5093

Technische Hogeschool Eindhoven
Vakgroep Meten en Regelen

ELECTRONISCH BRANDSTOF DOSERINGSSYSTEEM

DOOR: J.G.W.M. Oerlemans

Afstudeerverslag van de periode 1-9-1984 tot 29-8-1985,
onder begeleiding van Prof. ir. F.J. Kylstra (THE),
Ir. H.H. van de Ven (THE),
Ing. E.D. van Veldhuizen (Philips)

INHOUD

VOORWOORD

Het afgelopen jaar is voor mij een erg leerzame periode geweest. Ik heb in mijn afstudeerproject een unieke kans gehad te zien wat er allemaal komt kijken om bij de realisering van een project als dit.
Het heeft mij met name aangesproken omdat ik er vanaf het begin erg nauw bij betrokken ben geweest. Vanaf het schrijven van de specificaties totaan het geteste systeem zoals dat nu bestaat. Ik denk dat ik ook in de toekomst veel profijt zal hebben van de opgedane ervaring.

Het is niet mogelijk iedereen die bij mijn afstudeerverslag is betrokken is geweest hier te bedanken, maar een aantal wil ik toch graag op deze plaats toch zeker vernoemen.

Op mijn werkplaats bij PHILIPS is de dagelijkse begeleiding in handen geweest van Ing.E.van Veldhuizen. Samen met zijn collega A.Korteling heeft hij mij de nodige ondersteuning gegeven bij alles wat met het project te maken heeft.
Mijn contacten met VIALLE liepen grootdeels via Ing.A.v.d.Wildenberg. Ook hij heeft, vooral in de laatste maanden, veel tijd besteed aan mijn begeleiding.
Op de TH heb ik bij mijn afstudeerhoogleraar Prof.Ir.F.Kylstra en mijn begeleider Ir.H.v.d.Ven regelmatig de vorderingen van het project kunnen toetsen aan hun inzichten. Dat gaf soms aanleiding tot heroverwegen van genomen beslissingen hetgeen de kwaliteit van het project zeker ten goede is gekomen.

SAMENVATTING

Het mengen van brandstof en lucht is een essentieel deel van elke verbrandingsmotor. Wanneer men een universeel systeem wil maken dat brandstof en lucht kan mengen voor een groot aantal verschillende typen motoren, doet zich al snel een probleem voor. De gewenste lucht-brandstof-verhouding is namelijk niet alleen afhankelijk van de belastingstoestand, maar is ook sterk afhankelijk van het type motor.
Omdat het met zuiver mechanische componenten moeilijk is de mengselbereiding nauwkeurig in alle punten van het kenveld te realiseren wordt gezocht naar een mogelijkheid dit met behulp van een microprocessorgestuurd regelsysteem te bereiken.

Een eerste poging om tot een dergelijk ontwerp te komen heeft geresulteerd in een systeem dat op grond van een aantal gemeten grootheden een eerste benadering van de gewenste hoeveelheid brandstof doet. Aan de hand van het gemeten verbrandingsresultaat kan het systeem de hoeveelheid brandstof dan bijsturen.

Het systeem werkt nog steeds voor een groot deel met de conventionele componenten. Het electronische regelsysteem biedt echter de mogelijkheid het conventionele systeem nauwkeuriger te maken.

SUMMARY

Mixing fuel and air is an essential part of every combustion engine. Trying to make a universal system to mix fuel and air for many different types of engines will present major problems. The target air-to-fuel ratio is not only determined by the engine load, but also by the type of engine.
It is not possible to realise exactly the right mixture under all operating conditions, using mechanical components only. One of the possible solutions to this problem is a control system using a microprocessor.

A first design attempt resulted in a system that uses a few measured parameter values to assess the required amount of fuel. The system will then correct the fuel rate based on an analysis of the exhaust gases.

Conventional engine components still make up the greater part of the total fuel system. Electronic control, however, can improve a great deal on its operating accuracy.

LIJST VAN GEBRUIKTE AFKORTINGEN EN SYMBOLEN

AE: correctie term voor motortemperatuur op restrictiepositie. AE is een functie van Teng.

AH: correctieterm voor handinstelling in het stationaire gebied, functie van Ridle.

ATaf: correctieterm voor verhouding tussen de temperatuur van de lucht en het LPG. ATaf is een functie van Tair/Tfuel.

Dt: intervaltijd tussen twee zuurstofmetingen.

Ef: factor voor extra hoeveelheid brandstof bij smoorklep ver open.

FE: correctiefactor voor motortemperatuur op de gewenste zuurstofconcentratie. FE is een functie van Teng.

FH: eindcorrectiefactor, functie van Rend.

GLM: Gas/Lucht-Menger.

index: adres van ramlokatie die bij motordata wordt gevoegd.

KL15: signaal dat aangeeft dat het contact is ingeschakeld.

KL50: signaal dat aangeeft dat de startmotor wordt bekrachtigd.

lact: actuele waarde van lambda.

lambda: basiswaarde voor de gewenste lambda. Het is een functie van RPS en Pin.

ltar: gewenste waarde van lambda.

LPG/B: signaal dat aangeeft of op LPG of benzine wordt gereden.

nrcyl: aantal cylinders van de motor.

O2bas: basiswaarde voor de gewenste zuurstofconcentratie. O2bas is een functie van RPS en Pin.

O2con: tabel waarmee de gemeten zuurstofconcentratie wordt omgezet naar een werkbare grootheid.

O2low: ondergrens voor de meetbare zuurstofconcentratie.

O2tar: gewenste zuurstofconcentratie in de uitlaat.

Pcon: tabel waarmee de gemeten druk wordt omgezet naar een werkbare grootheid.

Pin: druk in het inlaatspruitstuk

POSadd: het aantal stappen dat moet worden opgeteld bij de gewenste restrictiepositie als gevolg van de terugkoppeling.

POSbas: basiswaarde voor instelling van restrictie bij voorwaartssturing en bij rijke mengsels.

POSdec: positie van restrictie tijdens deceleratie.

POSout: gewenste stand van restrictie.

POSref: signaal van de referentieschakelaar op de restrictie.

POSstrt: stand van restrictie tijdens startfase.

Raf: Raf = Tair/Tfuel

Rend: waarde van de handingestelde eindcorrectiefactor.

Ridle: waarde van de handingestelde stationaircorrectiefactor.

RPS: motorfrequentie.

RPSidle: grensfrequentie van stationair gebied.

RPSon: frequentie waarbij (na frequentiebegrenzing) de brandstoftoevoer weer wordt hervat.

RPSoff: frequentie waarbij begrenzing in werking treedt.

S: aantal stappen waarmee POSadd per Dt kan worden verhoogd of verlaagd.

stpint: intervaltijd tussen twee stappen van de stappenmotor.

tCH choketijd (tijd waarin gas aan de motor wordt toegevoerd zonder dat met starten wordt begonnen). tCH is een functie van Teng.

tmpcon: tabel waarmee de gemeten temperatuur wordt omgezet naar een werkbare grootheid.

Tair: temperatuur van de aangezogen lucht.

Teng: temperatuur van de motor.

Tfuel: temperatuur van het aangeboden brandstof.

THcl: signaal dat aangeeft dat smoorklep dicht staat.

THop: signaal dat aangeeft dat smoorklep ver open staat.

tOP: maximale tijd dat LPG-afsluiters geopend mogen zijn zonder dat een referentiepuls van de motor komt.

VDR: Verdamper-Druk-Regelaar.

LIJST VAN FIGUREN

INLEIDING

Dit afstudeerverslag beschrijft de opzet, de functies en de werking van een electronisch geregeld LPG-systeem. Het beschrijft de eerste stap naar de oplossing voor een aantal problemen die in LPG-instalaties bestaan. Het belangrijkste probleem hierbij is de grote verscheidenheid in typen motoren.
Hiertoe werd een Electronisch LPG Systeem (ELS) ontwikkeld, dat op basis van een microprocessor en met behulp van daarin opgeslagen tabellen en variabelen, invloed heeft op de mengselbereiding.
Bovendien is er een mogelijkheid geschapen om doormiddel van een personal computer de tabellen en variabelen aan de verschillende typen motoren aan te passen. Dit systeem heeft de naam 'Commands' gekregen.

Het project werd uitgevoerd in een samenwerkingsverband tussen VIALLE en CAB ELCOMA (PHILIPS).
Mijn taak hierin was allereerst het schrijven van de specificatie voor de hard- en software van het totale systeem. Dit gebeurde aan de hand van de wensen van VIALLE en met behulp van de ervaring die ELCOMA met soortgelijke systemen heeft.
Vervolgens ben ik werkzaam geweest aan het ontwerp en de realisering van het in die specificatie beschreven systeem.

In hoofdstuk 1 wordt het huidige LPG-systeem beschreven met daarbij de wensen voor het nieuwe systeem. In hoofdstuk 2 worden de belangrijkste ontwerpbeslissingen die zijn genomen toegelicht. Een eerste schets van de oplossing die is gekozen wordt gegeven in hoofdstuk 3.
In de hoofdstukken 4 en 5 wordt de hard- en software van het ELS beschreven. De beschrijving van hard en software voor het 'Commands'-systeem staat in de hoofdstukken 6 en 7. In hoofdstuk 8 worden de testresultaten toegelicht. In hoofdstuk 9 tenslotte, wordt de huidige staat van het ELS beschreven en enkele aanbevelingen voor de toekomst gedaan.

1. PROBLEEMANALYSE

Om een beeld te kunnen geven van de eisen die aan het te ontwerpen Electronische LPG-Systeem worden gesteld lijkt het mij nuttig de componenten van de LPG-installatie, zoals dat nu wordt geleverd, even de revu te laten passeren.
De installatie bestaat uit de volgende hoofd-componenten (zie ook fig. 1-1):

- de LPG-tank;
- de Verdamper-Druk-Regelaar (VDR);
- restrictie tussen VDR en GLM;
- de Gas-Lucht-Menger (GLM).

- In de tank wordt het in vloeibare toestand verkerende gas onder druk opgeslagen. Door deze druk wordt het LPG naar de VDR gebracht.
- De VDR heeft, zoals de naam al doet vermoeden, een tweeledige taak. De eerste is het verdampen van het LPG. Hierbij wordt warmte toegevoerd vanuit de motor. De tweede taak is het afgeven van de gasvormige brandstof met een nauwkeurig constante druk.
- Met behulp van de restrictie in de gasleiding tussen de VDR en de GLM kan de maximale gashoeveelheid met de hand ingesteld worden. Dit is een eenmalige afstelling die door de dealer wordt uitgevoerd.
- Ook de Gas/Lucht-Menger heeft twee taken. Het doorstromen van de lucht door de keel van de venturi veroorzaakt een zekere onderdruk. Deze onderdruk zorgt ervoor dat de eerste taak van de Gas/Lucht-Menger kan worden uitgevoerd, namelijk het aanzuigen van de gasvormige brandstof. De vorm van de venturi bepaalt de onderdruk als functie van de hoeveelheid doorstromende lucht.
De tweede taak van de Gas/Lucht-Menger bestaat uit het maken van een homogeen mengsel van het gas en de lucht. Dit wordt bewerkstelligd door het gas via kleine openingen aan de lucht toe te voegen.

Uit het bovenstaande blijkt dat het voor deze methode essentieel is het

gas ter beschikking te krijgen met een nauwkeurig constante druk. Dit stelt dus hoge eisen aan de VDR.
Een type verdamper-drukregelaar kan echter, mits deze voldoende capaciteit heeft in bijna alle typen automotoren worden toegepast.

Ook aan de Gas/Lucht-Menger worden hoge eisen gesteld. De vorm hiervan bepaalt immers, samen met de stand van de restrictie, de karakteristiek van de gerealiseerde lucht-brandstofverhouding. Vele factoren bepalen de vorm van de GLM. Allereerst moet worden gezorgd dat de invloed van de Gas/Lucht-Menger op het benzinesysteem, dat gehandhaafd bijft, zo klein mogelijk is. Daartoe moet de doorstroming van de lucht zo weinig mogelijk worden gestoord. Daarnaast moet de mengkarakteristiek een optimale werking op LPG opleveren.
Het ontwikkelen van deze Gas/Lucht-Mengers is een moeilijke en daarmee ook kostbare aangelegenheid. Mede omdat kleine veranderingen in de vorm grote gevolgen (kunnen) hebben voor de hele karakteristiek.
De Gas/Lucht-Menger moet voor vrijwel elk motortype speciaal worden ontwikkeld om een optimaal resultaat te verkrijgen op zowel LPG als benzine.

Het ligt dus bijna voor de hand dat gezocht wordt naar een methode om gas en lucht te mengen, waarbij het gemakkelijker is de karakteristiek aan elk motortype aan te passen. Daarvoor zal het mogelijk moeten zijn in delen van de mengkarakteristiek in te grijpen, zonder direct de rest ervan te beinvloeden. Tegelijkertijd zou men moeten proberen de mengkarakteristiek niet al te zeer afhankelijk te laten zijn van de druk van het gas. Daardoor kan men aan die druk minder hoge eisen stellen en zouden de kosten van de VDR gereduceerd kunnen worden.
Ook voor een betere beheersing van de uitlaatgas-emissie van de motor is een nauwkeuriger definieerbare Gas/Lucht-menging van belang. Door het op een juiste manier invullen van karkteristieken in het hele kenveld van de motor kan een optimale lucht-brandstofverhouding worden vastgesteld, waarmee dan een zo volledig mogelijke verbranding van het gas worden verkregen, waarbij de totale uitlaatgasemissie beter voldoet

aan de toekomstige wettelijke eisen.

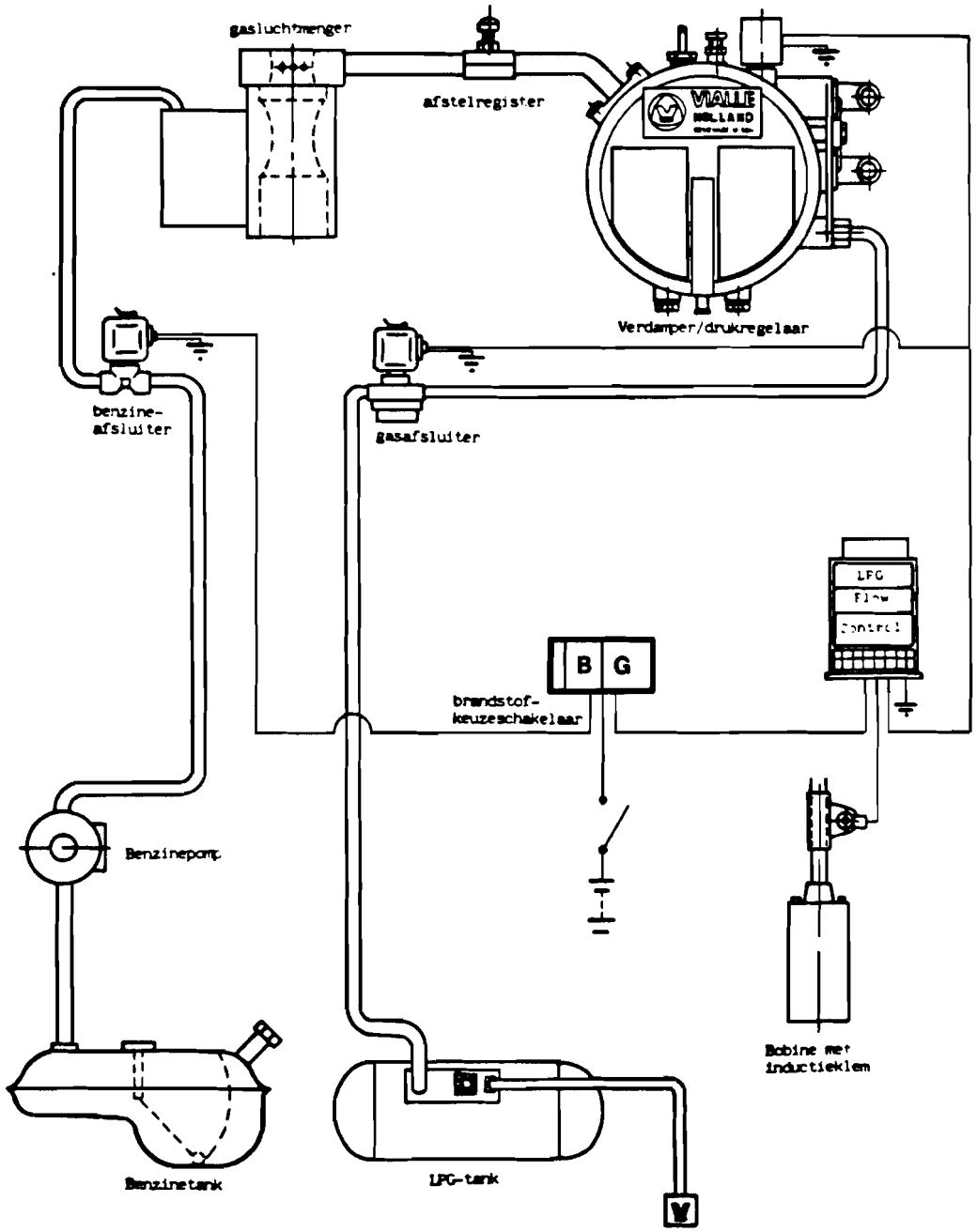


Fig. 1-1: Schema van de huidige LPG-installatie

Er zijn tamelijk veel ingangsgrootheden die invloed hebben op een optimale lucht-brandstofverhouding. Denk hierbij aan het toerental, de inlaatdruk en de temperatuur op verschillende plaatsen. Deze grootheden zijn allemaal, min of meer eenvoudig, te meten. Er zijn echter ook een aantal factoren die wel invloed op de verbranding hebben, maar niet

(eenvoudig) meetbaar zijn (denk aan klepspeling, brandstofsamenstelling e.d.).
Om deze factoren toch van invloed te laten zijn op de regeling kan men gebruik maken van terugkoppeling van het verbrandingsresultaat.

Het doel van het project is dus het ontwikkelen van een systeem dat de lucht-brandstofverhouding regelt op grond van:

- een aantal gemeten ingangsgrootheden;
- een aantal ingebrachte karakteristieken die van het type motor afhankelijk zijn;
- een terugkoppeling van het verbrandingsresultaat.

Om een systeem, zoals dat hierboven wordt beschreven, te kunnen realiseren is het bijna onvermijdelijk dat gebruik gemaakt wordt van een electronisch systeem met een zekere intelligentie.

