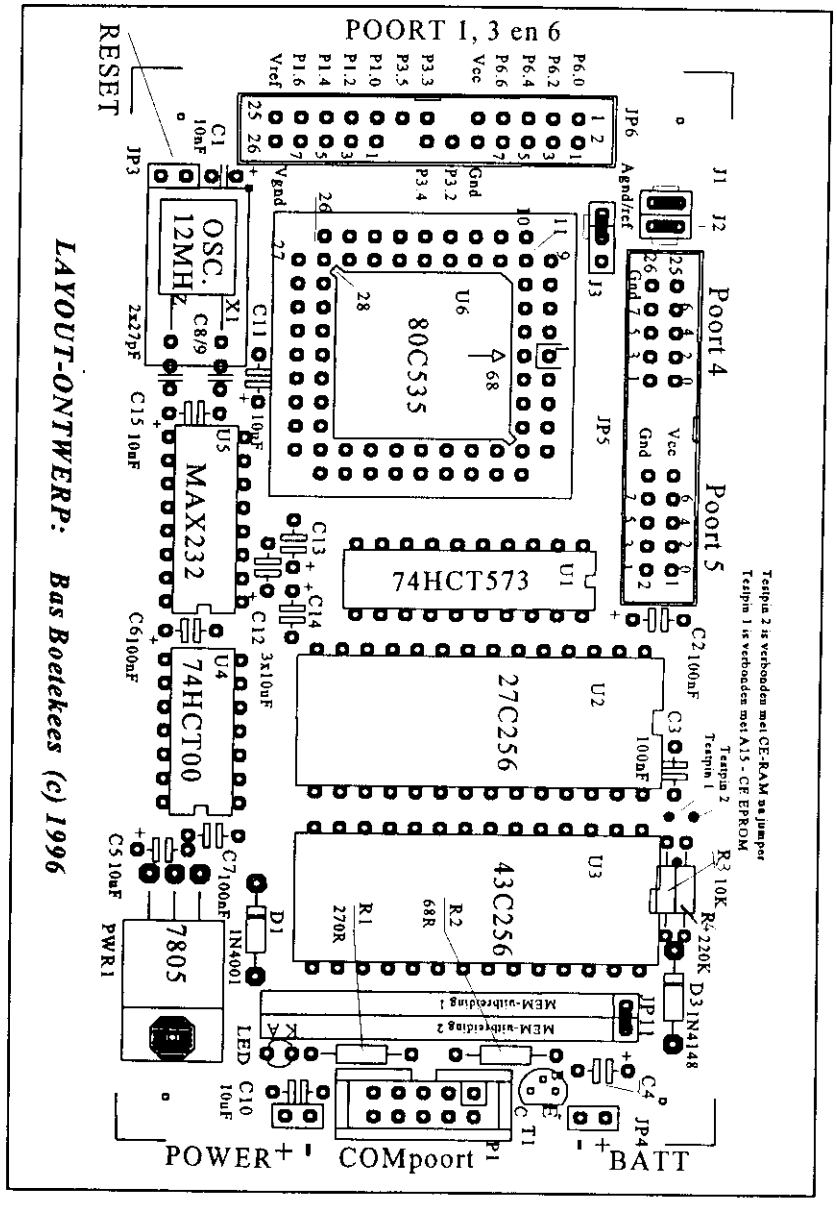
Ik had u ook willen vertellen, dat het bordje werkt. Helaas kan ik dat niet. Sorry, ik heb mijn best gedaan, maar het project loopt uit en is dus nog niet klaar nu de kopij voor On-line moet worden ingeleverd. Desalniettemin hoop ik een bord onder de taal *forth* (want ik heb geen B+-eprom) de komende dagen gereed te krijgen, die op de NewBraindag op 27 april 1996 gedoopt zal worden bij een kopje koffie. Als het kan natuurlijk met gebak, maar zonder doopwater, want dan is onze enige werkende print ook verleden tijd.

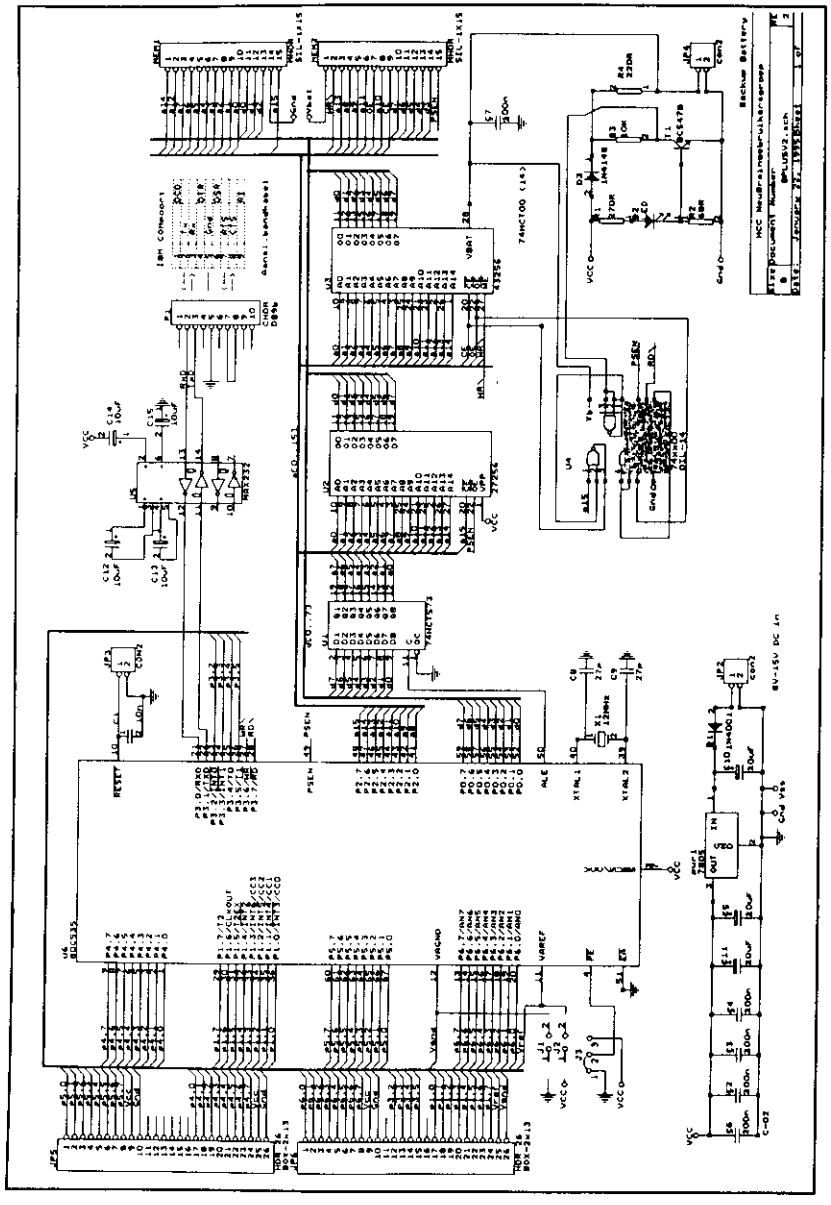
De levertijd van de B+-bordjes zal twee weken bedragen, als alle componenten aanwezig zijn. Op NewBraindagen, gebruikersdagen van de DOS-gg, HCC-dagen en dergelijke kunt u de bordjes ook bestellen en direct meenemen tegen contante betaling. Er geldt een speciale korting voor de leden van de NewBrain-gebruikersgroep.