2. ONTWERPBESLISSINGEN

2.1. KEUZE REGELPRINCIPE

Voor het te volgen regelprincipe kan uitgegaan worden van twee typen regelingen. Men moet derhalve allereerst een keuze maken uit die methoden. De beide mogelijkheden zijn:

- Trimregeling: hierbij wordt een grof geregelde basishoeveelheid gas m.b.v. een actuator bijgeregeld.
  Met deze methode kan men de LPG-installatie zoals die nu bestaat laten zoals hij is. De GLM heeft dan nog steeds de functie van hoofdregelaar. Een actuator in de vorm van een variabele restrictie tussen de VDR en de GLM kan dan de lucht-brandstofverhouding bijregelen.
- Hoofdregeling: hierbij wordt de hoeveelheid gas die aan de lucht wordt toegevoegd volledig bepaald door een actuator.
  Bij het toepassen van deze methode moet men vrijwel het gehele LPG-systeem herzien. Er zal een methode moeten worden ontwikkeld om een homogeen mengsel te maken met behulp van een actuator die door het electronische deel van het systeem kan worden bestuurd.

Als startfase voor het systeem wordt gekozen voor de eerste methode. Daarbij moet er echter wel rekening mee worden gehouden dat het eigenlijk de tweede methode is die leidt naar een systeem dat kan een grote flexibiliteit heeft. Er zijn een aantal redenen aan te geven waarom toch de eerste methode wordt gevolgd.
Ten eerste de beperking van de hoeveelheid onbekende factoren in het ontwerp. Men kan het trimmende systeem gebruiken als een opstap naar een hoofdregelsysteem. Met behulp van het trimsysteem kan men dan gevoel krijgen voor de invloeden die verschillende grootheden op het verbrandingsresultaat hebben.
Een tweede argument voor de trimregeling is dat een dergelijk systeem niet zo afhankelijk van de electronica is. Een eventueel uitvallen van

het electronische gedeelte heeft niet onmiddelijk tot gevolg dat niet meer op LPG kan worden gereden.
De verandering aan het huidige systeem blijft dan beperkt tot het vervangen van de al bestaande restrictie tussen VDR en GLM, door een variabele restrictie die kan worden gestuurd vanuit het electronische regelsysteem.

2.2. KEUZE VAN INGANGSGROOTHEDEN

2.2.1. Analoge ingangsgrootheden

In deze paragraaf worden de belangrijkste parameters beschreven die gebruikt worden ter bepaling van de gewenste lucht-brandstofverhouding. Het motortoerental (RPS) en de inlaatdruk (Pin) worden gebruikt om een indicatie te hebben voor de belastingstoestand van de motor.
De absolute temperaturen van de brandstof (Tfuel) en de lucht (Tair) worden gemeten juist voordat gas en lucht worden gemengd. Ze zijn namelijk een maat voor de dichtheid van de beide gassen. De verhouding van deze twee temperaturen is dan ook van invloed op de mengverhouding. Tenslotte wordt de temperatuur van de motor (Teng) gebruikt omdat het mengsel bij lagere motortemperaturen over het algemeen rijker moet zijn dan bij bedrijfstemperatuur.

Er zijn meer factoren van invloed op de lucht-brandstofverhouding, maar deze zijn niet (eenvoudig) te quantificeren. Denk hierbij aan brandstofsamenstelling (propaan/butaan), klepspeling e.d. Met deze grootheden wordt indirect toch rekening gehouden door terugkoppeling m.b.v. de gemeten zuurstofconcentratie in de uitlaat.

Die zuurstofconcentratie geeft aan hoe de verbranding van het gas-luchtmengsel is verlopen. Omdat die concentratie m.b.v. een sensor meetbaar is, kan deze grootheid dienen om het effect van alle factoren die niet direct meetbaar zijn, toch van invloed te laten zijn op de mengselbereiding.

Verder is het systeem te beinvloeden via een tweetal met de hand in te stellen spanningen.

2.2.2. Digitale ingangsgrootheden

Een aantal van de digitale inputs wordt uitsluitend gebruikt om vast te kunnen stellen in welke toestand de motor zich bevindt. Men kan ermee bepalen of er spanning op het systeem staat (KL15), of er gestart wordt (KL50) en of de geselecteerde brandstof LPG danwel benzine is (LPG/B). Ook is er een signaal dat een referentiepositie van de variabele restrictie aangeeft (POSref).

Een tweetal andere digitale signalen wordt in het regelalgoritme gebruikt. Deze signalen geven namelijk aan of de smoorklep (rechtstreeks verbonden met het gaspedaal) helemaal open (THop), of helemaal dicht (THcl) is. Van het eerste signaal wordt gebruik gemaakt om een extra verrijking van het mengsel bij vollast mogelijk te maken. Het tweede wordt nu (nog) niet gebruikt, maar kan, zo nodig, in de toekomst gebruikt worden om deceleratie te detecteren.

2.3. KEUZE VAN COMPONENTEN

2.3.1. Processor voor regeling

Voor de processor die in het regelsysteem wordt gebruikt is gekozen voor de Intel 8031. Deze processor is voor de taak die moet worden uitgevoerd goed geschikt. In de processor zijn timers, counters en werkgeheugen aanwezig en I/O is gemakkelijk te verwezenlijken. De keuze is ook op de 8031 gevallen omdat bij het CAB (PHILIPS) al ervaring werd opgedaan met deze processor in soortgelijke systemen. Bovendien was daardoor een ontwikkelsysteem voor de 8031 aanwezig.

2.3.2. Processorsysteem voor Commands

Het 'Commands'-gedeelte van het systeem is het deel dat nodig is om de tabellen en variabelen per motortype te kunnen ontwikkelen. Hiervoor is gekozen voor een P2000C van PHILIPS. Dit is een compleet computersysteem op basis van een Z80 processor. Het systeem bevat een beeldscherm, twee floppy-drives en een toetstenbord. Bovendien is het systeem draagbaar, zodat het in de toekomst wellicht in een rijdende proefauto kan worden gebruikt.

Met dit systeem is het mogelijk de software voor de 'Commands' in een hogere programmeertaal (UCSD-PASCAL) te schrijven.

3. BESCHRIJVING VAN HET SYSTEEM

Het totale ontwerp omvat twee microprocessorsystemen:

- Het motorregel gedeelte
- Het 'commands' gedeelte

- Het motorregelgedeelte is het deel dat in de auto is ingebouwd. Een actuator in de vorm van een variabele restrictie in de gasstroom wordt bestuurd door een microprocessor. De restrictie kan in een positie worden gezet waarmee een bepaalde doorstroomopening in de gasleiding tussen de VDR en de GLM wordt gerealiseerd. De processor berekent aan de hand van de ingangsgrootheden in welke belastingstoestand de motor zich bevindt. Met behulp van de in het geheugen aanwezige tabellen wordt bepaald welke zuurstofconcentratie en welke restrictiepositie daarbij wordt gewenst. De gewenste positie wordt ingesteld.
Ook de actuele zuurstofconcentratie wordt gemeten en met de gewenste vergeleken. Een eventuele afwijking wordt bijgeregeld door middel van een aanpassingsterm op de berekende positie van de actuator.

- Het 'commands' gedeelte is niet een werkelijk onderdeel van de motorregeling. Het is een hulpmiddel bij het samenstellen en ontwikkelen van de tabellen van de regeling. Het zorgt ervoor dat de tabellen die in het geheugen staan op een 'leesbare' manier op het beeldscherm kunnen verschijnen.

Beide bovengenoemde delen zijn te koppelen m.b.v. het gemeenschappelijk RAM waarin tijdens het ontwikkelen de tabellen worden opgeslagen.

3.1. MOTORREGELING

De belangrijkste componenten van de motorregeling zijn:

- ingangsgrootheden
- interfaces tussen sensoren en het processorsysteem
- processor hardware
- interfaces tussen de processor en de actuators
- actuators

In de volgende paragrafen worden de genoemde componenten nader bekeken.

3.1.1. Ingangsgrootheden

Om tot een juist regelalgorithme te kunnen komen is het nodig over een aantal ingangsgrootheden te kunnen beschikken. Deze worden gemeten met een aantal sensoren. Er is gekozen voor de hierna opgesomde grootheden. De functie ervan is beschreven in par 2.2.1. De namen van de veschillende signalen zijn vermeld. Deze zullen in het vervolg worden gebruikt om de signalen aan te geven.

De volgende analoge grootheden worden gebruikt:

- motortemperatuur (Teng)
- temperatuur van de aangezogen lucht (Tair)
- temperatuur van het gas (Tfuel)
- druk in het inlaatspruitstuk (Pin)
- zuurstofconcentratie in de uitlaat (O2in)
- handinstelling voor totale regelgebied (Rend)
- handinstelling voor stationair gebied (Ridle)

Daarnaast worden ook nog de volgende digitale grootheden gebruikt:

- motortoerental (RPS)
- contact (KL15)
- start (KL50)
- brandstof (LPG/B)
- smoorklep helemaal dicht (THcl)
- smoorklep ver open (THop)
- referentiepunt voor de actuator (POSref)

3.1.2. Sensoren en interfacing

Alle analoge signalen, die van de verschillende sensoren afkomstig zijn, moeten worden omgezet naar een spanning tussen 0 en 10 Volt. Deze spanning kan dan m.b.v. een AD-converter worden omgezet in een digitale waarde.
In deze paragraaf wil ik aan de verschillende interfaces een korte beschrijving wijden.

- De temperaturen worden opgenomen met sensoren waarvan de weerstand als functie van de temperatuur verandert. Over deze weerstand wordt een constante spanning gezet. De stroom die nu door de weerstand gaat lopen wordt ook door een bekende weerstand geleid, waardoor een van de temperatuur afhankelijke spanning ontstaat.

- De gekozen drukopnemer is een piezokristal. Het is zo uitgevoerd dat het vervangingschema een weerstandsbrug is. Bij absoluut vacuum (een druk van 0 Pa) is de 'brug' in evenwicht. De geleverde verschilspanning ligt in de orde van milliVolts. De interface bestaat uit een twee-traps versterker. De eerste trap zorgt voor het verwijderen van de common-mode spanning (ca. 5 V) en een kleine versterking van het signaal. De tweede trap geeft een grotere versterking.

- De O2-sensor is een sensor op basis van zirkonium-oxide die een spanning levert als functie van het in directe omgeving heersende zuurstofpercentage. Ook deze spanning ligt in de orde van milliVolts. De versterking vindt hier echter in een trap plaats.

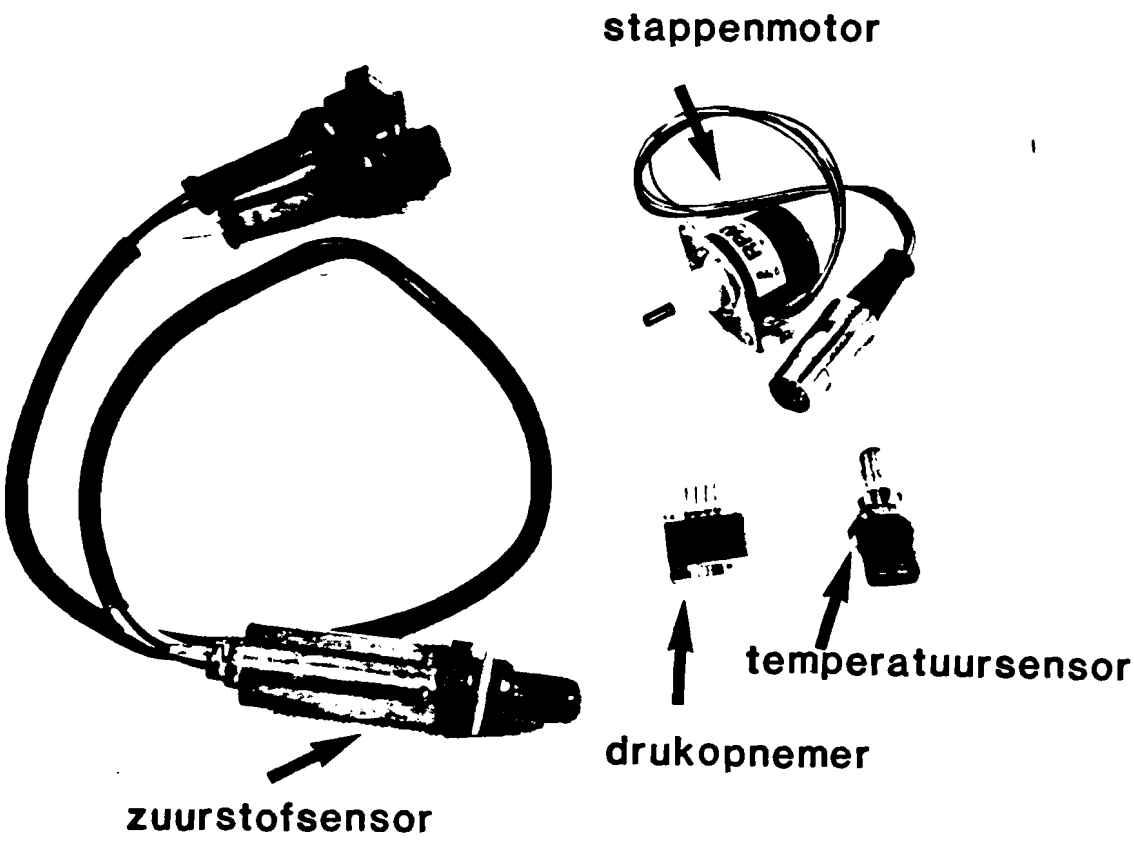


Fig. 3-1: sensoren en stappenmotor

- Wanneer we over handinstellingen spreken, spreken we eigenlijk niet over sensoren. De instellingen bestaan uit een potentiometer met daarachter een operationele versterker ter verhoging van de ingangsimpedantie.

3.1.3. Processor hardware

De motorregeling is ontwikkeld rond een 8031 microcontroller. De regeling bevat verder nog een 10 bit AD-converter (AD 573) met een multiplexer (HEF 4051) waarmee de verschillende ingangssignalen kunnen worden geselecteerd.

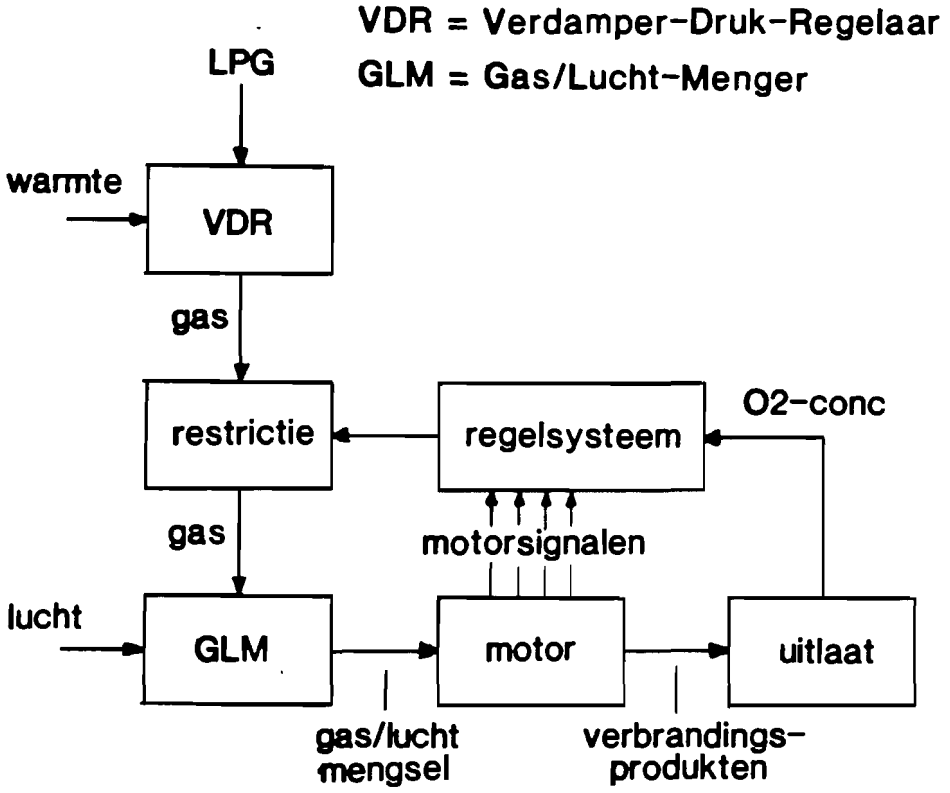


Fig. 3-2: blokschema van regelsysteem

Voor het programmageheugen en het tabellengeheugen worden EPROM's van 4k bytes gebruikt (2732A). Het werkgeheugen is opgenomen in de processor-chip.
In het databereik zijn tevens de AD-converter, de selectie van de multiplexer en de digitale inputs ondergebracht.

3.1.4. Output interfacing

De uitgangssignalen van de processor die de verschillende actuators moeten besturen moeten op het juiste niveau worden gebracht voordat ze werkelijk aan de actuators kunnen worden aangeboden.
De besturing van alle actuators gebeurt via vermogenstransistorschakelingen.

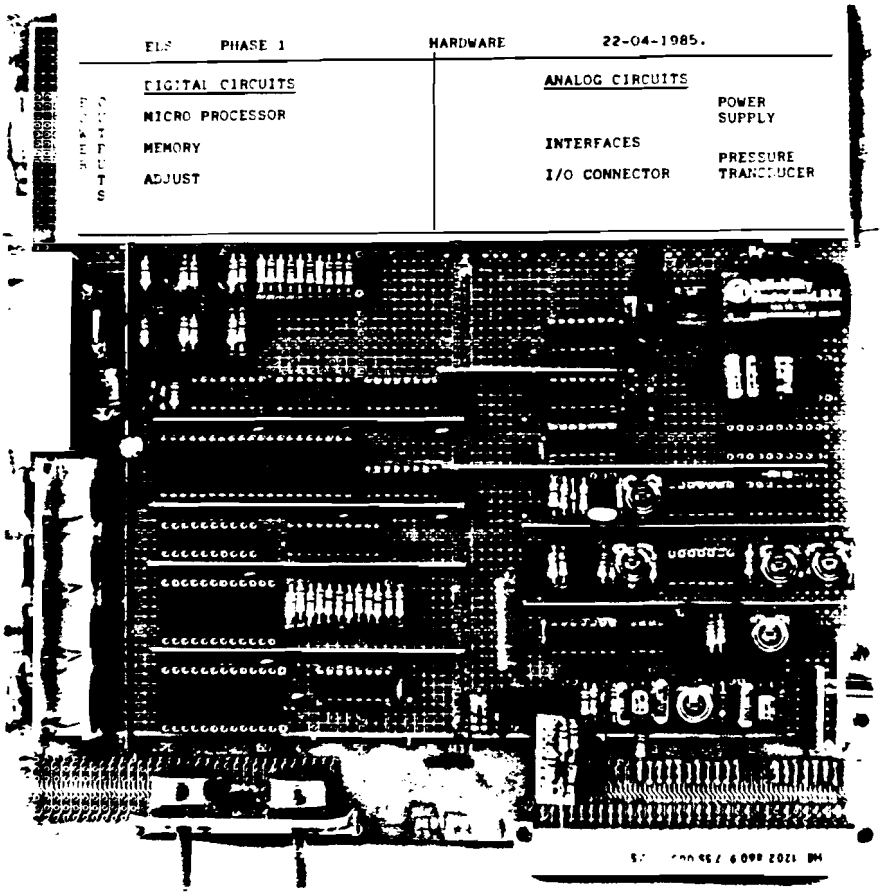


Fig. 3-3: hardware van het regelsysteem

3.1.5. Actuators

In het systeem zijn drie soorten actuators te onderscheiden: De variabele restrictie die de mengselbereiding kan beinvloeden en een aantal afsluiters die ervoor zorgen dat er alleen LPG wordt doorgelaten als

dit gewenst wordt. Ten slotte is er de verwarming van de zuurstofsensor die ook vanuit de processor kan worden bestuurd.
Nu volgt een korte beschrijving van de actuators:

- variabele restrictie:

In de leiding tussen de VDR en de GLM is m.b.v. een een lineaire stappenmotor van AIRPAX een regelbare restrictie aangebracht. Met behulp hiervan kan de doorstroomopening in kleine stapjes geregeld worden. In figuur 3-4 is een afbeelding van de restrictie te zien.

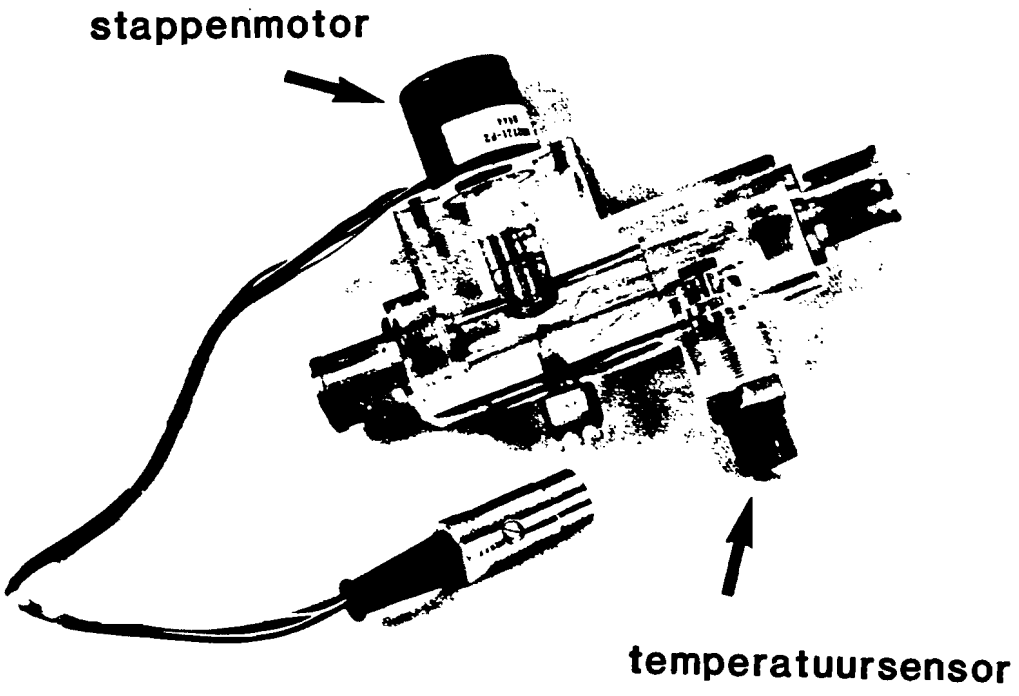


Fig. 3-4: variabele restrictie samen met temperatuuropnemer

- afsluiters in het LPG circuit:

a. VDR-afsluiter (LPG-afsluiter op VDR voor stationairstroom)
b. gasafsluiter (LPG-afsluiter tussen LPG-tank en VDR, bij VDR voor totale gasstroom)
c. tankafsluiter (LPG-veiligheidsafsluiter op LPG-tank)

Afsluiters a en b zijn in de huidige systemen al in gebruik. De derde (c) is een in dit systeem ingevoerde extra veiligheid. Een vierde afsluiter bevindt zich in het benzineciruit. Deze wordt (nog) niet door het electronische syteem bediend. Dit maakt het mogelijk bij eventuele storing in het electronische systeem toch op benzine te rijden.

- verwarming van de zuurstofsensor:

Omdat de gebruikte zuurstofsensor warm moet zijn om te kunnen functioneren, wordt deze op de gewenste tijd in en uitgeschakeld.

3.2. COMMAND'S

Omdat het invullen van de tabellen in de toekomst een taak zal zijn die veelvuldig wordt uitgevoerd, is het van belang de tabellen en variabelen gemakkelijk te kunnen beoordelen en modificeren. Hiertoe is aan het systeem reeds in 2.3.2 genoemde 'Commands'-gedeelte toegevoegd. De hardware van dit deel bestaat uit: een gemeenschappelijk RAM waarin de tabellen voor beide systemen bereikbaar zijn, een P2000C die als interface naar de operator dienst doet en een PROM-programmer waarmee een PROM kan worden gemaakt die de ontwikkelde data bevat. De genoemde delen worden in de volgende paragrafen beschreven. In figuur 3-5 wordt een blokschema gegeven.

3.2.1. Gemeenschappelijk RAM

Bij het systeem hoort een kaart met daarop 4k*8 RAM dat zowel vanuit de P2000C als vanuit de processor van de motorregeling (8031) kan worden geadresseerd. Het kan worden gebruikt op de plaats van het data-PROM in het regelsysteem. Het RAM bevat dan de variabelen en tabellen. Bovendien bevat het nog een zogenaamde 'Post-box' die dient als tussenstation voor motorgegevens die op het beeldscherm van de P2000C kunnen verschijnen.

De 8031 leest in het geheugen om de variabelen en tabellenwaarden te kunnen gebruiken in het regelalgorithme. Tijdens het samenstellen van die gegevens kunnen deze dan zichtbaar worden gemaakt op het beeldscherm van de P2000C. Het is ook mogelijk de data vanuit de P2000C te veranderen.

Een tweede mogelijkheid die deze configuratie biedt wordt ook dankbaar benut. Omdat het datageheugen van de regeling nu RAM is kunnen daar de belangrijkste grootheden van het regelsysteem op een vaste plaats in worden geschreven. Op deze manier worden deze gegevens dan elke motorcyclus aan de P2000C aangeboden. Deze kunnen dan desgewenst door de P2000C worden gelezen en op het beeldscherm zichtbaar worden gemaakt.

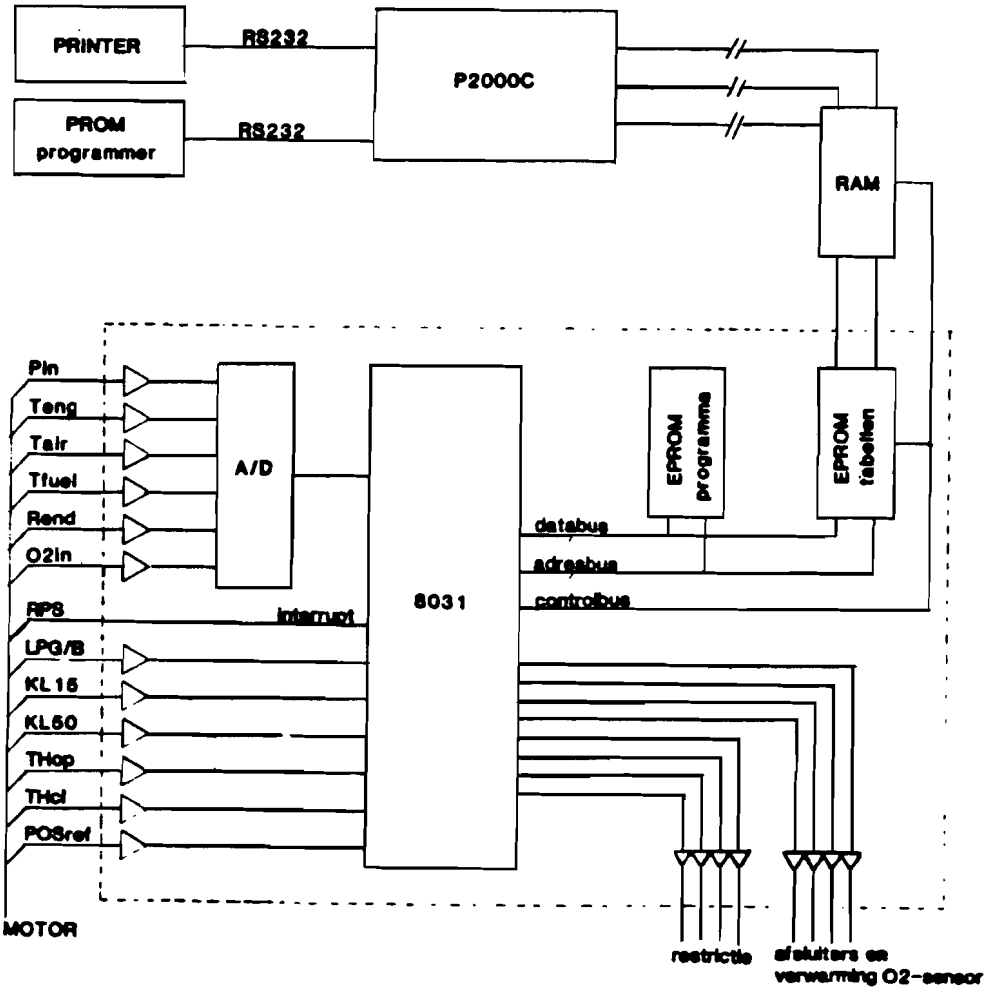


Fig. 3-5: blokschema van 'Commands'-hardware

3.2.2. PROM-programmer

De PROM-programmer is een DATA I/O 21A. Deze is op de P2000C aan te sluiten via een standaard RS-232C interface. Het EPROM dient voor opslag van (tussen)resultaten van de tabellenontwikkeling. Het kan geprogrammeerd worden met gegevens uit het gemeenschappelijk RAM of met gegevens vanuit de diskdrives van de P2000C. Andersom is het ook mogelijk om het RAM vanuit een EPROM (dat in de programmer is geplaatst) met gegevens te vullen.

4. HARDWARE VAN HET REGELSYSTEEM

In dit hoofdstuk wordt de hardware van het regelsysteem beschreven. Het systeem is te verdelen in een digitaal en een analoog gedeelte. De grens tussen de twee delen is o.a. de AD-converter.
In het analoge deel worden de verschillende ingangssignalen versterkt en gefilterd. Ook de vermogensversterkers voor de actuators worden tot het analoge gedeelte gerekend. In Appendix A en B worden de schema's gegeven.

4.1. DIGITALE DEDEELTE

Processor:
Het centrum van het digitale deel is de processor. Het is een 8031 uit de Intel 8051-familie. De processor kent twee adresbereiken: het programmageheugen en het datageheugen (beide maximaal 64 kByte groot). De digitale outputs worden door een van de poorten van de processor bestuurd. Bovendien bevat de processor de benodigde timers en counters. De processor wordt bedreven met een klok van 12 MHz. De timers en counters werken met een daarvan afgeleide klok van 1 MHz.

Geheugen:
In het regelsysteem is voor het programmageheugen niet meer dan 4 kByte nodig. Hiervoor wordt een EPROM (2732A) gebruikt. Het bereik van het programmageheugen loopt dus van adres 0 tot FFFh. De rest van het adresbereik van dit geheugen wordt niet gebruikt.
Ook de grootte van het datageheugen beperkt zich tot 4 kByte (2732A). Dit geheugen bevat de tabellen en variabelen en loopt van adres 0 tot FFFh. Behalve het datageheugen zijn ook de adressen van de I/O-devices in het databereik ondergebracht.
Het werkgeheugen van 128 bytes bevindt zich op de processor-chip.

AD-converter:
Op de adressen 1000h en 1001h van het datageheugen is de digitale uit-

gang van de AD-converter ondergebracht (1000h=high byte en 1001h=low byte).
De AD-converter is een 10-bit converter van het type AD 573.
De convert-puls, die nodig is om de conversie te starten, wordt afgeleid van de write-strobe van een schrijfinstructie naar adres 1000h. Wat er dan geschreven wordt is dus niet belangrijk.
Maximaal 20 microseconden na de conversiepuls is dan de conversie klaar en kan de geconverteerde waarde worden gelezen op de addressen 1000h en 1001h.

Digitale inputs:
Op adres 2000h van het datageheugen zijn de digitale inputs ondergebracht. De inkomende signalen van de motor hebben de spanning van de accu als hoog niveau (tussen ca.12 en 15 Volt), en het aardpotentiaal als laag niveau. Via een weerstandnetwerk worden deze inputs aangeboden aan een latch (HEF40173). De andere kant van de latch is op de databus van het systeem aangesloten.

Een uitzondering onder de digitale inputs is het RPS signaal. Dit wordt gemeten aan de primaire zijde van de bobine. Het signaal wordt in de vorm van een interrupt aan de processor aangeboden. M.b.v. een 16-bits counter wordt de intervaltijd tussen twee opeenvolgende interrupts opgenomen. Om tot een juiste waarde voor RPS te komen wordt over de motorcycli een voortschrijdend gemiddelde van die intervaltijd genomen. Fouten, die als gevolg van de verschuiving van het ontstekingstijdstip t.o.v. de motorcyclus kunnen optreden worden hiermee verkleind.

Selectie van analoge signalen:
Door naar adres 3000h een waarde te schrijven wordt een van de acht mogelijke analoge ingangssignalen geselecteerd. Dit selecteren wordt met behulp van een analoge multiplexer (HEF4051) uitgevoerd. Het nummer van de geselecteerde input wordt in een latch (HEF40174) vastgehouden.
Tenslotte is er dan nog een level-converter nodig om het logisch niveau van 10 Volt, dat nodig is op de selectie-ingang van de multiplexer te

maken uit het logische niveau van de latch (TTL niveau).

4.2. ANALOGE GEDEELTE

De interfaces die geplaatst zijn tussen de sensoren en de AD-converter hebben tot taak de signalen die de sensoren aanbieden, om te vormen tot een spanning die zo goed mogelijk is verdeeld over een spanningsgebied van 0 tot 10 Volt. Deze spanning wordt dan belast met de ingang van de AD-converter. Deze heeft een impedantie van ongeveer 5 kOhm. In Appendix A zijn de verschillende schemata gegeven.

4.2.1. Drukopnemer interface

Voeding: De drukopnemer is uitgevoerd als een brug waarop een voedingsspaning van 10 Volt moet worden aangebracht. Omdat de druksensor op de print aanwezig is, levert het aanbrengen van deze spanning geen problemen op.
Ingang: Het signaal dat door de sensor wordt geleverd is een verschilspanning (met een common mode spanning van ca. 5 Volt) en ligt in het gebied van 0 tot 80 mVolt.
Uitgang: De uitgangsspanning ligt tussen de 0 en 10 Volt en wordt belast met een impedantie van ca. 5 kOhm.
De opnemer is op de print aangebracht en is via een slang met het inlaatspruitstuk verbonden.
De druksensor zelf is al voorzien van een temperatuurcorrectie.

4.2.2. Temperatuuropnemers interface

Voeding: De temperatuuropnemers hebben het karakter van een weerstand die van de temperatuur afhankelijk is (223-423 Kelvin = 1-4 kOhm). Deze weerstand is in een schakeling opgenomen.
Ingang: Als ingang geldt de weerstand van de opnemer. Deze wordt aangestuurd met een constante spanning van 1 Volt. De stroom die gaat lopen wordt door een bekende weerstand geleid en levert daarmee een spanning

die een maat is voor de temperatuur.
Uitgang: De uitgangsspanning ligt tussen 0 en 10 Volt en wordt belast met een impedantie van ca. 5 kOhm.
De kabel naar de sensor moet goed afgeschermd worden om zo weinig mogelijk storing op te vangen.

4.2.3. Handinstellingen interface

De handinstellingen zijn beide uitgevoerd met een potentiometer van 100k. Ze bevinden zich op de print. Een buffer zorgt ervoor dat de uitgangimpedantie van de schakeling constant is over het gehele bereik.
Voeding: De potentiometers worden gevoed met een spanning van 10 V over de uiteinden.
Ingang: De loper van de potentiometer levert een spanning in het gebied van 0 tot 10 Volt.
Uitgang: De uitgangsspanning ligt tussen 0 en 10 Volt en wordt belast met een impedantie van ca. 5 kOhm.

4.2.4. Zuurstofsensor interface

Voeding: De sensor levert een spanning als functie van de zuurstofconcentratie. Hiervoor is het niet nodig een voedingsspanning aan te brengen. Voor de verwarming van de sensor echter moet de accuspanning worden aangelegd.
Ingang: De sensor levert een spanning in het gebied van 0-60 mV. De sensor zelf wordt belast met een impedantie die groter is dan 500 kOhm.
Uitgang: De uitgangsspanning ligt tussen 0 en 10 Volt en wordt belast met een impedantie van ca. 5 kOhm.
Omdat de afgegeven spanningen erg laag zijn is de verbindingsdraad afgeschermd om storingsinvloed te verminderen.

4.2.5. Output naar de restrictie

De output naar de variabele restrictie (gestuurd m.b.v. een stappenmotor) zal in de eerste opzet van het systeem gerealiseerd worden door vier digitale uitgangen van de processor. Deze worden door de software in de juiste volgorde geactiveerd om de gewenste positie te bereiken. Om een stap met de stappenmotor te kunnen zetten is een stroom van ca. 0,2 Amp. nodig.

4.2.6. Output naar de afsluiters

Voor de output naar de afsluiters wordt gebruik gemaakt van drie uitgangen van de processor. De afsluiters worden actief geopend door een stroom van 0.5 of 1 Ampere door de bekrachtigingsspoel te laten lopen.

5. REGELALGORITHME

In dit hoofdstuk wordt beschreven hoe het regelalgorithme is opgebouwd. De hierbij gebruikte afkortingen worden in de symbolenlijs verklaard. Meer details over het regelalgorithme staan in appendix D.
Allereerst worden de fasen die worden doorlopen tijdens het starten van de motor besproken. Daarna worden de closed-loop en open-loop cycli beschreven. De beschrijving van het algorithme wordt toegelicht m.b.v. in PASCAL geschreven programmadelen. Het werkelijke algorithme is geimplementeerd in ASM51 (een assembler voor de 8031).

Het brandstofregelsysteem werkt alleen wanneer de brandstof LPG is geselecteerd (LPG/B=1). Op benzine wordt gebruik gemaakt van het door de fabrikant in de auto ingebouwde carburatie- of injectie-systeem. Het is dan echter wel mogelijk de motorgegevens m.b.v. de P2000C uit te lezen zoals dat bij het lopen op gas gebeurt. Dit om de verschillen tussen benzine en LPG tijdens de ontwikkeling van de tabellen gemakkelijk te kunnen beoordelen.