Veel succes met B+ en laat eens wat zien, wat u ermee gemaakt hebt, want daarvoor ben ik vanaf begin januari to april 1996 met dit project bezig geweest.

Bas Boetekees

nummer	type	nummer	type
U1	74HCT573	IC-voet	DIL-20
U2	27C256		DIL-28
U3	43256		DIL-28
U4	74HCT00		DIL-14
U5	MAX232N		DIL-16
U6	80C535		PLCC-68
PWR1	7805	C1	CRM 10nF
R1	270R	C2, 3, 4, 6, 7	CAP 100nF 5x
R2	68R	C8, 9	CAP 27pF 2x
R3	10K	C5, 10, 11	TANT 10μF 3x
R4	220R	C12, 13, 14, 15	TANT 10μF 4x
T1	BC547B	D1	1N4001
X1	CRIST 12M	D2	LED
P1	2x05 PINS	D3	1N4148
JP5	2x13 PINS	(J1+2+3)	1x07 PINS
JP6	1x13 PINS	(JP2+3+4)	1x06 PINS





hijskraan

Herman van Baarzel heeft van Mecano-onderdelen een hijskraan gebouwd, die bestuurd wordt door een B+-controller. Het is een kraan, zoals die in het verleden in havens en op bouwplaatsen werd gebruikt. De snelheid waarmee de motoren draaien, kan worden beïnvloed door de stroom niet constant maar pulsgewijs aan te voeren. Door de breedte van de puls te verkleinen of te vergroten, draaien de motoren langzamer, dan wel sneller.
In dit artikel wordt de beschrijving gevolgd door de volledige programmalisting, met tot slot, van de hand van Jan Wubben, enige uitleg bij de compacte code van de B+-taal, voor wie daar niet zo vertrouwd mee is.

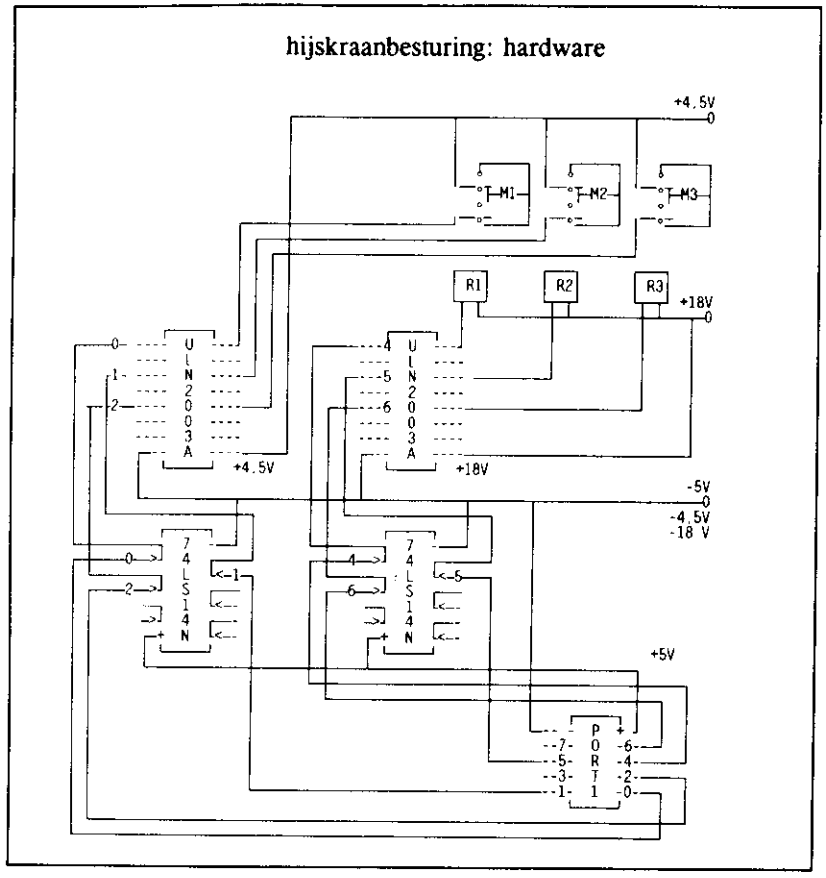
hardware

Deze zogenaamde tophijskraan heeft 3 motoren:

motor1 voor het hijsen en zakken
motor2 voor het heffen en dalen van de arm
motor3 voor linksom en rechtsom draaien

De motoren worden bestuurd vanaf poort P1 via twee 7414 buffers en twee L293D IC's (zie ook het B+ Applicatieboekje). Deze 7414 IC's zijn omkeerbuffers. De pennen van poort P1 zijn normaliter hoog, op +5 V, waarbij de uitgangen van de buffers dus laag zijn. Bij het inschakelen van B+ gebeurt er dus niets.

Poort P1 was bij voorbaat uitverkoren om de drie motoren en de magneetspoel te besturen, omdat het ingebouwde B+-pulsbreedtemodulatiesysteem voor vier motoren werkt op de eerste vier pennen van poort P1, dat wil zeggen op de bits P10, P11, P12 en P13.



De 'hijshaak' bestaat uit de combinatie van een permanente magneet en een magneetspoel. De permanente magneet pakt de last (een blikken kubusje) en houdt hem vast. Bij bekrachtiging neutraliseert de spoel het veld van de permanente magneet, zodat de last wordt gedropt.

Zonder pulsbreedtemodulatie worden de drie motoren gewoon aangezet met de bits Q10-, Q11- en Q12-, terwijl Q13- de magneetspoel bekrachtigt voor het droppen van de last, als volgt:

Q10+/- = motor 1 aan/uit Q14+/- = motor 1 omlaag/omhoog
Q11+/- = motor 2 aan/uit Q15+/- = motor 2 omlaag/omhoog
Q12+/- = motor 3 aan/uit Q16+/- = motor 3 linksom/rechtsom
Q13+/- = last hangt/valt Q17 = ongebruikt

signaleringen

Op de poorten P4 en P5 ontvangt B+ in beginsel voor elke motor drie soorten signaleringen, namelijk telpulsen en signalen die het bereiken van de hoogste en van de laagste toegelaten stand aangeven. De signaleringen houden steeds in dat de betreffende pen van register 4 of 5, die normaliter op +5 V staat, aan aarde wordt gelegd.