5.1. Choke- en Startfase

Bij het op 'contact' zetten van de auto (KL15:=1) wordt er spanning op het regelsysteem gezet (power-up). Allereerst worden dan alle systeemdelen geinitialiseerd. Dit houdt in dat alle gebruikte variabelen hun initiele waarde krijgen, en de referentiepositie van de restrictie wordt bepaald.
Vervolgens wordt de restrictie in de startpositie POSstrt gebracht (POSstrt is instelbaar).
Dan wordt gedurende tCH seconden LPG doorgelaten om het brandstofsysteem en een deel van de motor van brandstof te voorzien voordat gestart wordt (choketoestand, zie ook fig. 5-1). tCH heeft een waarde die afhankelijk is van de motortemperatuur (Teng). Deze wordt bepaald uit een tabel met als ingang Teng: tCH = tCH(Teng);
Nadat tCH is verstreken wordt de VDR-afsluiter gesloten. Dan wordt nog

eens tOP seconden gewacht met het sluiten van de gas- en tankafsluiter (stand-by-toestand, tOP is instelbaar).
tCH ligt in de orde van 1,5 seconden, tOP in de 5 a 10 seconden.
Tenslotte komt het systeem in een wachttoestand waarbij alle afsluiters zijn gesloten.
Voor een overzicht van choke- en startfase wordt verwezen naar fig 5-1.

Alle hierboven beschreven toestanden kunnen onmiddelijk worden verlaten door te starten. Het inschakelen van de startmotor (KL50:=1), of het ontvangen van een ontstekingspuls van de motor, luidt dan de startfase in. Tijdens de startfase zijn alle afsluiters weer geopend.

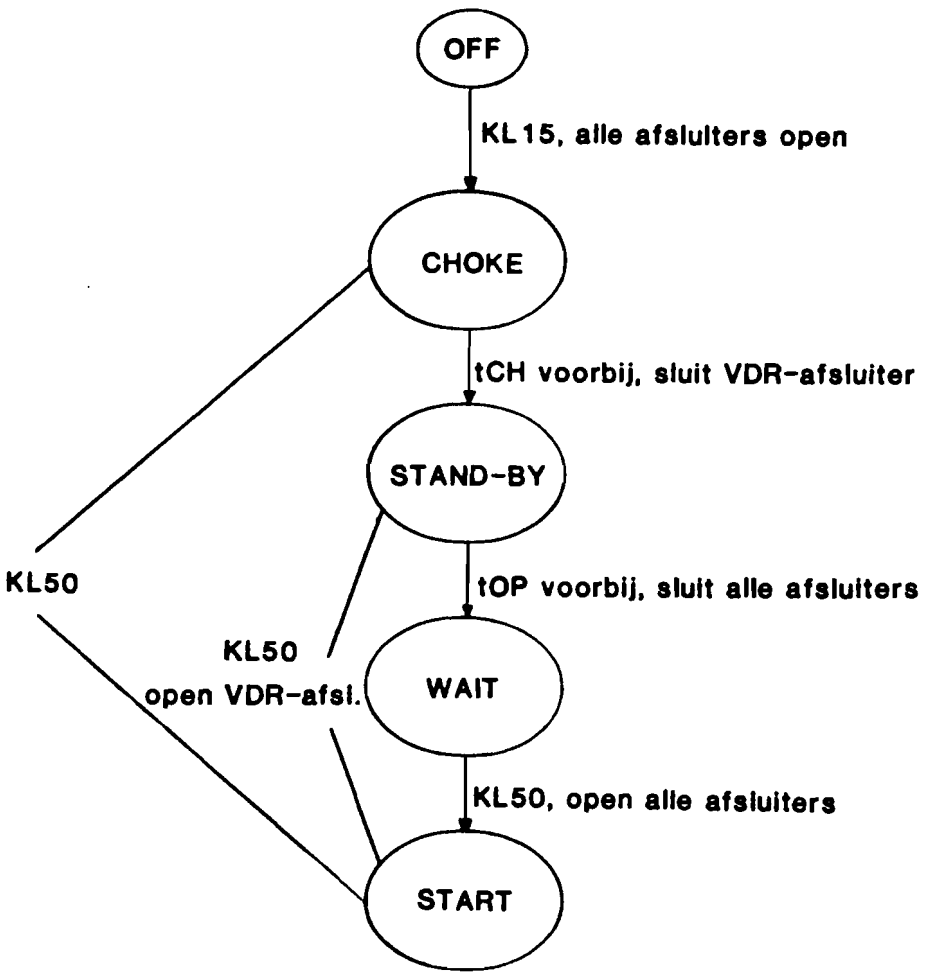


Fig. 5-1: toestanden bij choke- en startfase

De startfase wordt verlaten door de bekrachtiging van de startmotor te verbreken.
Om de accu niet extra te belasten tijdens het starten wordt de verwarming van de zuurstofsensor pas ingeschakeld op het moment dat de motor loopt. Het systeem beschouwt de motor als lopend wanneer er ontstekingspulsen komen met een intervaltijd kleiner dan 325 msec (RPS van ca. 7,5 Hz bij 4 cylinders).

5.2. Closed-loop-mode

Bij normaal bedrijf met warme zuurstofsensor, bestaat de regeling uit twee delen. Een snelle voorwaartse sturing (I) en een langzame teruggekoppelde regeling op basis van de zuurstofconcentratie (II). Alle afsluiters zijn geopend. De regeling vindt nu uitsluitend plaats d.m.v. de restrictie. De positie daarvan wordt voortdurend geregeld (de positie van de restrictie wordt aangegeven met een getal (POS) tussen 0 en 255, 0 is de referentiepositie).

I. Snelle voorwaartse sturing:

Er wordt telkens bepaald wat de positie van de restrictie moet zijn in de op dat moment heersende belastingstoestand van de motor.
Hiertoe wordt een basiswaarde voor de positie uit een tabel POSbas gehaald. Het toerental (RPS) en de inlaatdruk (Pin) zijn een maat voor de belastingstoestand (POSbas = POSbas(RPS,Pin). Uit deze basiswaarde wordt dan de restrictie-positie berekend met behulp van een aantal correctie-termen:

- Uit de verhouding tussen de absolute temperaturen van de brandstof en de aangezogen lucht wordt een correctieterm ATaf bepaald. Hiervoor bestaat een tabel ATaf(Raf) met Raf (=Tair/Tfuel). Met de waarden uit die tabel wordt gecorrigeerd voor verschillen in dichtheid tussen het gas en de lucht als gevolg van het temperatuurverschil.

- Uit de temperatuur van de motor Teng wordt de term AE bepaald dit gebeurt eveneens m.b.v. een tabel (AE(Teng)). De motortemperatuur wordt gebruikt om tijdens koudloop van de motor desgewenst het mengsel iets rijker te kunnen maken. Deze temperatuurcorrectie zal echter nauwelijks meer invloed hebben wanneer de motor op bedrijfstemperatuur is. De inhoud van de tabel AE wordt experimenteel bepaald.

- Een laatste toegevoegde term (POSadd) komt voort uit de terugkoppeling. Het is een correctie op de voorwaartssturing op grond van de verbrandingsresultaten.

Met deze termen komen we tot een gewenste restrictie-positie volgens onderstaand algorithme:

POSout = (POSbas + ATaf + AE + POSadd) * Ef;

Waarbij POSbas = POSbas(RPS,Pin) en met ATaf = ATaf(Tair/Tfuel) en AE = AE(Teng) uit tabellen te bepalen. Voor POSadd en Ef zie 5.4.

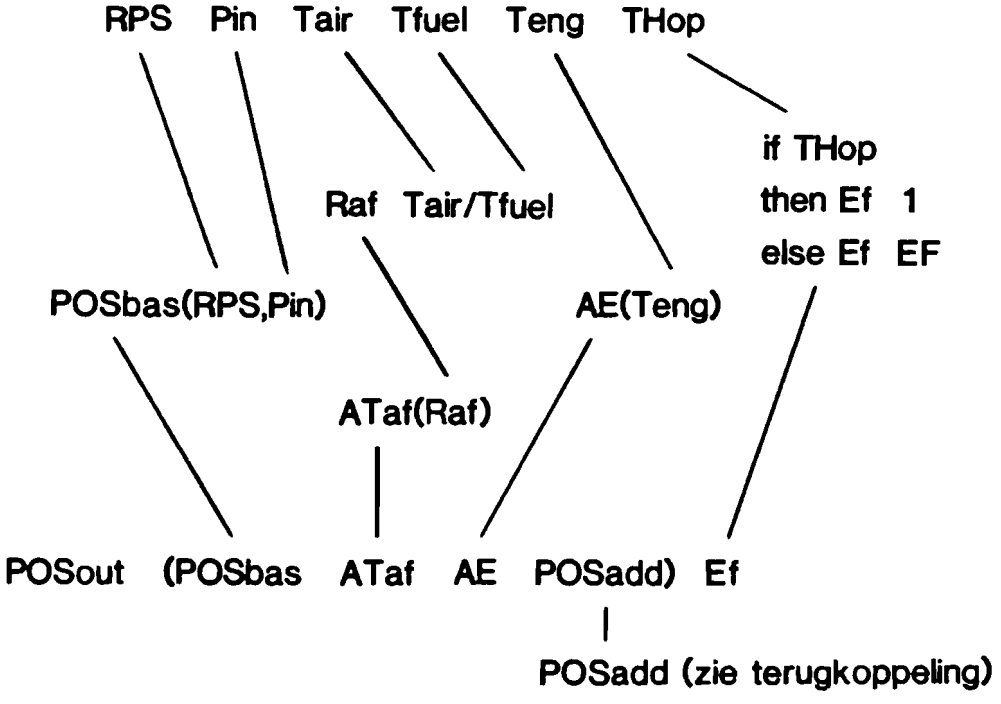


Fig. 5-2: overzicht van gebruikte tabellen bij voorwaartsregeling

II. Langzame teruggekoppelde regeling:

Elke Dt seconden (ordegrootte 2 sec.) wordt de gemeten zuurstofconcentratie (O2in) gebruikt om hiermee de terugkoppeling mogelijk te maken. Deze O2-concentratie wordt met een berekende setpoint vergeleken. Allereerst wordt nu aangegeven hoe de targetwaarde voor de zuurstofconcentratie (O2tar) wordt bepaald uit de ingangsgrootheden (zie ook fig. 5-3):

Met als ingang het motortoerental (RPS) en de druk in het inlaatspruitstuk (Pin) wordt de basiswaarde voor de gewenste zuurstofconcentratie (O2tar) uit de tabel O2bas(RPS,Pin) gehaald. Op deze basiswaarden zijn dan nog een aantal correctiefactoren van toepassing. Deze worden bepaald uit de volgende ingangsgrootheden:

- Uit een met de hand in te stellen waarde (Rend) wordt de factor FH bepaald. FH dient als eindcorrectie voor de regeling en geeft daarmee de mogelijkheid het gehele systeem een weinig (±10%) te trimmen. FH(Rend) wordt in tabelvorm gegeven. Deze tabel bepaalt het bereik en de gevoeligheid van de handinstelling.

- Uit de temperatuur van de motor Teng wordt m.b.v. een tabel de factor FE bepaald. Tabel FE bevat de gewenste verrijking van het mengsel als functie van de motortemperatuur. Deze factor heeft ook tot doel tijdens koudloop van de motor het mengsel iets rijker te maken. Deze temperatuurcorrectie zal echter nauwelijks meer invloed hebben wanneer de motor op bedrijfs temperatuur is.

De formule voor de targetwaarde van de zuurstofconcentratie wordt nu:

O2tar = FH * FE * O2bas;

Waarbij O2bas = O2bas(RPS,Pin), FH = FH(Rend) en FE = FE(Teng) op bovengenoemde wijze worden bepaald. De gebruikte tabellen kunnen voor

elk motortype opnieuw worden gekozen.

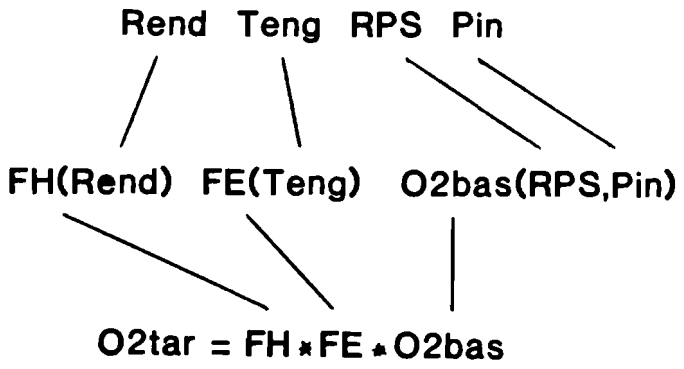


Fig. 5-3: overzicht van gebruikte tabellen bij terugkoppeling

POSadd wordt bepaald door de gewenste zuurstofconcentratie met de gemeten concentratie te vergelijken:

```
BEGIN
  IF O2in>O2tar THEN POSadd(t):=POSadd(t-Dt) + S;
  IF O2in=O2tar THEN POSadd(t):=POSadd(t-Dt);
  IF O2in<O2tar THEN POSadd(t):=POSadd(t-Dt) - S;
END
```

Waarbij S = stapgrootte en Dt = intervallengte (beide instelbaar).

Samenvattend zien wij een snelle voorwaartse regeling op basis van de tabel POSbas, die voortdurend wordt bijgeregeld met behulp van de terugkoppeling van de zuurstofconcentratie.

5.3. Open-loop-mode

Er bestaan een aantal toestanden waarin niet met behulp van het bovenstaande algorithme kan worden geregeld. Het systeem moet dan in de open-loop-mode werken. Hierna worden die toestanden beschreven. Voor elk van de situaties wordt het algorithme aangepast.

Toestanden waarin geen terugkoppeling mogelijk is:

I -Wanneer de zuurstofsensor nog niet op bedrijfstemperatuur is.

II -Bij vollast en stationair, waar het mengsel rijk moet zijn. De zuurstofconcentratie is dan altijd 0%.

III-In het gebied waar deceleratie optreedt.

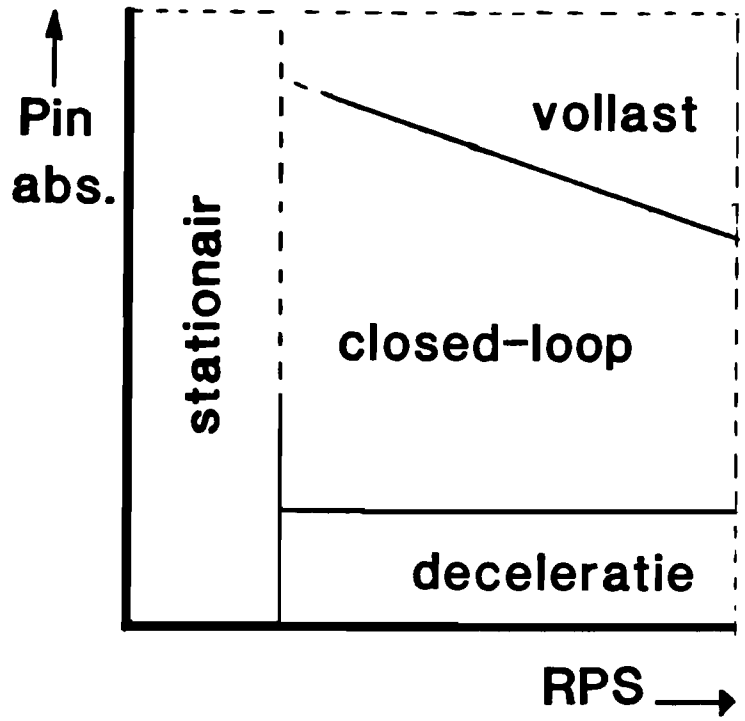


Fig. 5-4: schematisch voorbeeld van gebieden in kenveld

Ad I. Zolang de zuurstofsensor nog niet op zijn bedrijfstemperatuur is zal alleen een voorwaartssturing van de restrictie-positie worden gebruikt. POSadd krijgt initieel de waarde 0.
Na ca. 2 minuten zal de bedrijfstemperatuur van de sensor zeker bereikt zijn. Nu zal de terugkoppeling met behulp van de zuurstofsensor in werking treden en wordt het systeem gesloten. Een afwijking van de gewenste zuurstofconcentratie wordt nu langzaam weggeregeld.

Ad II. Bij het rijden met rijke mengsels kan men uit de zuurstofconcentratie niet meer bepalen hoe rijk het mengsel was. Deze situatie wordt door het systeem herkend doordat dan de berekende O2tar onder een

bepaalde drempelwaarde (O2low) komt.
De actie die dan genomen moet worden hangt af van de toestand van de motor (zware belasting of stationair):

- Bij zware belasting (d.w.z. O2tar<O2low en RPS>RPSidle) wordt de POSadd die geldt op het moment dat het gebied wordt bereikt vastgehouden. Wanneer de gewenste zuurstofconcentratie weer boven O2low komt wordt de regeling weer gesloten.

- In het stationairgebied (d.w.z. O2tar<O2low en RPS<RPSidle) wordt POSadd bepaald door een lineaire functie van een vaste AH en de laatst gevonden POSadd (POSadd(saved)). AH is de waarde die POSadd heeft bij het minimale toerental (RPSmin), en POSadd(saved is de waarde die POSadd heeft bij RPSidle. De waarde van AH is een functie van Ridle (een handinstelling). De overgang van POSadd naar AH (en terug) wordt continu gemaakt met behulp van het volgende algorithme:

$$POSadd(t) = AH + (POSadd(saved) - AH) * \frac{RPS - RPSmin}{RPSidle - RPSmin};$$

met POSadd(saved) = de waarde van POSbas bij het bereiken van het open-loop-gebied, AH = AH(Ridle), RPSidle instelbaar en RPSmin constant.

Ad III. Ook wanneer het deceleratiegebied wordt bereikt, wordt de terugkoppeling onderbroken. Deceleratie wordt herkend in de O2bas-tabel. In het gebied dat alleen kan worden bereikt wanneer gedecelereerd wordt (onder lijn van minimale belasting) staat op iedere plaats in de O2bas-tabel de waarde 255.
Bij het binnentreden van dit gebied wordt allereerst de restrictie tot een bijna gesloten positie (POSdec) gebracht (POSdec is instelbaar). Op het moment dat deze positie is bereikt, wordt ook de afsluiter op de verdamper gesloten, waardoor de toevoer van gas aan de motor afneemt en tenslotte stopt. De restrictie wordt niet helemaal gesloten om te

voorkomen dat de druk in de verdamper te ver zou kunnen oplopen.
Bij binnenkomen en verlaten van het deceleratiegebied wordt, om oscillatieverschijnselen te verminderen, een bepaalde beslissingsdrempel ingebouwd. Er wordt pas actie ondernomen wanneer het gevonden werkpunt in de tabel is omgeven door de waarde 255. Bij het verlaten van het gebied geldt hetzelfde, maar dan moet het werkpunt worden omgeven met waarden kleiner dan 255. Het terugkomen in de normale situatie gebeurt door de genomen acties in omgekeerde volgorde uit te voeren.