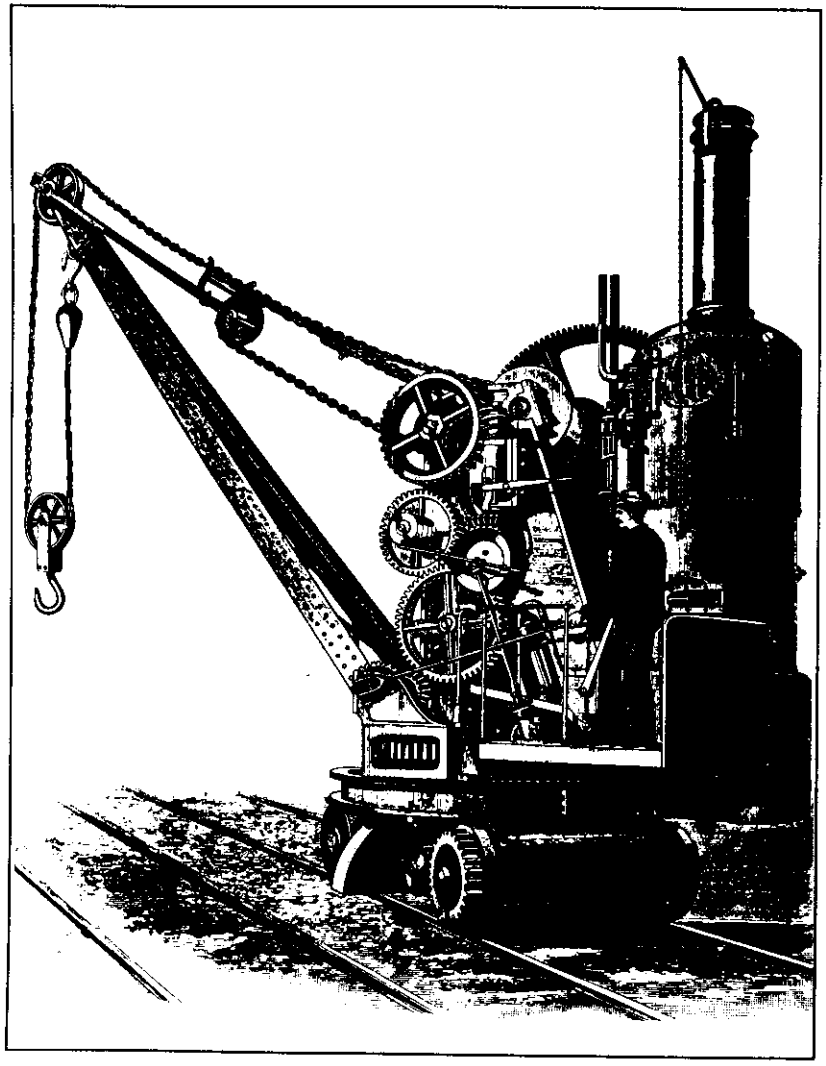
telpulsen

De telpulsen worden opgewekt door een reedrelais, dat zoals bekend door een magneetveld wordt gesloten. Op de hijsas is een koperen tandwiel bevestigd, dat door de motor 1 wordt aangedreven. Op dit tandwiel is met siliconenkit een permanent staafmagneetje geplakt. Evenwijdig aan het tandwiel is een reedrelais opgesteld, dat bij elke omwenteling van het tandwiel eenmaal wordt gesloten door het passerende magneetje. Het reedrelais ligt aan de ene zijde aan aarde, terwijl de andere zijde met pen P40 (bit 0 van poort 4) is verbonden. P40 ontvangt dus voor iedere omwenteling van de hijsas een telpuls. Aangezien een omwenteling van de hijsas overeenkomt met ruwweg 1 cm kabel, vormen de telpulsen een maat voor de plaats van de haak.

Op soortgelijke wijze worden telpulsen voor motor 2 toegevoerd aan P42 en voor motor 3 aan P45.

signalering van de uiterste standen

De hijskabel wordt gevormd door een dunne geïsoleerde koperdraad, die aan aarde is gelegd. Deze koperdraad zorgt voor de signaleringen. Als de haak te hoog gaat, komt een blank stukje hijsdraad in contact met een



wieltje, dat in verbinding staat met P41 (bit 1 van register 4). Een tweede draad voor het bekrachtigen van de magneetspoel is een losse draad vanaf de kraan direct naar de spoel.

De uiterste standen voor motor 2 worden gesignaleerd door contacten die langs de beweegbare arm van de hijskraan zijn opgesteld. Het signaal voor de laagste stand wordt toegevoerd aan P43 en voor de hoogste stand aan P44. Voor motor 3 worden bij het draaien van de kraan contacten gesloten, die voor de uiterst linksom gedraaide stand aan P46 wordt toegevoerd en voor de uiterst rechtsom gedraaide stand aan P47.

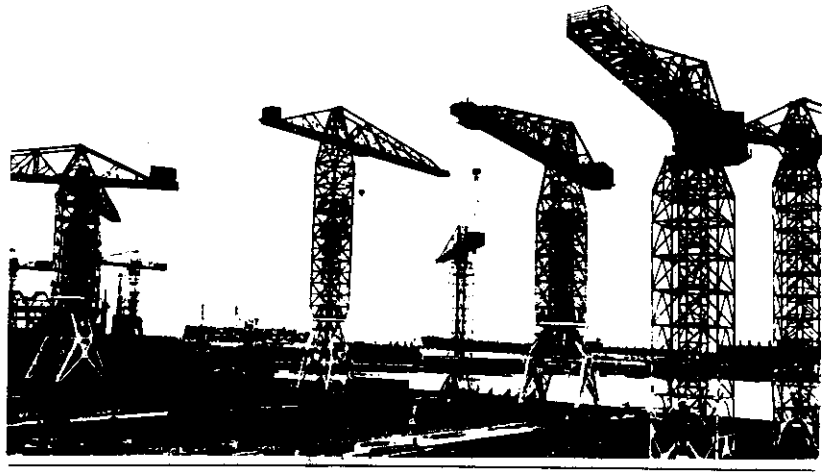
De onderste grens voor de hijsmotor 1 is iets gecompliceerder. Op de grondplaat zijn 8 cirkel- en 64 stervormige geleidende lijnen aangebracht. De 8 cirkelvormige lijnen zijn met de pennen van poort P5 verbonden voor het leveren van de signaleringen.

Als een door de haak vastgehouden kubusje de grondplaat raakt, wordt deze via de geaarde hijsdraad en de geleidende kubus aan aarde gelegd. De computer test 'continue' of P5 = FF, en zo nee, dan is een kubus op de grondplaat geland (eventueel kan de computer ook vaststellen op welke cirkelvormige geleider de kubus is geland).

programmering

In beginsel test de computer continue of er telpulsen optreden; bij het hijsen en zakken wordt dus getest op Q40. Als de kraan hijst, wordt echter eerst steeds getest op het bereiken van het hoogste toegelaten punt, dus op Q41. In de kern zou het (niet goed werkende) basisprogramma er ongeveer zo uitzien:

```
IF
X
   1  S1 = "\n telpuls gedetecteerd"
   2  C1 ,!S1 ,!C0
   3  B
   4  V1=1  Q10-
   5  L0
   6  Q41 ,<@  Q10+ ,!L2
   7  Q40 ,>@  V1:0 ,=C1  V1=1 ,!L0
   8  V1=0 ,!L0
   9  L2
X
```



Op regel 4 wordt motor 1 aangezet. Zodra het hoogste punt is bereikt, wordt Q41 laag. De test op regel 6 is dan raak, zodat met Q10+ motor 1 wordt uitgezet en het programma stopt. Als het hoogste punt nog niet is bereikt, is Q41 hoog, zodat naar regel 7 wordt gestapt. Met Q40 hoog worden regels 5, 6 en 7 steeds herhaald, zonder dat er iets gebeurt.