Het algorithme dat elke Dt seconden (Dt is instelbaar) POSadd bepaalt komt er dus in het totaal als volgt uit te zien:

```
IF (O2tar>O2low) AND (t-to>2 minuten)
THEN
  BEGIN
    IF O2in>O2tar THEN POSadd(t):=POSadd(t-Dt) + S;
    IF O2in=O2tar THEN POSadd(t):=POSadd(t-Dt);
    IF O2in<O2tar THEN POSadd(t):=POSadd(t-Dt) - S;
  END
ELSE
  IF (O2tar<=O2low) AND (RPS<RPSidle)
  THEN POSadd(t):=AH(Ridle)+(POSadd(saved)-AH(Ridle))*RPS/RPSidle
  ELSE POSadd(t):=POSadd(saved)
```

Waarbij S = (instelbare) stapgrootte en Dt = (instelbare) intervallengte, AH = AH(Ridle).
to is het tijdstip waarop wordt de motor begint te lopen. POSadd(to)=0.

5.4. Uitzonderingen

Smoorklep ver open:

Wanneer de smoorklep ver geopend is (THop=0) zal een extra hoeveelheid brandstof worden gegeven. De positie van de restrictie wordt dan met

een instelbare factor Ef vermenigvuldigd. Normaal is deze factor 1.

Toerentalbegrenzing:

Om te voorkomen dat motorschade als gevolg van een te hoog toerental kan optreden, worden de LPG- en VDR-afsluiters gesloten bij een toerental boven RPSoff. Bij een lager toerental RPSon worden de afsluiters weer geopend. RPSoff en RPSon zijn beide per motortype vast te leggen.

6. HARDWARE VOOR COMMANDS

6.1. NOODZAAK VAN EEN RAM-CIRCUIT

Om het mogelijk te maken het datageheugen van het regelsysteem te bekijken en veranderen m.b.v. een ander systeem is het noodzakelijk een hoeveelheid hardware toe te voegen. Een eis daarbij is dat er geen extra hardwarecomponenten aan de regeling worden toegevoegd. De oplossing daarvoor is een RAM-circuit te maken dat vanuit de hardware van het regelsysteem wordt geadresseerd als was het een PROM. De regeling leest dan de voor het algorithme noodzakelijke informatie uit het RAM. Om de tabellen in een EPROM te kunnen bewaren en om tijdens het lopen van de motor de tabellen te kunnen veranderen, kan ook van een andere kant in het RAM worden gelezen en geschreven m.b.v. een P2000C.

Er zijn geen extra hardwarecomponenten aan de regeling toegevoegd. Er is wel rekening mee gehouden dat de software van de motorregeling naar het RAM moet kunnen schrijven. Hiervoor is het nodig de writestrobe ook beschikbaar te hebben bij de PROM-socket. Tijdens de ontwikkeling van de tabellen is het namelijk ook mogelijk de motordata die door de sensoren van de regeling worden gemeten via het RAM op een beeldscherm te laten verschijnen.

6.1.1. De taak van het RAM-circuit

De taak van het RAM-circuit is dus het toegang verlenen aan een tweetal processoren. Daarbij beschouwt de processor in de regeling het RAM nog steeds als EPROM en zal dus ook in de proefopstelling prioriteit moeten hebben.

6.1.2. Beschrijving van de systemen

De bus van de 8031 is direct verbonden met de EPROM-socket. Aan de andere kant wordt een P2000C gebruikt om de informatie in het RAM te kun-

nen lezen en modificeren. De P2000C bevat een Z80 processor die met een klok van 4 MHz zijn werk doet. De bus van de Z80 is via een driverkaart met het RAM-circuit verbonden. Zie figuur 6-1 voor een blokschema van het totaal. Zie Appendix C voor schema's.

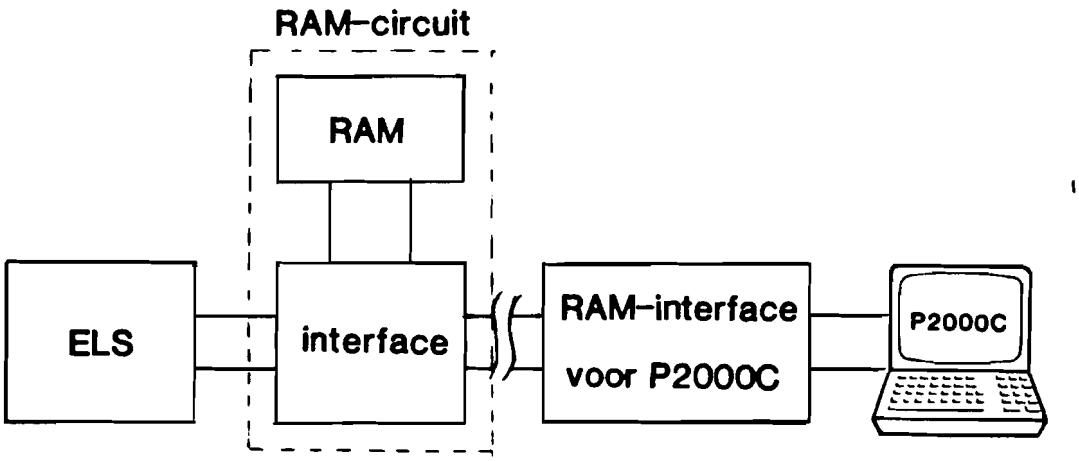


Fig. 6-1: blokschema van RAM-circuit

6.2. UITVOERING

6.2.1. Architectuur

Het RAM-circuit is grofweg in te delen in twee delen:

- Er bestaat een gebufferde busverbinding met de 8031 van de motorregeling via de PROM-socket.
- Er bestaat een soortgelijke busverbinding met de Z80 via het expansieslot van de P2000C.

Met behulp van een multiplexer kan gekozen worden welke van de beide processoren met het RAM verbonden is. De selectie wordt gemaakt door het signaal CS (Chip Select) van het 8031-systeem. Normaal, d.w.z. wanneer de 8031 geen toegang tot het RAM vraagt, heeft altijd de P2000C toegang. Op het moment echter waarop de 8031 een access wil doen op het RAM zal zijn CS de multiplexer omschakelen naar zijn kant. Hiermee

wordt bereikt dat de 8031 nooit zal hoeven wachten tot het RAM voor hem beschikbaar is.

Een probleem dat hierbij opduikt is dat de Z80 in de P2000C niet zomaar op elk moment van het RAM kan worden afgeschakeld. Het kan dan voorkomen dat de verkeerde informatie wordt gelezen of dat de informatie die geschreven moet worden verloren gaat.

6.2.2. Stilleggen van de Z80

Een oplossing voor dit probleem is het stilleggen van de activiteiten van de Z80 op het moment dat de 8031 op het RAM is geschakeld. Dat kan door gebruik te maken van de WAIT-pen van de Z80. Hiermee kan deze gestopt worden midden in een instructiecyclus. Om nauwkeuriger te zijn: op de achterflank van T2 wordt beslist of er een T3 danwel een Tw zal volgen. Telkens op het einde van Tw wordt dan deze beslissing opnieuw genomen. De beslissing hangt af van het niveau op de WAIT-pen.
Tijdens een Tw blijven alle lijnen die in T1 en T2 een bepaald logisch niveau hebben gekregen op dat niveau staan. Pas bij een T3 worden data in of uit geklokt.

Een gevaar dat bestaat is dat na de achterflank van T2 de CS van de 8031 komt. De Z80 loopt dan toch door naar T3 en zal verkeerde informatie lezen of schrijven, omdat de juiste data nog niet beschikbaar zijn. Om dit probleem te voorkomen wordt er altijd ten minste een Tw tussengevoegd. Hierdoor wordt eigenlijk de beslissing om door te gaan uitgesteld tot een later tijdstip. Dan zijn de te lezen of te schrijven data stabiel en kan de Z80 in alle gevallen doorgaan. Dus ook wanneer de 8031 er nog tussen komt (zie fig. 6-2).

6.2.3. Timing

Het toevoegen van de Tw(s) gebeurt door op het moment dat de Z80 een access op het geheugen doet WAIT actief (laag) te maken (VALID wordt

hoog en non-Qd is laag) en deze dan enige tijd actief te houden (na minimaal 375 nsec wordt non-Qd hoog). Deze tijd is voldoende lang om te zorgen dat alle data stabiel zijn voordat de flank die de data leest of schrijft komt.

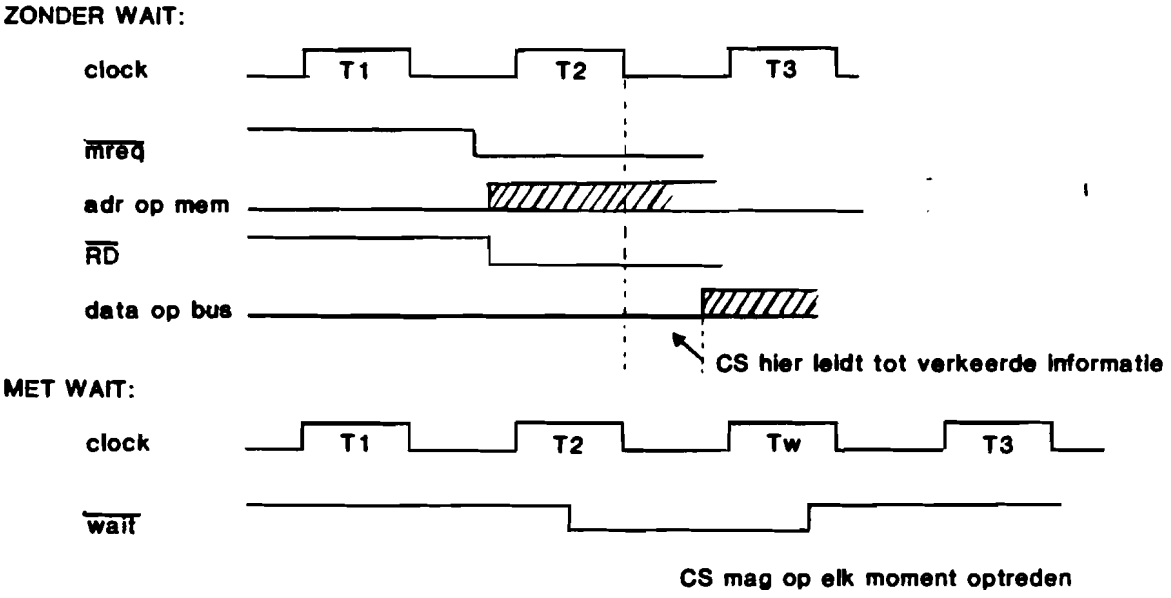


Fig. 6-2: schets van timing bij access vanuit P2000C

Wanneer tijdens een willekeurige fase van een instructiecyclus van de Z80 een CS van de 8031 komt zal de Z80 gewoon doorgaan met zijn werk totdat een access gedaan wordt op het gemeenschappelijk geheugen. Dan wordt namelijk VALID hoog en zal daardoor WAIT actief worden. WAIT wordt dan niet eerder inactief dan 375 nsec nadat CS van de 8031 inactief (hoog) is geworden.

Komt de CS op het moment dat de Z80 al op het gemeenschappelijk geheugen aan het werk wil, d.w.z. hij is T2 al voorbij en het tellen van de toegevoegde wachttijd is al begonnen, dan zal onmiddelijk nadat CS actief wordt de wachttijd opnieuw worden geinitialiseerd. Het opnieuw starten van de wachttijd zal echter pas gebeuren op het moment dat de CS van 8031 inactief wordt.
Komt de CS nadat WAIT juist weer inactief is geworden en de data nog niet zijn in- of uit-geklokt dan mag en zal de Z80 doorgaan met zijn

cylus omdat de data stabiel zijn. Men is er dan zeker van dat de juiste data zullen worden gelezen en de juiste data worden geschreven (de flank voor het schrijven wordt gemaakt doordat de multiplexer, als CS actief wordt, overschakelt naar de WR van de 8031 die op dat moment nog hoog is).

6.2.4. Het schuifregister

Om de bovengenoemde wachttijd die nodig is voor het stabiel worden van de data af te meten wordt gebruik gemaakt van een schuifregister dat wordt geklokt met een frequentie van 8 MHz (rechtstreeks beschikbaar op het expansieslot van de P2000C).
Het schuifregister is normaal altijd gevuld met enen, maar tijdens het hoog zijn van non-(VALID en non-CS) zullen er nullen naar binnen geschoven worden. Na vier schuifacties (minimaal 375 nsec) zal het WAIT-signaal weer inactief worden en kan de lees- of schrijfactie worden afgerond met het in of uitklokken van de informatie.
Wordt tijdens het schuiven CS actief, dan zal het schuifregister worden gereset en weer met enen gevuld. Hij gaat dan pas weer schuiven nadat CS inactief geworden is.

6.2.5. De bufferkaart

Omdat de afstand tussen de P2000C en het motorregelsysteem groot is, is de kabel lang (ca 150 cm). Daar de storingen die in een auto kunnen voorkomen nogal groot zijn moet het niveau op de kabel goed vastliggen. Dit kan gebeuren met behulp van drivers. Deze drivers zijn aan de kant van de P2000C geplaatst omdat er dan bij de regelprocessor hiervoor geen extra maatregelen getroffen hoeven worden. Het RAM-circuit wordt dus zo dicht mogelijk bij de motorregeling geplaatst.
Op de driverkaart in de P2000C wordt ook een VALID signaal gemaakt. Dit signaal heeft logisch gezien altijd dezelfde waarde als VALID op het RAM-circuit, maar voor de timing is het noodzakelijk dat het signaal al op de bufferkaart gemaakt wordt (looptijden).

7. SOFTWARE VOOR COMMANDS

Het 'commands' gedeelte wordt op de P2000C gestart door aanbrengen van de juiste floppy-disk (naam: COMMANDS) in drive 1 en het indrukken van de RESET toets.
In de tweede drive kan eventueel een floppy-disk met gegevens worden geplaatst.
Op het display van de P2000C verschijnt nu het commands-menu:

COMMANDS: C(ollect, D(isplay, M(odify, P(rint, R(ead, W(rite, <cr>.

Het systeem is nu klaar om commando's uit te voeren. Een commando kan worden gegeven door het intoetsen van de eerste letter van een van de in het menu gegeven commando's.

7.1. COMMANDO'S ALGEMEEN

Wanneer het menu is verschenen kan met het intypen van de commando's worden begonnen. Een commando bestaat uit een commando-woord, een object-woord en eventueel een of meer parameters. Voor het bepalen van het commando-woord is het intoetsen van de eerste letter voldoende om het gewenste commando te selecteren.
Na het bepalen van het commando verschijnt het menu van de bij dit commando behorende object-woorden:

COLLECT: E(ngine data, Q(uit.
DISPLAY: E(ngine data, T(able, V(ariables, Q(uit.
MODIFY: E(ngine data, T(able, O(ffset-Multiply, V(ariables, Q(uit.
PRINT: E(ngine data, T(able, V(ariables, Q(uit.
READ: D(isk, P(ROM, Q(uit.
WRITE: D(isk, P(ROM, Q(uit.

Ook hieruit kan een keuze worden gemaakt door het intoetsen van de eerste letter.

Na het fout intypen van de commando's of objecten kan het gehele commando worden afgelast door het gebruiken van de in het menu voorkomende commando Quit.
Wanneer het systeem het gevraagde commando niet kent zal het niet reageren. Als RETURN wordt ingedrukt zonder een nieuw commando, zal het vorige commando nog eens worden uitgevoerd. Alleen het invullen van de parameters zal opnieuw moeten gebeuren.
Het commando 'WRITE PROM' mag alleen gegeven worden wanneer de motor NIET loopt. Wordt dit commando toch gegeven dan reageert het systeem met de melding 'RUNNING ENGINE'.
Als het commando 'COLLECT ENGINE DATA' wordt ingetoetst terwijl de motor niet loopt wordt de melding 'NO RUNNING ENGINE' op het display gezet.

7.2. DE COMMANDO'S

De volgende commando's zijn te gebruiken:

D(isplay) E(ngine data) 'FROM' <xxx> 'TILL' <yyy>
D(isplay) T(able) <table number>
D(isplay) V(ariables)

M(odify) T(able) <table number>
M(odify) O(ffset-Multiply)
M(odify) V(ariables)

P(rint) E(ngine data) 'FROM' <xxx> 'TILL' <yyy>
P(rint) T(able) <table number>
P(rint) V(ariables)

R(ead) D(isk) <filename>
R(ead) P(ROM)
W(rite) D(isk) <name/date> <filename>
W(rite) P(ROM)

C(ollect) E(ngine data) S(ingle) 'INTERVAL' <xxx>

C(ollect) E(ngine data) C(ontinuous) 'INTERVAL' <xxx>

7.2.1. READ PROM of DISK

Dit commando leest de inhoud van het EPROM in de PROM-programmer of de inhoud van de opgegeven diskfile en schrijft de gegevens naar de overeenkomende plaatsen in het gemeenschappelijk RAM.
Tijdens het lezen zal de naam en de datum van de gelezen PROM of file op het display verschijnen.
Als er geen gegevens in de PROM staan zal het systeem reageren met 'PROM EMPTY'. Wanneer de genoemde file niet bestaat wordt de melding 'FILE NOT FOUND' gegeven.

7.2.2. WRITE PROM of DISK

Dit commando dient voor het programmeren van de EPROM of het opslaan van de tabellen in een diskfile.
Men kan de EPROM laden rechtstreeks vanuit het RAM. Het systeem begint met het vragen naar een naam en een datum met de mededeling 'NAME/DATE:'. De naam en de datum worden dan in het RAM bijgeschreven, om dan later samen met de gegevens in de EPROM opgeslagen te worden.
Wanneer een fout optreedt bij het uitvoeren van dit commando zal de foutmelding 'WRITE ERROR' op het display verschijnen.
Voor het opslaan van de gegevens in een diskfile gelden dezelfde opmerkingen. Wanneer echter al een file bestaat met dezelfde naam wordt de waarschuwing 'OVERWRITE OLD FILE? (Y/N)' gegeven.

7.2.3. DISPLAY VARIABLES

Bij het uitvoeren van dit commando worden alle variabelen op het display getoond. Boven de gegevens staat weer de naam en de datum van de tabel. De namen van de verschillende variabelen worden samen met hun

waarden gegeven.

7.2.4. DISPLAY TABLE

Na intoetsen van dit commando zal een menu met tabelnamen en nummers worden aangeboden. Hieruit kan dan een keuze worden gemaakt. De gevraagde tabel zal dan op het beeldscherm verschijnen. Ook hier zal bovenaan in beeld een mededeling te vinden zijn met daarin opgenomen de datum en de naam van de PROM en het nummer en de naam van de tabel.