Als een telpuls optreedt waarbij Q40 laag wordt, wordt naar regel 8 gestapt waar V1 = 0 wordt, maar er wordt teruggegaan naar label L0, zodat de lus niet verder komt zolang de telpuls duurt. Als de telpuls afgelopen is en Q40 weer hoog wordt, wordt regel 7 doorlopen met V1 = 0, zodat call C1 wordt gemaakt, waarbij de opgeroepen S1 de telpulsdetectie op het scherm aankondigt. Daarna gaat de test gewoon door.

De tests voor de verschillende motoren kunnen achter elkaar worden gezet. De computer is zo snel, dat dit geen probleem oplevert (zie L10 - L40 in het programma).

Herman van Baarzel

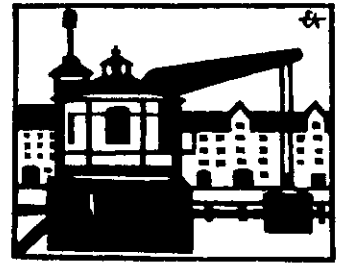

```
IF

X
; vast automatisch programma of separate opdrachten
; 3 motoren + drop-last werken tegelijk
; reedrelais met telpulsdetectie via software flip-flop
; pseudo-interrupties op uiterste standen
; snelheidsregeling door pulsbreedtemodulatie
; zoekbewerking

---

S0  = "\n      H Y S K R A A N P R O G R A M M A  333 "
S1  = "\n telstand 1 = \V1 "
S2  = "\n telstand 2 = \V2 "
S3  = "\n telstand 3 = \V3 "
S4  = "\n interruptie 1 "
S5  = "\n interruptie 2 "
S6  = "\n interruptie 3 "
S7  = "\n interruptie 4 "
S8  = "\n bypass "
S9  = "\n    "
S10 = "\n hysbewerking klaar "
S11 = "\n armbeweging klaar "
S12 = "\n draaien klaar "
S13 = "\n droplast klaar "
S15 = "\n zoekbewerking klaar "
S14 = "\n opdracht \VA uitgevoerd "
S1A = "\n A = Automatisch hijsprogramma "
S1B = "\n B = Handbediening (separate opdrachten) "
S1C = "\n C = Blokjes opzoeken en stapelen "
S1D = "\n D = Continu bedrijf "
S1E = "\n E = Exit;  Enter A, B, C, D of E \R "
S20 = "\n 0 = motor \VC omlaag/linksom "
S21 = "\n 1 = motor \VC omhoog/rechtsom "
S22 = "\n Moet motor \VC omlaag/linksom/drop of
      omhoog/rechtsom gaan? \R "
S23 = "\n Hoeveel stappen moet motor \VC maken ? \R "
S24 = "\n Geef de snelheid van motor \VC , 00-FF? \R "
S30 = "\n Interuptie Q50 blokje op 1e ring "
```

```
S31 = "\n Interuptie Q51 blokje op 2e ring "
S32 = "\n Interuptie Q52 blokje op 3e ring "
S33 = "\n Interuptie Q53 blokje op 4e ring "
S34 = "\n Interuptie Q54 blokje op 5e ring "
```

```
C1 V1-1 ,!S1 ,!C0
C2 V2-1 ,!S2 ,!C0
C3 V3-1 ,!S3 ,!C0
C4 V4-1 ,!C0
C5 PCB=00 Q14+ V11=0 VB-1 ,!S10 ,!C0 ;Q10+
C6 PC3=00 Q15+ V21=0 VB-1 ,!S11 ,!C0 ;Q11+
C7 PC5=00 Q16+ V31=0 VB-1 ,!S12 ,!C0 ;Q12+
C8 PC7=00 Q17+ V41=0 VB-1 ,!S13 ,!C0 ;Q13+
C9 PC5=00 Q16+ V31=0 ,!C0

C20 V9=14 ,!S30 ,!C0
C21 V9=0D ,!S31 ,!C0
C22 V9=09 ,!S32 ,!C0
C23 V9=05 ,!S33 ,!C0
C24 V9=02 ,!S34 ,!C0

CA Q50,<C20 Q51,<C21 Q52,<C22 Q53,<C23 Q54,<C24
Q55,<C25
,!C10 VA=9 V2=V9 V6=0  ,!C11 ;arm-V9
,!C10 VA=A      V1=24 V5=0    ,!C11 ;hys-24 pak-last
,!C10 VA=B      V1=1B V5=1    ,!C11 ;hys+1B
,!C10 VA=C V27=80  V2=20 V6=0 V3=18 V7=0 ,!C11
;arm-20 drl-18
,!C10 VA=D      V1=13 V5=0    ,!C11 ;hys-13
,!C10 VA=E     V1=15 V5=1  V4=01 V8=0 ,!C11 ;hys+15
drop-last
,!C10 VA=F V2=08 V6=1  ,!C11 ;arm+08
V31=1 ,!C0

C10 V1=0 V2=0 V3=0 V4=0
   V25=FF V26=FF V27=A0 V28=FF ,!C0
C11 VB=4 V11=1 V21=1 V31=1 V41=1 P8=01 PC1=AA
,!L5 ,!C0
```

```
B

L1 VE=3 VD=1
,!S9 ,!S0 ,!S9
,!S1A ,!S1B ,!S1C ,!S1D ,!S1E ,!S9 V6=V7F
V6:A,=L2 V6:B,=L3 V6:C,=L4 V6:D,=L9 V6:E,=L70 ,!L1

L9 VD=0 ,!L4

L2
,!C10 VA=1  V2=20 V6=0 V3=30 V7=0 ,!C11 ;arm-20
draail-30
,!C10 VA=2 V1=15 V5=0    ,!C11 ;hys-15 pak-last
,!C10 VA=3 V1=10 V5=1 V2=20 V6=1 V3=30 V7=1 ,!C11
;hys+10 arm+20 draair+30
,!C10 VA=4 V1=23 V5=0    ,!C11 ;hys-23
,!C10 VA=5 V1=08 V5=1    V4=01 V8=0 ,!C11 ;hys+08
drop-last
,!C10 VA=6 V1=17 V5=1 V2=10 V6=0 V3=10 V7=0 ,!C11
;hys+17 arm-10 draail-10
VE-1,#L2 VD:0,=L4 ,!L1

L3 ;separate opdrachten voor 4 motoren tegelijk
VC=1 ,!S20 ,!S21 ,!S22 V5=V7F ,!S23 V1=V7F ,!S24
                                          V25=V7F
VC=2 ,!S20 ,!S21 ,!S22 V6=V7F ,!S23 V2=V7F ,!S24
                                          V26=V7F
VC=3 ,!S20 ,!S21 ,!S22 V7=V7F ,!S23 V3=V7F ,!S24
                                          V27=V7F
VC=4 ,!S20 ,!S21 ,!S22 V8=V7F ,!S23 V4=V7F ,!S24
                                          V28=V7F
,!C11 ,!L1

L4
  ,!C10 VA=1  V3=35 V7=0 ,!C11 ;drl-35
L6 ,!C10 VA=1 V2=20 V6=1  ,!C11 ;arm+20
  ,!C10 VA=2  V3=30 V7=1 ;,!C12 drr+30 zoek...
   V31=1 P8=01 PC1=AA Q16- PC5=90
```

```
L7  P5:FF,=@ PC5=00 Q16+  VB=4 V21=1  ,!CA ,!S7 ,!L6
;,!C9
   Q45,>@ V30:0,=C3 V30=1 V3:0,=C9 V31:0,=L8 ,!L7
   V30=0 ,!L7
L8
,!C10 VA=C       V1=06 V5=1 V2=14 V6=0 V3=10 V7=0
,!C11 ;hys+6 arm-14 drl-10
VD:0,=L2 ,!L1
```



```
L5
V1:0,#@ ,!C5
V2:0,#@ ,!C6
V3:0,#@ ,!C7
V4:0,#@ ,!C8

L10 V11:0,=L20
   V5:0,#@ Q14+ PCB=V25 ,!L11  ;Q10-
   Q14- PCB=V25       ;Q10-
   Q41,<@ ,!S4 ,!C5 ,!L20
   ,!L12
L11 P5:FF,=@ ,!S4 ,!C5 ,!L20
L12 V13-1,#$ Q40,>@ V10:0,=C1 V10=1 V1:0,=C5 ,!L20
   V10=0

L20 V21:0,=L30
   V6:0,#@ Q15+ PC3=V26 ,!L21  ;Q11-
   Q15- PC3=V26       ;Q11-
   Q44,<@ ,!S5 ,!C6 ,!L30
   ,!L22
L21 Q43,<@ ,!S5 ,!C6 ,!L30
L22 V13-1,#$ Q42,>@ V20:0,=C2 V20=1 V2:0,=C6 ,!L30
   V20=0

L30 V31:0,=L40
   V7:0,#@ Q16+ PC5=V27 ,!L31  ;Q12-
   Q16- PC5=V27       ;Q12-
   Q47,<@ ,!S6 ,!C7 ,!L40
   ,!L32
L31 Q46,<@ ,!S6 ,!C7 ,!L40
```



```
L32 V13-1,#$ Q45,>@ V30:0,=C3 V30=1 V3:0,=C7 ,!L40
   V30=0

L40 V41:0,#@ ,!L50
   PC7=V28 Q17+ V13=0 V14=0    ;Q13-
   V13-1,#$ V14-1,#$ V4=0 ,!C8

L50 VB:0,#@ P01=FF ,!S14 ,!C0
   ,!L10

L70 PCB=00 PC3=00 PC5=00 PC7=00 PC1=00 P8=00
   P01=FF
X
```

verklarende aantekeningen bij de programmalisting

IF betekent *Input Fast*; het wordt gebruikt om programma's, die bijvoorbeeld al op schijf staan, snel te laden.

X geeft het begin (en het einde) van een *programma* aan.

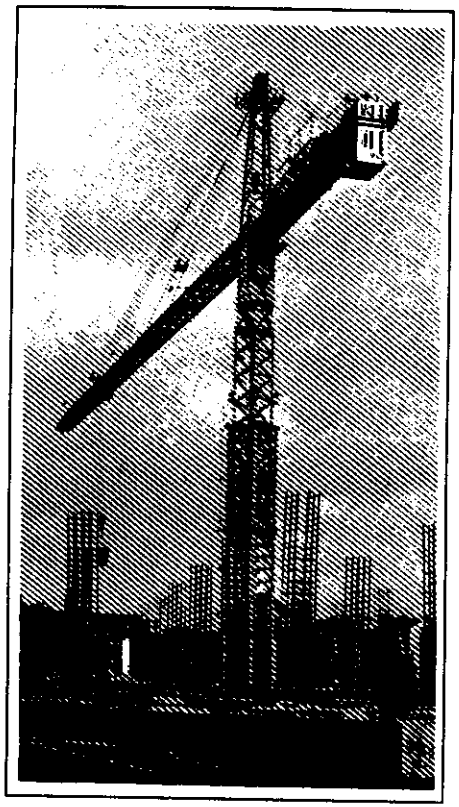
; geeft het begin aan van *commentaar*, precies zoals een REM-statement in basic. De compiler doet verder niets met de rest van de regel.

Snn definieert een *ascii-string*, met het erbij behorende nummer. Mits in hexadecimale notatie gedefinieerd zijn er 254 strings mogelijk (van S0 tot en met SFE). Sommigen vinden het gemakkelijker alleen de hexadecimale nummers te gebruiken, waarin geen letters voorkomen. Dan lijken het decimale nummers, maar het aantal strings is dan beperkt tot 99 stuks (100 als ook S00 gebruikt wordt). In alle strings samen mogen maximaal 3840 tekens worden gebruikt.
Binnen de strings kunnen parameters worden meegegeven; deze worden steeds vooraf gegaan door een backslash, dus de \. Zo betekent de \n, 'geef line-feed'.
Bij \Vnn wordt de waarde van de betreffende variabele afgedrukt.
Bij \R wordt naar de betreffende poort gekeken en de daar gevonden waarde afgedrukt.

Cnn definieert een *call* naar een subroutine. Een subroutine is een stuk(je) programma, dat meerdere malen gebruikt gaat worden. Er zijn er slechts 99 mogelijk (zie het handboek).
Overigens C0 is een sprong terug, naar de plaats waarvandaan de vorige call werd gemaakt; in feite een *return!*

Vnn declareert een *variabele*. Hiervan zijn er 127 mogelijk, mits hexadecimaal genoteerd; anders weer slechts 99.

Het uitroepteken, voorafgegaan door de komma, wordt gebruikt als een soort test, waarvan de uitkomst altijd *waar* is, zodat altijd de sprong achter de test wordt uitgevoerd.



Zo betekent de regel:

C1 V1-1 ,!S1 ,!C0

- dit is call numero 1
- verlaag de waarde in variabele numero 1 met 1
- print de string numero 1
- ga terug naar waar je vandaan bent gekomen

Pnn bestuurt de *poorten*, dan wel kijkt ernaar.

Q zet de verschillende bits in de poorten per stuk aan, dan wel af (zet in feite *schakelaars* aan of uit).

Laat u niet in de war brengen door het niet normaal doorlopen van nummeringen. Als u ineens CA tegenkomt is dat call A (hexadecimaal) en dat is call 11 (decimaal). Zie hiervoor de nummering van de strings, die

tot en met S1E oplopend is; daarna verwacht je S1F, maar die zal om een of andere reden niet meer gebruikt worden.
U mag nu proberen de decimale nummers van de poorten en andere variabelen in te vullen. Nog eenmaal zal ik u helpen: als u ziet staan VA = A, of VA = B, dan staat daar: 'de elfde variale krijgt de waarde elf', of 'de elfde variabele krijgt de waarde twaalf'.

Bij B begint het eigenlijke programma; alles wat daarvoor staat zijn declaraties en definities.