- De 3-D tabellen (met twee ingangsvariabelen) bestaan uit 256 (16*16) waarden. De linker onderhoek van de matrix is de oorsprong (f(0,0)). Boven de tabel staan de RPS waarden.
- De 2-D tabellen worden weergegeven op een lijn. Boven de waarden staan de waarden van de ingangsvariabelen.

In de 3-D-tabellen wordt het actuele werkpunt van de motor aangegeven met een de cursor.

7.2.5. PRINT...

Dit commando zorgt voor het afdrukken van de tabellen of variabelen op de printer. De informatie die wordt geprint is ingedeeld zoals dat bij 'DISPLAY...' het geval is.

7.2.6. MODIFY...

Het eerste deel van het MODIFY-commando is gelijk aan het DISPLAY-commando. Nadat de gevraagde tabel of de variabelen op het scherm zichtbaar gemaakt zijn is het mogelijk met behulp van de cursor de waarde die men wil veranderen aan te wijzen. Het veranderen zelf kan men op twee manieren doen. Allereerst door een nieuwe waarde in te typen en vervolgens de SPATIE-balk in te drukken. De nieuwe waarde wordt dan in de tabel opgenomen. Er kan een klein verschil bestaan tussen de nieuwe waarde en de ingetypte waarde. Dit is een gevolg van

de afronding naar de kleinste resolutie van de tabel.
Een tweede manier om de waarde in de tabel te veranderen is het indrukken van de '+' of de '-' toets. Met de '+' wordt de waarde met telkens de kleinst mogelijke verandering opgehoogd, met de '-' wordt de waarde met de kleinst mogelijke stapjes verminderd. Door het vasthouden van de toets repeteert deze verandering.
Het modificeren wordt beeindigd met het indrukken van de ESCape-toets.

7.2.7. COLLECT ENGINE DATA

Met dit commando kunnen de motorgegevens vanuit de 'post-box' naar het geheugen van de P2000C worden geschreven. Op het display verschijnt het menu:

COLLECT ENGINE DATA: S(ingle, C(ontinuous

Als een 'S' wordt ingetoetst, wordt een maal het gehele buffer volgeschreven met gegevens. Aan het einde van het buffer wordt dan gestopt met het bewaren van de motorgegevens.
Wanneer de 'C' wordt ingetoetst zal het buffer als een ringbuffer fungeren. Het verzamelen van de gegevens stopt dan pas nadat een willekeurige toets wordt ingedrukt.
Nadat een keus is gemaakt uit C en S, wordt gevraagd met welk interval de gegevens moeten worden opgeslagen ('INTERVAL: '). Dit interval moet worden gegeven in het aantal ontstekingspulsen tussen twee monsters.
Wanneer na het commando 'C(ollect)' met een RETURN wordt gereageerd, zullen de op dat moment geldige motorgegevens eenmaal op het beeldscherm verschijnen. Als het gewenst is meer waarden op het scherm te laten zien kan het commando met het indrukken van RETURN worden herhaald.

7.2.8. DISPLAY ENGINE DATA 'FROM' <xxx> 'TILL' <yyy>

Met dit commando kunnen de gegevens, zoals die met COLLECT ENGINE DATA

zijn verzameld, op het beeldscherm zichtbaar gemaakt worden. Op de vraag 'FROM' en 'TILL' kan het begin resp. het eindnummer van de gewenste gegevens worden aangegeven.
Tijdens het schrijven van de gegevens kan met behulp van een willekeurige toets de gegevensstroom worden gestopt.
Herhalen van deze actie zal de stroom weer voortzetten. Met een ESCape kan het gehele commando worden gestopt.

7.2.9. PRINT ENGINE DATA 'FROM' <xxx> 'TILL' <yyy>

Dit commando zorgt voor het afdrukken van de motorgegevens op de printer. De informatie die wordt geprint is ingedeeld zoals dat bij het commando 'DISPLAY...' het geval is. Ook een eventuele onderbreking van het printen geschiedt weer door het indrukken van een willekeurige toets.

7.3. DISPLAY OUTPUT

7.3.1. Schaalfactoren (M:varname en V:varname)

Er zijn voor elke ingangsgrootheid twee schaalgrootheden nodig om de waarde met de optimale resolutie te kunnen gebruiken. Voor elke waarde is er een offset V en een vermenigvuldigfactor M. De namen van de variabelen waarop deze waarden van toepassing zijn zijn een M: of een V: met daarachter de naam van de variabele.

7.3.2. De indeling van de tabellen

De ingangen naar de verschillende tabellen zijn de volgende:

RPS: motorfrequentie in Hz.
Pin: output van de druksensor, gecorrigeerd met V:Pin en M:Pin.
Rend: output van een spanningsdeler waarmee een correctiefactor op het gehele werkgebied kan worden ingesteld.

Ridle: output van een spanningsdeler waarmee een correctie op het stationaire gebied kan worden uitgevoerd.

Teng: output van de temperatuursensor in de motorkoelvloeistof, gecorrigeerd met V:Teng en M:Teng.

Tair: output van de temperatuursensor in het luchtfilter, gecorrigeerd met V:Tair en M:Tair.

Tfuel: output van de temperatuursensor in de gasleiding, gecorrigeerd met V:Tfuel en M:Tfuel.

Na deze opsomming volgt nu een beschrijving van de tabelingangen:

Motorsnelheid (RPS): Voor de tabellen waarbij de snelheid van de motor een ingang is worden 16 ingangswaarden onderscheiden. De omwentelingstijd wordt aangegeven in eenheden van 1024 microsec. De volgende eenheden zijn daarvoor gekozen: 65,47,32,27,23,20,17,15,13,11,9,8,7,6,5,4. Hetgeen overeenkomt met RPS'en van resp.:
7.51;10.39;15.26;18.08;21.23;24.41;28.72;32.55;37.56;44.39;54.25;61.04;69.75;81.38;97.66;122.07

Druk (Pin): De waarde zoals die via de A/D-converter van de druksensor komt wordt in een lineaire schaal verdeeld. Hierbij wordt met behulp van V:Pin de offset, en met M:Pin de schaalfactor zo bepaald dat een zo groot mogelijke nauwkeurigheid bereikt kan worden.

Correctiefactoren (Rend,Ridle): De correctiefactoren direct over de 255 waarden verdeeld zoals ze van de AD-converter komen. M.b.v tabellen (AH en FH) worden ze vertaald naar een factor tussen 0 en 2.

Temperatuur (Teng,Tair,Tfuel): Voor de temperatuur geldt hetzelfde als voor de druk. De waarde die van de A/D-converter komt wordt lineair over de schaal verdeeld m.b.v. V:T en M:T.

Hieronder volgt een opsomming van de gebruikte tabellen (3D en 2D) en variabelen (1D):

3D: O2bas, lambda en POSbas

2D: AE, AH, ATaf, FE, FH, tmpcon, O2con, Pcon

1D: Dt, stpint, Ef, O2low, POSdec, POSstrt, RPSidle, RPSlow, RPSon, RPSoff, S, Top, nrcyl en index

3D tabellen: De drie 3D-tabellen hebben als ingang het motortoerental en de druk in het inlaatspruitstuk (RPS en Pin). Het toerental staat boven de tabel in RPS vermeld boven elke kolom en loopt op van links naar rechts. De druk wordt niet bij de tabel vermeld. De laagste druk (absoluut) staat op de onderste rij van de tabel.

- O2bas(RPS,Pin) wordt gegeven in percenten*100 (0-700);
- lamda(RPS,Pin) wordt gegeven in 1/1*100 (100-150);
- POSbas(RPS,Pin) wordt gegeven in stappen vanaf de referentie (0-255).

2D tabellen: De 2D-tabellen hebben allemaal 16 ingangen. Op het beeldscherm wordt zowel de ingang als de in de tabel gegeven waarde aangegeven.

- AE(Teng) wordt gegeven in stappen (0-255);
- AH(Ridle) wordt gegeven in stappen (0-255);
- ATaf(Raf) wordt gegeven in stappen (0-255);
- FE(Teng) wordt gegeven in 1/1*100 (0-200);
- FH(Rend) wordt gegeven in 1/1*100 (0-200);
- tmpcon is de linearisatietabel voor de temperatuur (0-255);
- O2con is de linearisatietabel voor de zuurstofconcentratie (0-255);
- Pcon is de linearisatietabel voor de druk (0-255);

1D variabelen: De Variabelen worden allemaal gelijktijdig op het beeldscherm getoond. Hun naam en de inhoud worden beide gegeven.

- Dt wordt gegeven in seconden*100 (0-1670);
- stpint wordt gegeven in msec*100 (0-65);
- Ef wordt gegeven in 1/1*100 (0-200);
- O2low wordt gegeven in percenten*100 (0-700);
- POSdec wordt gegeven in stappen vanaf referentie (0-255);

- POSstrt wordt gegeven in stappen vanaf referentie (0-255);
- RPSidle wordt gegeven in Hz (1.91-488.28);
- RPSlow wordt gegeven in Hz (1.91-488.28);
- RPSon wordt gegeven in Hz (1.91-488.28);
- RPSoff wordt gegeven in Hz (1.91-488.28);
- S wordt gegeven in stappen (0-255);
- Top wordt gegeven in seconden*100 (0-1670);
- nrcyl wordt gegeven in 1/1 (0-255);
- index wordt gegeven in 1/1 (0-255);

7.3.3. Motorgegevens

Wanneer het commando 'Display Engine Data' wordt gebruikt zullen op het beeldscherm de volgende grootheden verschijnen:

RPS: De motorfrequentie in Hz;
Teng: temperatuur van de motor in Kelvin;
Tair: temperatuur van de aangezogen lucht in Kelvin;
Tfuel: temperatuur van de brandstof in Kelvin;
Pin: de druk in het inlaatspruitstuk in kPa;
O2in: Zuurstofconcentratie in de uitlaat gegeven in percenten*100;
O2tar: Gewenste zuurstofconcentratie in de uitlaat gegeven percenten*100;
POS: De huidige restrictiepositie in stappen vanaf referentie;
POSout: de uitgestuurde restrictiepositie in stappen vanaf referentie;
POSadd: de correctie op de voorwaartstabel in stappen;
FH: correctiefactor voor O2tar over het hele werkgebied in 1/1*100;
AH: correctieterm voor restrictiepositie in het stationaire gebied in stappen;
lact: Huidige waarde van lambda in 1/1*100;
ltar: Gewenste waarde van lambda in 1/1*100;
mem1: Inhoud van de met de variabele index aangegeven geheugenplaats.

8. TESTRESULTATEN

De testresultaten, zoals die in dit hoofdstuk worden beschreven zijn niet de testresultaten van het totale systeem van motor, LPG-installatie en electronisch regelsysteem. Het systeem is namelijk allen nog maar getest op zijn functies.
Hoe en in welke volgorde ik de verschillende delen van het systeem heb getest wil ik in de volgende paragrafen beschrijven.

8.1. Digitale hardware

De eerste testfase omvatte het testen van de digitale hardware. Hierbij heb ik getest of de juiste digitale informatie werd ingelezen. De digitale inputs kon ik simuleren m.b.v. schakelaars op een testkast die bij VIALLE speciaal voor testdoeleinden was gemaakt. Tevens heb ik daarmee de digitale uitgangen getest. Voor elke digitale uitgang (4 voor de stappenmotor, 3 voor de afsluiters en een voor de verwarming van de zuurstofsensor) is op de testkast een LED aangebracht.

8.2. Schakelende software

Toen bleek dat de hardware goed functioneerde kon een deel van de software van het regelsysteem worden getest. De stukken die getest konden worden omvatten de initialisatie,het openen en sluiten van afsluiters, het in- en uitschakelen van de zuurstofsensor en het goed reageren op de digitale inputs.

8.3. Berekenende software

Een erg belangrijke fase was het testen van de routines die een aantal ingewikkelde en veel voorkomende berekeningen en interpolaties uit te voeren hebben. Door deze routines nu al te testen kon ik verderop in de tests volledig op de juistheid van deze routines vertrouwen. Bovendien werd hiermee feitelijk de datastructuur en de representatie van de

grootheden in de processor getest.

8.4. Hardware van RAM-circuit

Toen was de tijd aangebroken om de werking van het regelalgorithme te gaan testen. Daarvoor is het al noodzakelijk de gegevens in het gemeenschappelijk RAM te kunnen bekijken en modificeren. Het was dan ook wenselijk op dit moment al over de hard- en software voor de 'Commands' te kunnen beschikken.
Het RAM-circuit vormde daarvan het belangrijkste hardwaregedeelte.[1] Dit stuk heb ik kunnen testen m.b.v. de emulator aan de kant van de 8031 en aan een monitorprogramma voor de P2000C-kant aan de andere kant.

8.5. Software voor commands

Het grootste gedeelte van de software voor de 'Commands' kon worden getest zonder extra hardware. Dat deel is dan ook tijdens het ontwikkelen al getest. Een deel van de software was echter wel hardware-afhankelijk (denk aan de transfer van motorgegevens van de regeling naar de P2000C). Dit deel werd in dit stadium ontwikkeld en getest.

8.6. Analoge hardware

Nu was het dus moglijk op een efficiente manier de data in het gemeenschappelijk geheugen te bekijken en modificeren. Het was nu bovendien gemakkelijk de motorgegevens zichtbaar te maken.
Hiermee was dus nu de tijd gekomen om de analoge hardware te testen. Op de ingangen werden eerst m.b.v. spanningen (vanaf de testkast) en regelbare weerstanden de sensoren gesimuleerd. Later werden de interfaces ook getest met de werkelijke opnemers.

8.7. Totale systeem

Tenslotte werd het gehele systeem getest. Hierbij werden alle verschil-

lende mogelijkheden (closed-loop, open-loop, deceleratie, volgas) kritisch bekeken. Toen bleek dat het totaal functioneerde is tenslotte de werkelijke restrictie aangebracht.
In de tabellen zijn voor deze tests slechts waarden ingevuld die geen betekenis hebben in de werkelijkheid. Er werden waarden ingevuld om een fuctionele test van het geheel te kunnen uitvoeren.

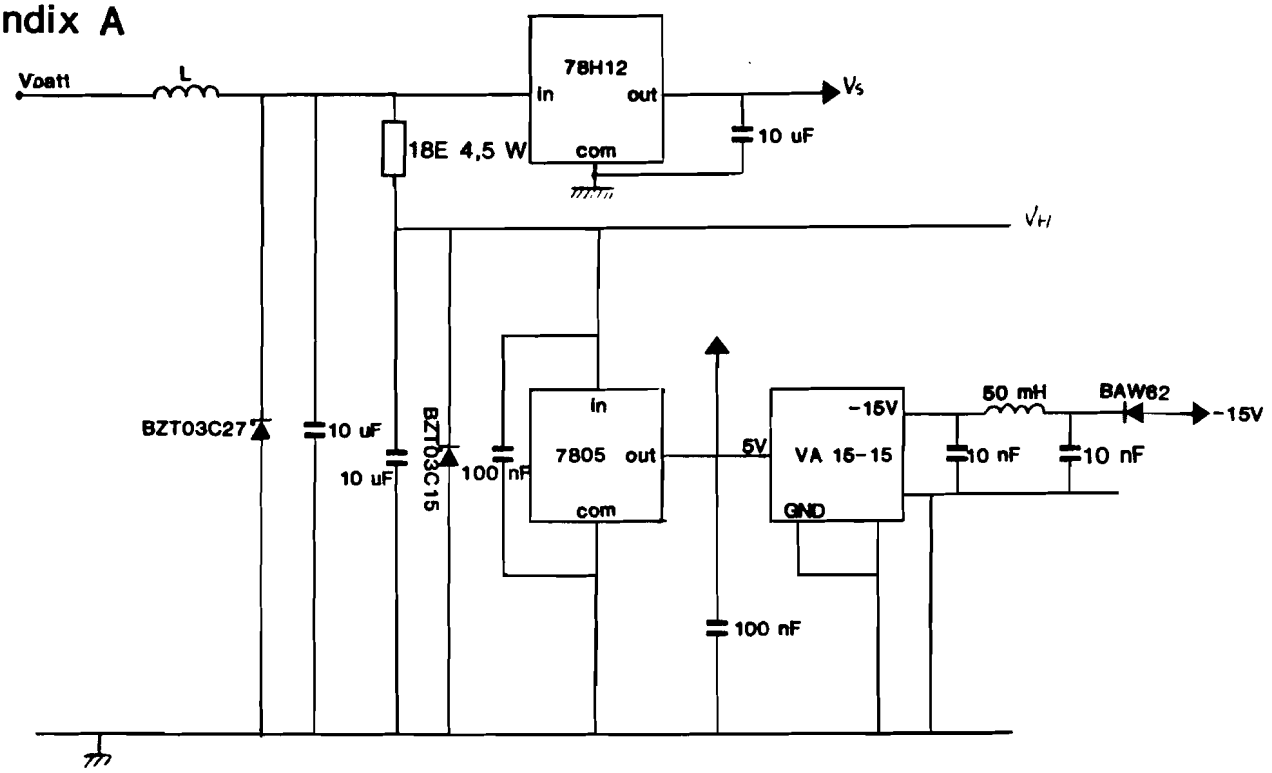
9. CONCLUSIE

Het hiervoor beschreven Electronische LPG Systeem is nu zover klaar dat de proeven op de motorproefstand kunnen beginnen. Er zijn dus op dit moment nog geen conclusies te trekken met betrekking tot resultaten uit de praktijk. Het systeem is tot nu toe getest met behulp van gesimuleerde signalen. Daarmee kon het gewenste algorithme in zekere mate worden getest. Voor het beter testen van de terugkoppeling moet men het echt proces gebruiken.

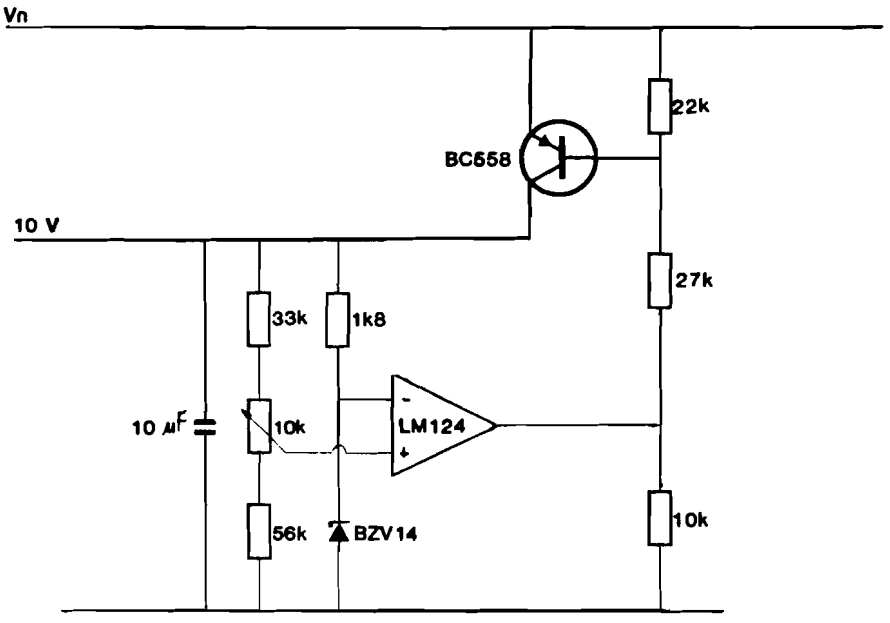
Voor de nabije toekomst zal dus het testen van het systeem op de motorproefstand de belangrijkste taak zijn. Met name het analoge deel kan daarbij beter worden bekeken.