Lnn is de definitie van een *label*, dat dient als sprongadres. Er zijn 127 labels mogelijk. Binnen deze labels zijn twee tekens vermeldenswaard, omdat zij niet als label gedefinieerd behoeven te worden en toch als zodanig kunnen worden gebruikt:

@ betekent: sla de volgende instructies over en ga naar het begin van de volgende regel
# betekent: ga terug naar het begin van deze regel

Het is niet de bedoeling om het hele B+-handboek (ruim 50 pagina's) hier te behandelen, maar een paar dingen wil ik nog kwijt.

- tests worden voorafgegaan door een komma
- verder worden de volgende tekens gebruikt:
  - = is gelijk aan
  - # is niet gelijke aan
  - > is groter dan 0
  - < is kleiner dan 0
  - ! altijd waar

Met de hierboven gegeven uitleg moet het (min of meer) mogelijk zijn, zonder B+ handboek, het
hijskraanprogramma te volgen. In ieder geval moet duidelijk kunnen worden, hoe het B+-besturingssysteem met de hardware en software functioneert.
Als er nog vragen zijn, dan horen wij dat wel...

Jan Wubben

kunt u solderen? soldeer dan mee!

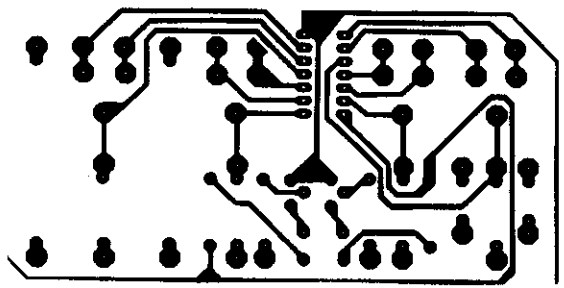
wij zoeken enthousiaste nieuwe leden. doel is computers (en niet alleen de newbrain) te gebruiken voor het besturen van allerlei apparaten en systemen

de belangrijste activiteit zal zijn het bij elkaar brengen van personen en groepen met gelijke belangstelling waarbij uitwisseling van ervaringen voorop staat

daartoe zullen de volgende activiteiten worden ontplooid:

- het (verder) ontwikkelen van interfaces en andere hardware met de bijbehorende software
- het verzamelen van informatie terzake
- het verwerken van die informatie
- het demonstreren met werkende apparaten en systemen
- het beschikbaar stellen van informatie onder meer in de vorm van publicaties
- het bevorderen van activiteiten leidende tot bovengenoemde doelstellingen

wie?

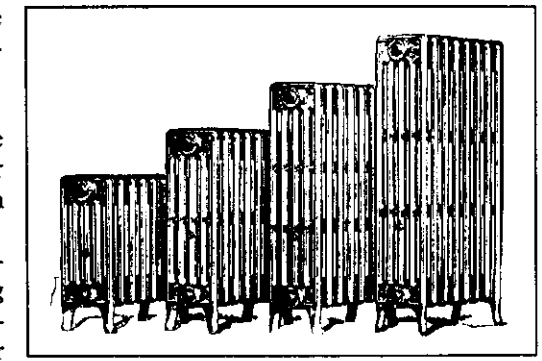


hcc newbrain-gebruikersgroep
postbus 94494
1090 GL amsterdam
telefoon (010) 4557698 / (020) 6924137

thermostaat

B+ CV-regeling

Een thermostaat regelt de kamertemperatuur in uw woning, mits je natuurlijk over een eigen CV-installatie beschikt. In allerlei doe-het-zelfzaken en bij de vakhandel zijn verschillende thermostaten te koop. Is uw oog gevallen op een thermostaat met geheugenfuncties, dan moet je hiervoor toch al gauw enige tientallen guldens neertellen.
B+-gebruikers, of toekomstige gebruikers, kunnen overwegen zelf een CV-regeling te ontwerpen met als basis een B+-bord.



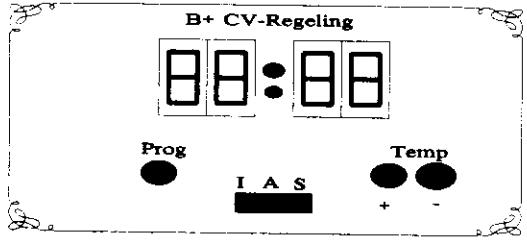
Wat zijn de minimale eisen waaraan een CV-regeling moet voldoen:

- De ingestelde en de actuele temperatuur worden getoond op een display.
- Signalering of de kamertemperatuur te laag is, en dus of de CV-ketel op temperatuur gebracht of gehouden moet worden.
- Instellen van de maximale temperatuur, meestal * 20 graden Celsius
- Instellen van de minimale temperatuur, meestal * 15° C
- Instellen van de save-stand: de kamertemperatuur wordt op de minimale temperatuur gehouden; de maximale temperatuur wordt genegeerd (bijvoorbeeld als je niet aanwezig bent).
- Instellen van de perioden dat de maximale en de minimale temperatuur moet worden bewaakt.

Natuurlijk kunnen er nog een aantal functies, mogelijk eisen, aan worden

toegevoegd. Omdat het B+-bord is te programmeren, behoeft dit geen bezwaar te zijn. Wat in eerste instantie bepaald moet worden, is welke hardware (displays, schakelaars en drukknoppen) benodigd is om een zo voordelig mogelijk en goed functionerend bordje te maken. Software kan altijd aangepast worden zonder kosten. Hardware kost extra geld en als het tegen zit, moet er soms een nieuwe print gemaakt worden en dan blijken heel vaak de oude onderdelen onbruikbaar.

Mijn eerste gedachten over de lay-out zijn de volgende:

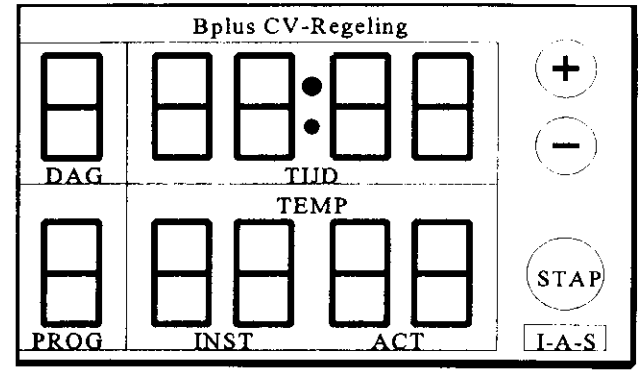


De linker twee cijfers geven de actuele temperatuur aan en de rechter twee de ingestelde minimale of maximale temperatuur.
Wanneer er op de knop Prog wordt gedrukt, verschijnt in het rechter display een speciaal teken, codering 10 tot en met 15, die het mogelijk maakt de programmatijden aan te passen. Wanneer alle displays knipperen kan bijvoorbeeld de actuele tijd worden ingesteld.
Het geschatte bedrag aan hardware componenten zal ongeveer *f* 50,00 bedragen.
In het hierboven aangegeven bordje zijn niet al te veel componenten aanwezig om een uitgebreide instellingsprocedure via de toetsen Prog, + en - in te voeren. Maar het zou best met enig denkwerk en een slim software-programma, al of niet via de compoort te besturen zijn.

Een veel royalere uitvoering vereist meer displays. Hieronder vind je de lay-out.