Voor de wat verdere toekomst zou een verbetering van het Command's-gedeelte een stap vooruit kunnen zijn. Het is namelijk mogelijk het invullen van de tabellen tot in zekere mate te automatiseren. Er bestaat immers een afhankelijkheid tussen de positie van de restrictie en de gewenste zuurstofconcentratie. Met andere woorden, met het instellen van de gewenste zuurstofconcentratie in een bepaald punt is de bijhorende restrictiepositie ook vastgelegd.

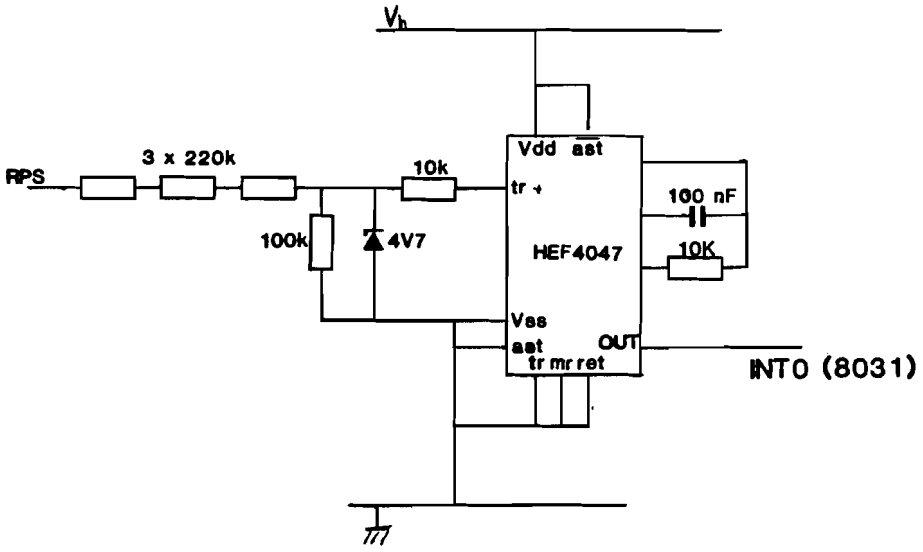
Appendix A



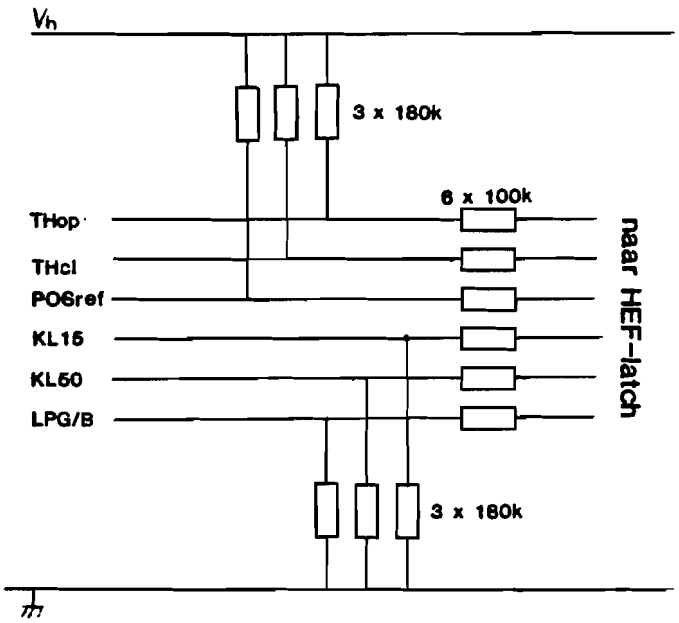
kortsluitbeveiliging, voeding: 5V, -15V



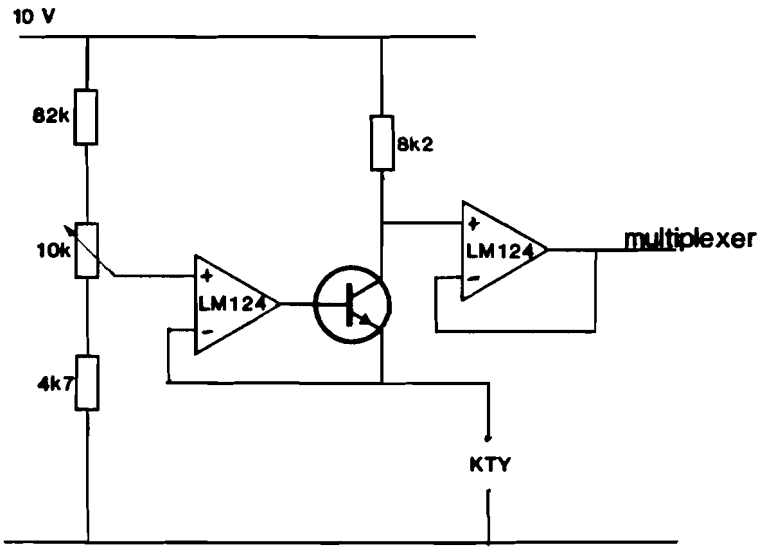
voeding 10V



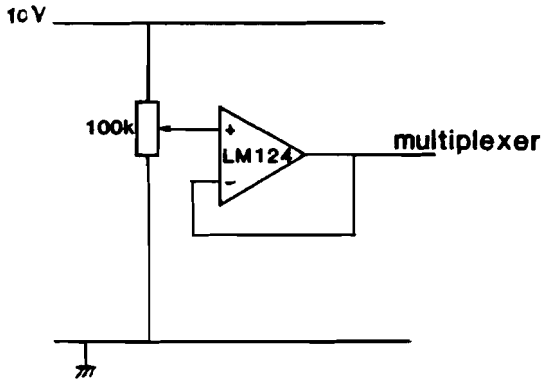
interface voor RPS



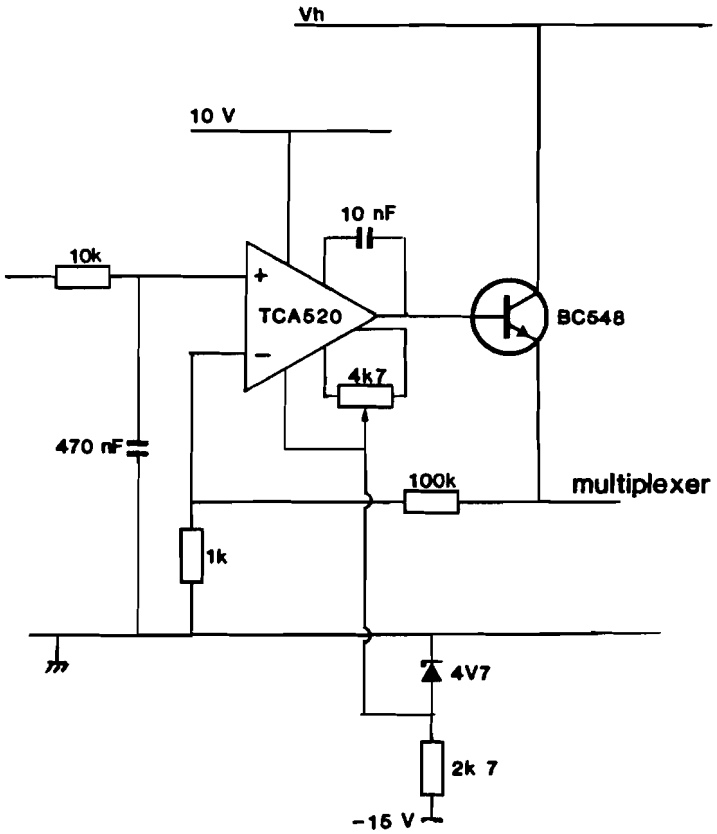
digitale inputs



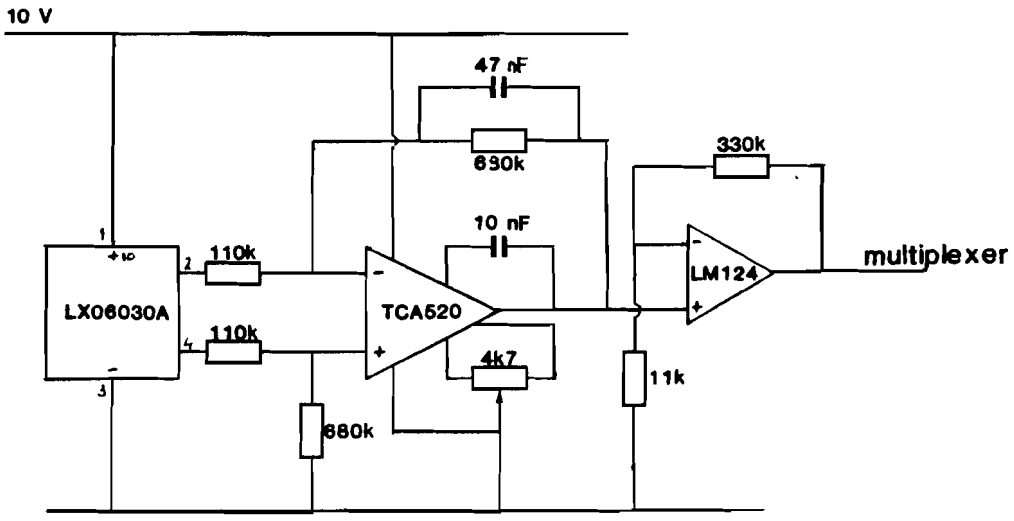
interface voor temperatuuropnemers



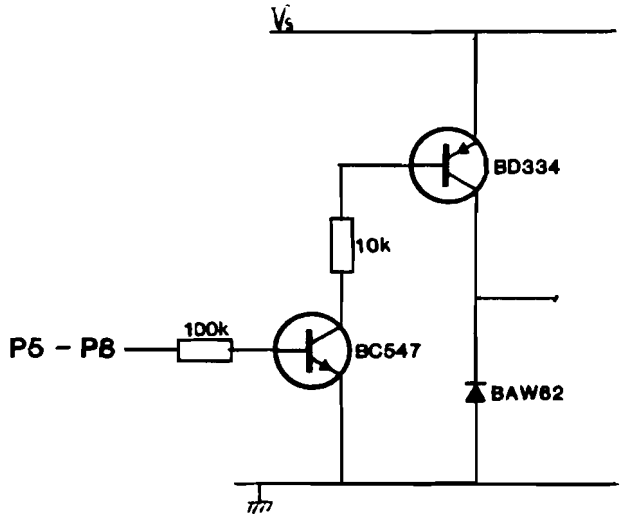
handinstelling met interface



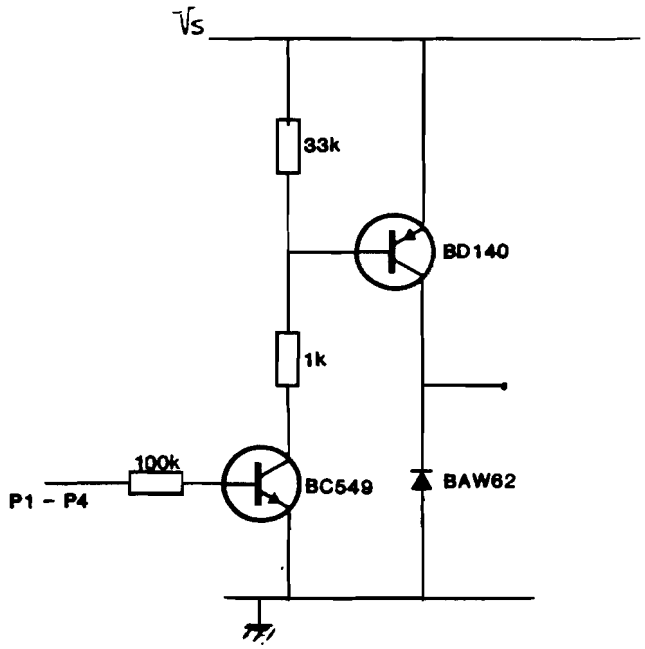
interface voor O2-sensor



drukopnemer met interface

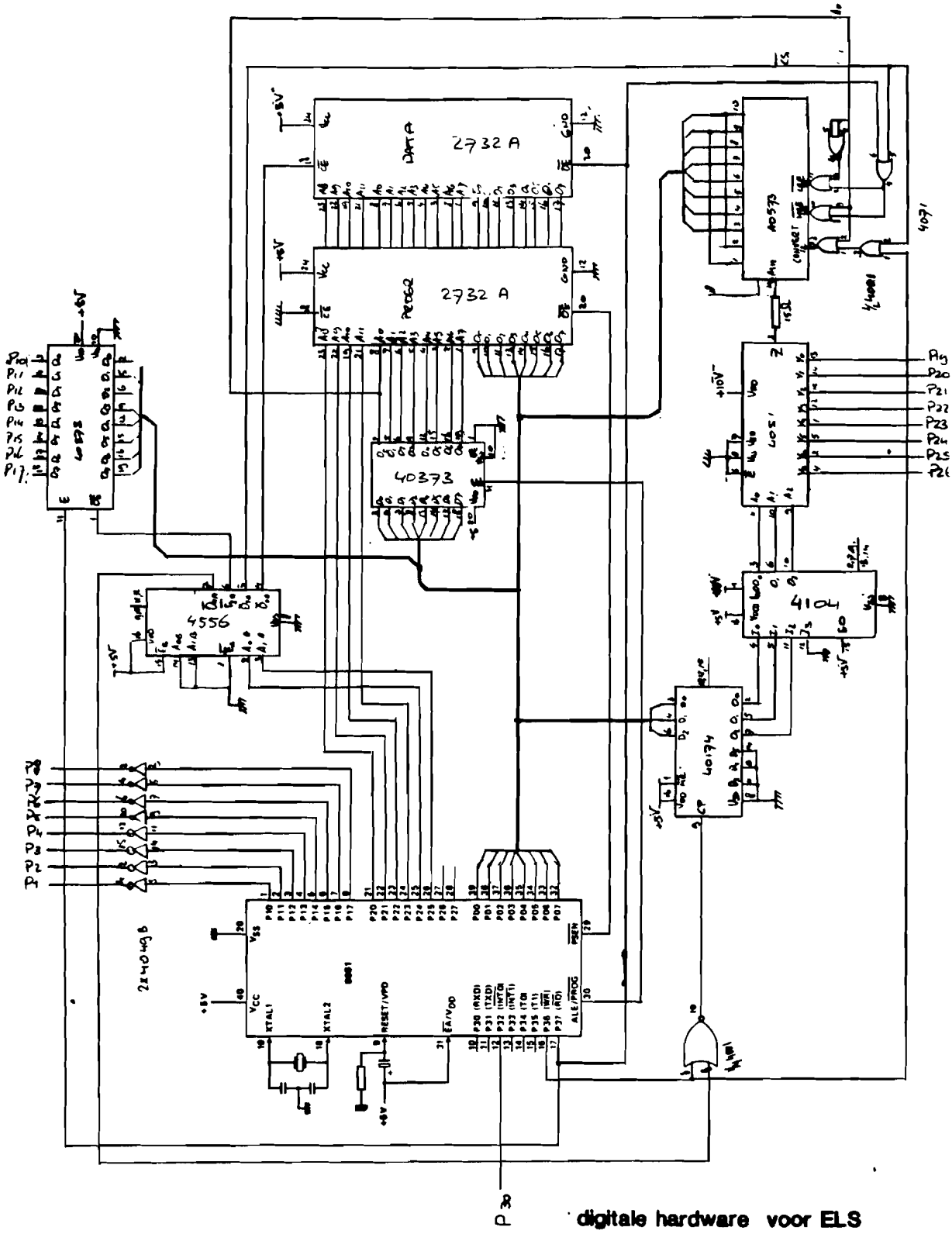


schakeling voor bekrachtiging van afsluiters en verwarming



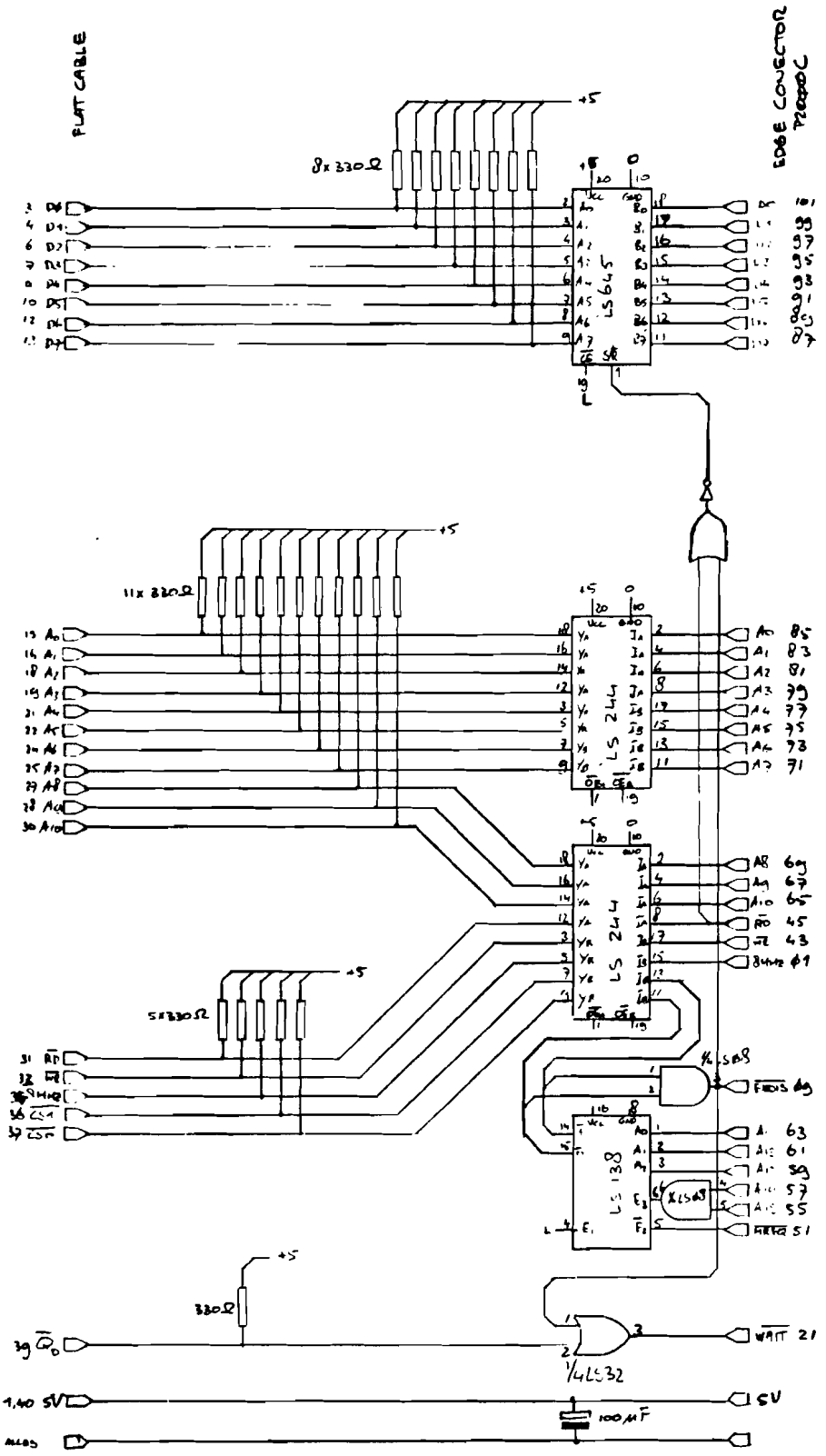
schakeling voor bekrachtiging van stappenmotor