Dat is even schrikken. Maar de mogelijkheden van zo'n display zijn zeer groot. De kaders geven groepen aan, die de gebruiker allemaal met de rechterknoppen kan instellen.

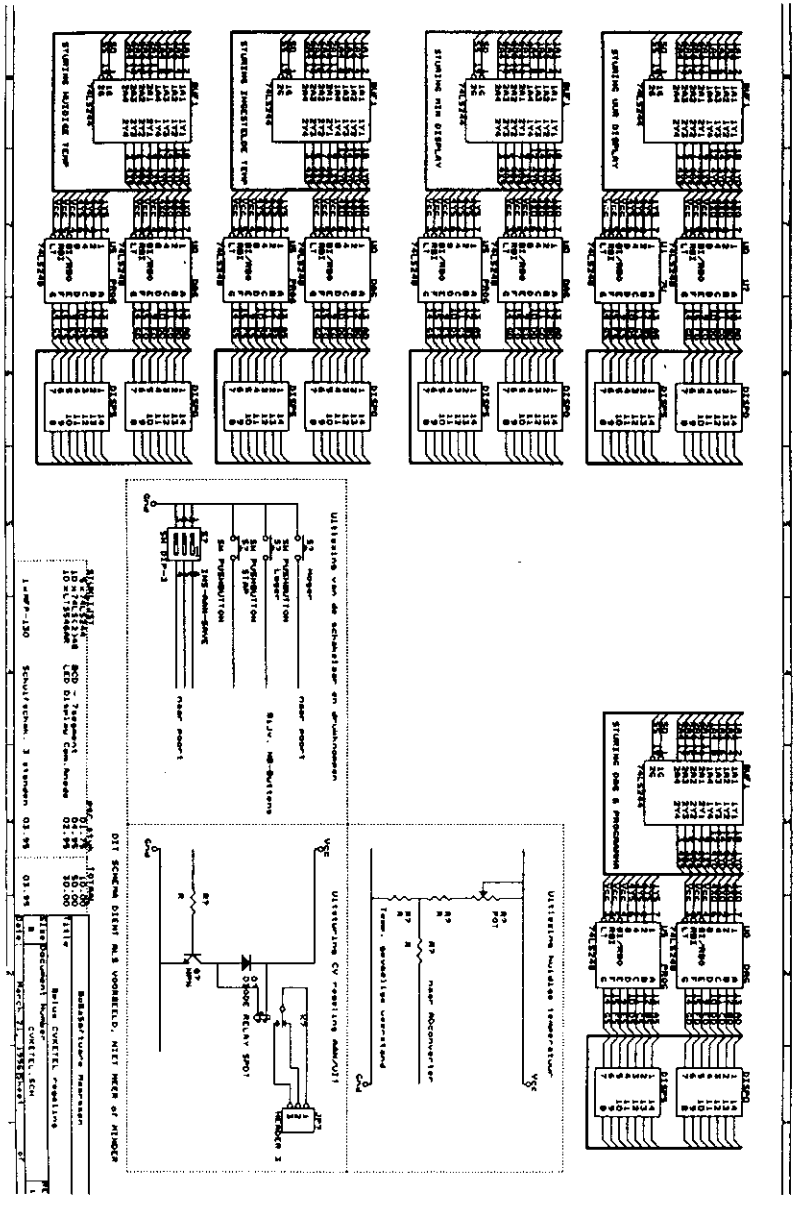
In dit voorbeeld ga ik uit van een programma-omgeving die het mogelijk maakt voor iedere dag een afzonderlijk programma in te toetsen. De tabel hieronder geeft een voorbeeld van de in te stellen tijden.



	programma 1			programma 2			programma 3			programma 4		
dag	IN	UIT	°C	IN	UIT	C	IN	UIT	C	IN	UIT	C
1 = ma	07.00	10.00	20	12.00	15.00	21	15.00	22.00	22	22.00	24.00	20
2 = di	07.00	10.10	20	12.15	16.00	21	16.00	22.00	22	22.00	23.00	20
3 = wo	07.00	10.20	20	12.30	16.00	21	16.00	22.00	22	22.00	23.00	20
4 = do	07.00	10.30	20	12.45	16.00	21	17.00	22.00	22	22.00	23.00	20
5 = vr	07.30	10.40	20	13.00	16.00	21	17.00	22.00	22	22.00	23.00	20
6 = za	09.00	12.00	20	14.00	18.00	21	18.00	22.00	22	22.00	02.00	22
7 = zo	10.00	12.00	20	15.00	18.00	21	18.00	22.00	22	22.00	24.00	22

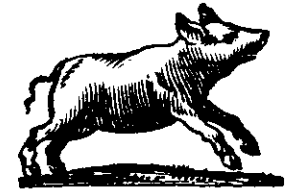
Wanneer je dit leest, zul je misschien denken, *waar ik dit bestellen?* Nergens, is mijn antwoord. Dit artikel dient alleen om geïnteresseerden rond de tafel te krijgen en er voor de komende maanden een projectgroep van te maken, die voor de landelijke HCC-Dagen in Utrecht, een werkende CV-regeling maakt rond het B+-bord.
Geïnteresseerd? Geef je dan op bij onze gebruikersgroep. We kunnen dan kijken of we op afdelingsvergaderingen bij elkaar kunnen komen om te zien, hoe we het project starten enzovoort.

Bas Boetekees

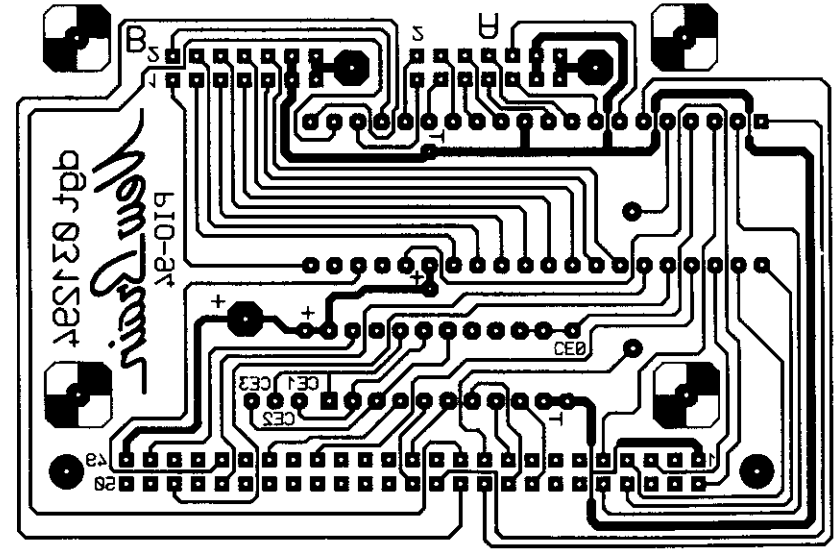


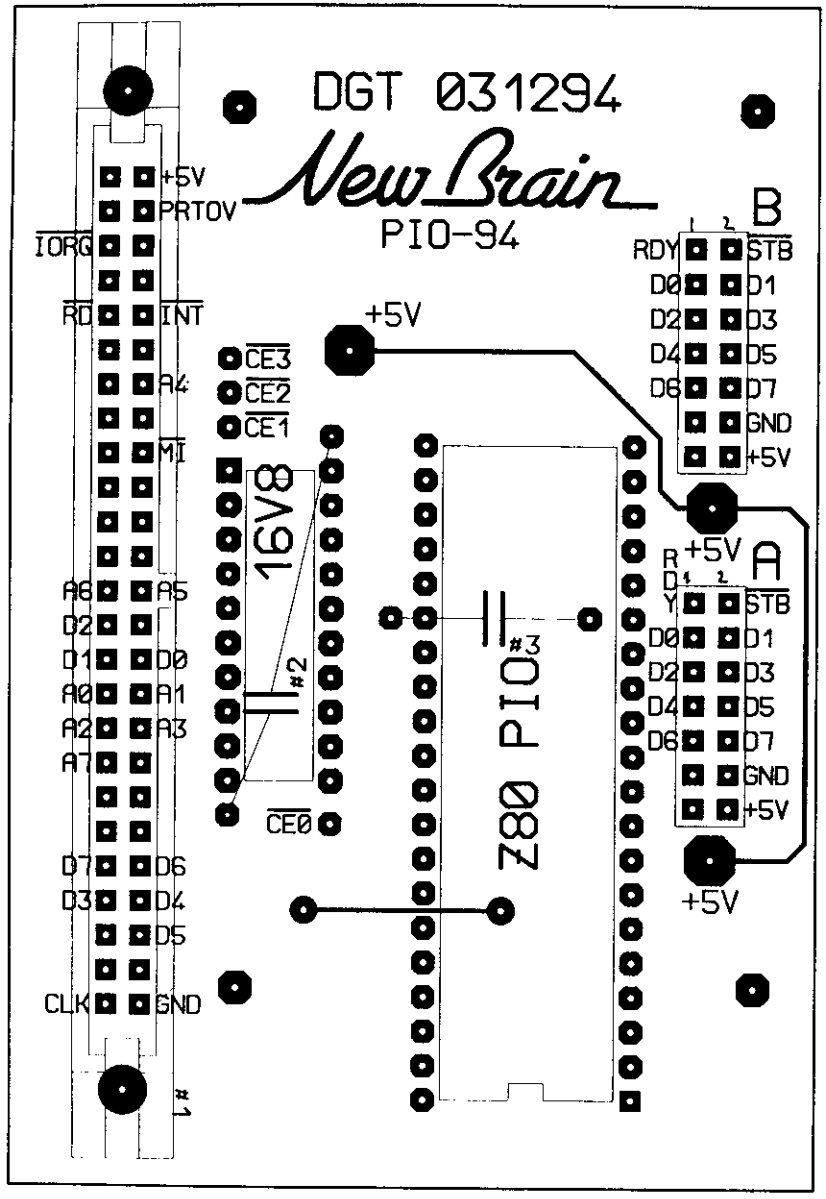
pio

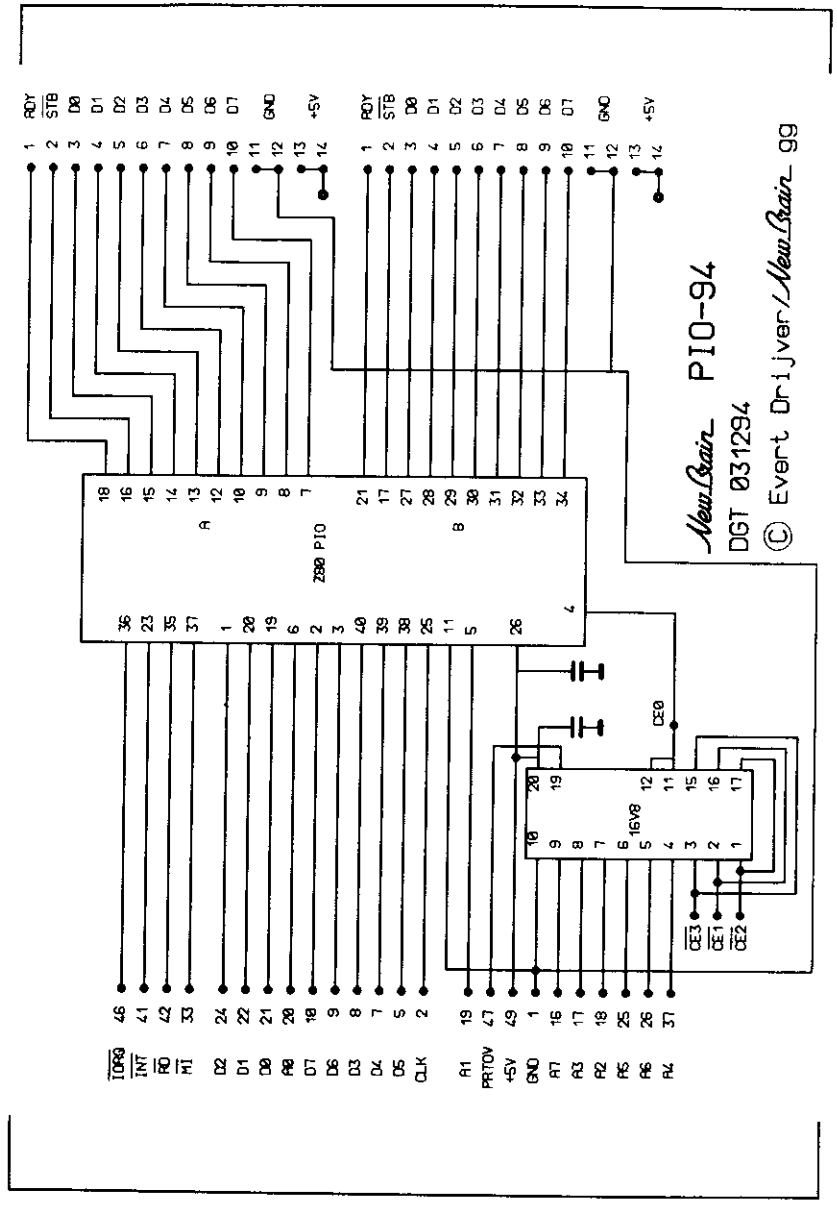
het ontwerp



in newbrain on-line 19 (pagina 2) heeft evert drijver een beschrijving gegeven van zijn voordelige i/o op basis van de pio. nu heeft hij ook het schema en de printlay-out ervan beschikbaar gesteld. die drukken we hierbij af




DGT 031294
New Brain
PIO-94
+5V
PRTOV
IORG
RD
INT
A4
MI
A6
A5
D2
D1
D0
A0
A1
A2
A3
A7
D7
D6
D3
D4
D5
CLK
GND
CE3
CE2
CE1
CE0
16V8
#2
#3
Z80 PIO
B
RDY
STB
D0
D1
D2
D3
D4
D5
D6
D7
GND
+5V
A
R
D
Y


RDY
STB
D0
D1
D2
D3
D4
D5
D6
D7
GND
+5V
Z80 PIO
16V8
CE0
CE3
CE1
CE2
IORQ
INT
RD
MI
PRTOV
CLK
A7
A3
A2
A5
A6
A4
New Brain PIO-94
DGT 031294
© Evert Drijver/New Brain gg

inhoud on-line 20