Appendix B

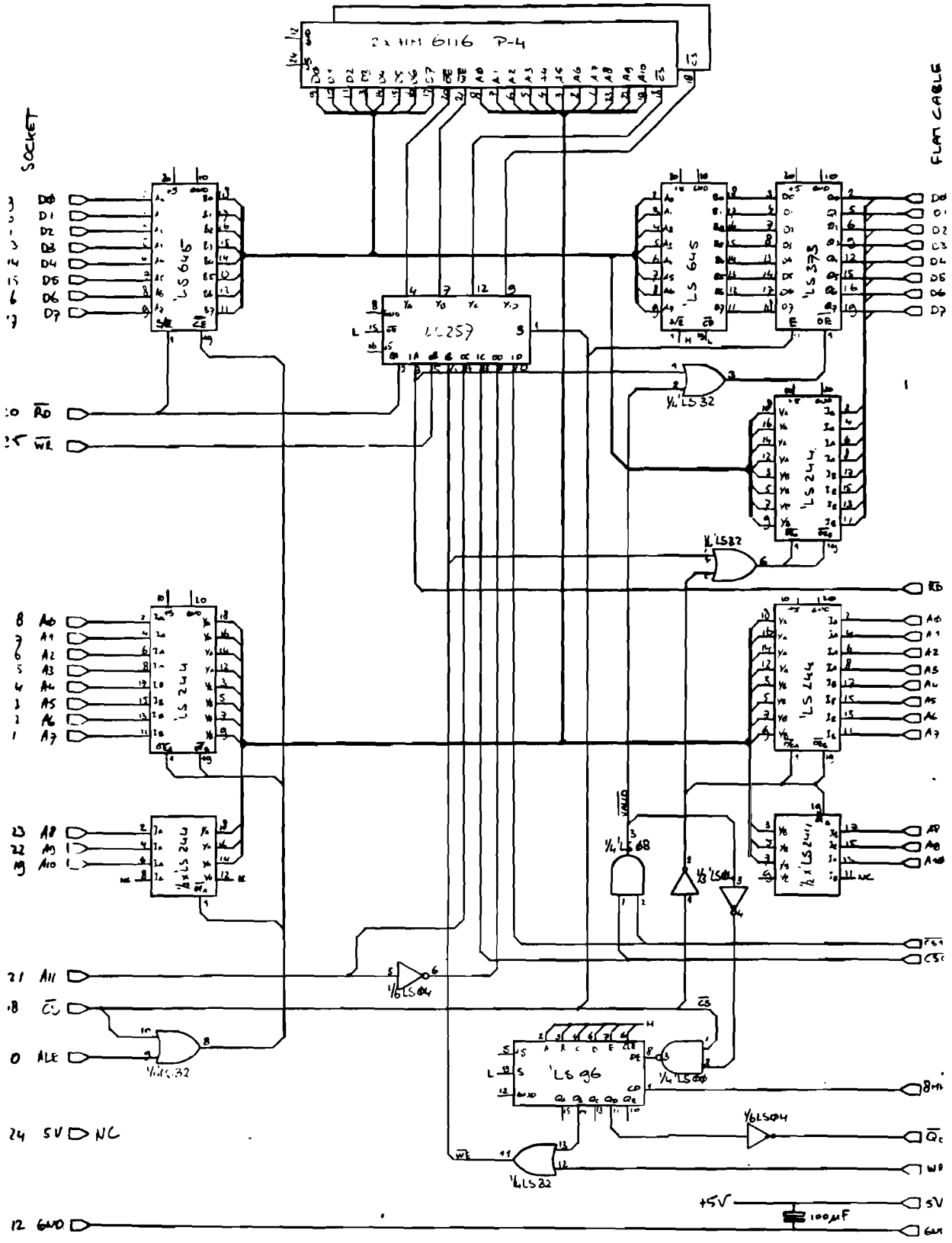


digitale hardware voor ELS

Appendix C



RAM-interface voor P2000C


2 x HM 6116 P-4
SOCKET
FLAT CABLE
'LS 645
'LS 245
'LS 244
'LS 373
'LS 257
1/4 'LS 32
1/6 'LS 04
1/4 'LS 08
'LS 96
RD
WR
ALE
+5V
100µF
GND

Ram-circuit

Appendix D PASCAL-listing van het regelalgorithme

```
INTERRUPT INT0:                                    {treedt op bij elke motorcyclus}

BEGIN
  TR1:=FALSE;                                                        {stop timer}
  TIMEL:=TL1;
  TIMEM:=TH1;                                      {lees inhoud van hardwaretimer1}
  TIMEH:=EXTTIM;
  TL1:=15;TH1:=0;                                  {zet hardwaretimer1 weer op nul}
  EXTTIM:=0;                                    {zet overflowbyte van timer1 op nul}
  TR1:=TRUE;                                                  {start timer opnieuw}
  flTOV:=FALSE;                                                {clear overflowvlag}
  flIGN:=TRUE;                                              {set vlag voorontsteking}
END;

INTERRUPT TIMER1   {treedt op bij overflow van de 16-bits hardwaretimer TIMER1}

BEGIN
  save(PSW,ACC);                                          {save status en accumulator}
  EXTTIM:=EXTTIM+1;
  IF EXTTIM=5                                             {wanneer te lange tijd dan:}
  THEN flTOV:=TRUE;                                           {set vlag voor overflow}
  restore(ACC,PSW)
END;


INTERRUPT TIMER0:                                 {treedt elke 256 microseconde op}

BEGIN {interrupt timer 0}
  save(registers);                               {save registers en pointers op stack}
  PSW:=00001000B;                                             {secteer registerbank 1}
  IF softtim3>0
  THEN softtim3:=softtim3-1
  ELSE flSTF:=TRUE;
  IF LPG/B
  THEN
    BEGIN {tellers bijwerken}
      IF softtim2>0
      THEN softtim2:=softtim2-1
      ELSE flDT:=TRUE;
      IF softtim1>0
      THEN softtim1:=softtim1-1
      ELSE timout:=TRUE;
    END;
  IF flIGN
      THEN
        BEGIN
          Mtemp:=65536*TIMEH+256*TIMEM+TIMEL;                {tijd in microseconden}
          Mtemp:=Mtemp*(nr_cyl);                        {omrekenen naar een viercylinder}
          Mtemp:=(Mtemp DIV 16);                            {in eenheden van 4 microsec}
          RPS:=(RPS+Mtemp)/2;                      {bepaling voortschrijdend gemiddelde}
          RPSH:=RPS DIV 256;RPSL:=RPS MOD 256;
          memwrite(postbox,RPSH);memwrite(postbox+1,RPSL);
          memwrite(postbox+2,Teng);memwrite(postbox+3,Tair);
          memwrite(postbox+4,Tfuel);memwrite(postbox+5,Pin);
          memwrite(postbox+6,O2in);memwrite(postbox+7,O2tar);
          memwrite(postbox+8,POS);memwrite(postbox+9,POSout);
          memwrite(postbox+10,POSadd);memwrite(postbox+11,FH);
          memwrite(postbox+12,AH);memwrite(postbox+13,O2in);
          memwrite(postbox+14,O2tar;memwrite(postbox+15,RAM(index));
          flPBX:=TRUE;           {set flag to show that the postbox is updated}
          memwrite(postbox+16,flags);memwrite(postbox+17,errors);
```

```
        flIGN:=FALSE                                    {reset ignition flag}
      END;
  IF flSTP                    {wanneer de stappenmotor weer een stap kan doen:}
  THEN
    BEGIN
      softtim3:=stpint;                                          {reload timer}
      IF POS=POSout
      THEN                                  {als de targetpositie al is bereikt}
        IF POS=POSdec AND flDEC         {bereikte positie is deceleratiepositie}
        THEN
          BEGIN
            vdrval:=TRUE;                    {dan wordt de afsluiter gesloten}
            gasval:=TRUE;
          END
        ELSE                           {als de targetpositie nog niet is bereikt}
          BEGIN
            IF POS<POSout
            THEN                                                   {stap verder}
              IF POS<=POSmax
              THEN
                BEGIN
                  status:=(status+1) MOD 4;
                  POS:=POS+1
                END
            ELSE                                                    {stap terug}
              IF POSref
              THEN
                BEGIN
                  status:=(status+3) MOD 4;
                  POS:=POS-1
                END
              ELSE POS:=0;                       {herdefinitie referentiepositie}
            digout:=(digout AND stpmsk)
            digout:=(digout OR sequence[status]);
            flSTP:=FALSE
          END;
    END;
  restore(registers);
END;


{*** MAIN PROGRAM ***}

{*** INIT ***}

BEGIN
  EA:=FALSE;                                        {disable alle interrupts}
  digout:=255;                                      {alle actuators inactief}
  SP:=stackpointer;                                   {initieer stackpointer}
  memwrite(sel_addr,0);                        {selecteer eerste analoge signaal}
  wait(20 microsec);                              {spanning moet stabiel worden}
  memwrite(AD_addrL,dummy);                                    {start converter}
  FOR i:=0 to 128 DO ram[i]:=0;                                      {clear RAM}
  dig:=memread(dig_addr);                           {inlezen digitale inputs}
  TH1:=0;TL1:=0;                                    {initieer hardwaretimers}
  TH0:=0;
  POSadd:=0;PADtem:=0;       {definieer POSadd en de backup van POSadd (PADtem)}
  status:=0;                                 {definieer status van de stappenmotor}
  softtim3:=stepint;              {initieer timer op interval voor stappenmotor}
  sfrIP:=xxxx1x01B;                                  {3:1=timer1 high priority}
                                                     {1:0=timer0 low priority}
                                                     {0:1=INT0 high priority}
  sfrTMOD:=00010010B;                          {7:0=gate timer1 gesloten}
```

```
                                          {6:0=timer1 is timer}
                                          {5 en 4:01=timer1 is 16-bit timer}
                                          {3:0=gate timer0 gesloten}
                                          {2:0=timer0 is timer}
                                          {1 en 0:10=timer0 is 8-bit autoload}
  sfrTCON:=x0x1xxx1B;                             {6:0=stop timer1}
                                                  {4:1=start timer0}
                                                  {0:1=INT0 is edge triggered}
  sfrIE:=1xx01011B;                       {7:1=globale interrupt enable}
                                          {4:0=disable interrupt seriele poort}
                                          {3:1=enable interrupt timer1}
                                          {2:0=disable externe interrupt 1}
                                          {1:1=enable interrupt timer0}
                                          {0:1=enable externe interrupt 0}

{*** REFERENTIE ***}

  POS:=255;     {voorlopige definitie van POS en POSout, motor gaat nu stappen}
  POSout:=0;
  REPEAT
    dig:=memread(digadr);
  UNTIL (POS=0) or not-POSref;                  {wacht tot POSref is bereikt}
  IF POS=0
  THEN error(0)
  ELSE
    BEGIN
      TR0:=FALSE;               {stop stappenmotor voor debouncing van POSref}
      wait(4 millisec);
      TR0:=TRUE;                                      {start stappenmotor weer}
      POS:=0;POSout:=4;                  {laat motor loskomen van referentie}
      REPEAT
        dig:=memread(digadr);
      UNTIL (POS=4) or POSref;
      IF POS=4
      THEN error(1);
    END;
  POS:=1;
  POSout:=POSstrt
  totAD;                        {inlezen en converteren van de analoge inputs}

{*** CHOKE FASE ***}

  IF LPG/B                                    {wanneer op LPG wordt gestart:}
  THEN
    BEGIN
      softtim1:=getROM(tCH,Teng);                        {zet time-out-timer op tCH}
      timout:=FALSE;
      vdrval:=FALSE;gasval:=FALSE;tnkval:=FALSE;            {open afsluiters}
      REPEAT
        dig:=memread(dig_addr);                                 {tot:-start}
      UNTIL fIIGN OR KL50 OR not-LPG/B OR timout;           {-tCH is verlopen}
    END;
  IF not-fIIGN AND not-KL50
  THEN
    IF LPG/B
    THEN
      BEGIN
        vdrval:=TRUE;                                     {sluit VDR-afsluiter}
        softtim1:=tOP;
        timout:=FALSE;
        REPEAT
          dig:=memread(digadr);
        UNTIL fIIGN OR KL50 OR not-LPG/B OR timout;
```

```
      END;
TO:
  IF not-flIGN AND not-KL50
  THEN
    BEGIN
      tnkval:=TRUE;gasval:=TRUE;
      vdrval:=TRUE;htr_02:=TRUE;      {sluit alle afsluiters en stop verwarming}
      REPEAT
        dig:=memread(digadr);
      UNTIL flIGN OR KL50
    END;

{*** START FASE ***}

  IF LPG/B
  THEN
    BEGIN
      vdrval:=FALSE;gasval:=FALSE;tnkval:=FALSE;              {open afsluiters}
      softtim1:=2*17;{sec}
      timout:=FALSE;
      REPEAT                                                    {wachten tot:}
        dig:=memread(dig_addr);                      {met starten wordt gestopt}
      UNTIL NOT-KL50 OR timout;                                 {34 sec na start}
      softtim1:=0;

{*** MAIN LOOP ***}

  htr_02:=FALSE;                          {schakel verwarming van 02-sensor in}
  softtim2:=8*17;{sec}                        {initieer Dt-timer met 2 minuten}
  REPEAT
    IF flTOV THEN GOTO TIME_OUT;                      {bepalen of de motor loopt}
    totAD;                          {inlezen en converteren van de analoge inputs}
    wrkpnt;                                             {bepaal het werkpunt}
    IF LPG/B                                               {Alleen voor LPG:}
    THEN
      BEGIN
        IF not-flDEC
        THEN
          BEGIN
            IF RPS>RPSoff
            THEN
              BEGIN
                tnkval:=TRUE;gasval:=TRUE;vdrval:=TRUE; {sluit alle afsluiters}
              END
            ELSE
              IF RPS<RPSon
              THEN
                BEGIN
                  tnkval:=FALSE;gasval:=FALSE;vdrval:=FALSE;  {open afsluiters}
                END;
            IF flDt
            THEN                                        {nieuwe POSadd berekenen}
              BEGIN
                softtim2:=Dt;
                02tar:=tabval(02bas,pinrps,rpsprt,pinprt);
                02tar:=02tar*getROM(FE,Teng);
                FHval:=getROM(FH,Rend);
                02tar:=02tar*FHval;
                IF 02tar>02low;
                THEN
                  BEGIN
                    IF 02in>02tar THEN POSadd:=POSadd+S;
                    IF 02in<02tar THEN POSadd:=POSadd-S;
```

```
                END
              ELSE
                IF RPS<RPSidle
                THEN
                  BEGIN
                    AHval:=getROM(AH,Ridle);
                    n:=div0_2(RPS-RPSmin,RPSidle-RPSmin)
                    POSadd:=AHval+(PADtem-AHval)*n
                  END
                ELSE POSadd:=POSadd;
              PADtem:=POSadd;                          {update backup of POSadd}
            END; {bepaling van POSadd}
          IF allFF(O2bas,pinrps)
          THEN
            BEGIN POSout:=POSdec;flDEC:=TRUE END;
          ELSE
            BEGIN
              POStem:=tabval(POSbas,pinrps,rpsprt,pinprt);
              Raf:=div0_2(Tair,Tfuel);
              POStem:=POStem+getROM(ATaf,Raf);
              POStem:=POStem+getROM(AE,Teng);
              POStem:=POStem+POSadd;
              IF THop THEN POStem:=POStem*Ef;
              POSout:=POStem;
            END;
        END
      ELSE IF allNFF(O2bas,pinrps) THEN flDEC:=FALSE;
    END
  ELSE tnkval:=TRUE;gasval:=TRUE;vdrval:=TRUE;        {sluit alle afsluiters}
UNTIL false
END.

{*** PROCEDURES en FUNCIONS ***}

PROCEDURE totAD;
BEGIN
  sel:=0
  flLAD:=FALSE;
  REPEAT
    tempH:=memread(AD_addrL);              {haal resultaat van de AD-converter}
    tempL:=memread(AD_addrH);
    sel:=sel+1;
    IF sel=7 THEN BEGIN flLAD:=TRUE;sel:=0 END;
    memwrite(sel_addr,sel);
    wait(20 microsec);{signaal kan nu stabiel worden}
    memwrite(AD_addrH,dummy);                                {start converter}
    IF flLAD THEN sel:=6 ELSE sel:=sel-1;
    temp:=scale(tempH,sel);                        {over hele bereik verdelen}
    input[sel]:=lintab(temp,sel);                              {lineariseren}
    IF not-flLAD THEN sel:=sel+1;
  UNTIL flLAD
END;

FUNCTION scale(value,sel):BYTE;
VAR n:INTEGER
BEGIN
  n:=n-offset[sel];
  IF n<=0
  THEN n:=0
  ELSE
    BEGIN
      n:=n*mulfac[sel];
      IF n>255 THEN n:=255;
```

```
    END;
  scale:=n;
END;

FUNCTION lintab(value,sel):BYTE;
VAR n:INTEGER;
BEGIN
  tabadr:=getROM(adradr+2*sel)*256+getROM(adradr+2*sel+1)
  n:=getROM(tabadr,value);              {haal waarde uit geselecteerde functie}
  lintab:=n;
END;

FUNCTION getROM(tabadr,value):BYTE;
BEGIN
  part:=value mod 16;
  IF part=0 THEN row:=15;
            ELSE row:=value div 16;
  getVAL(tabadr+row,part);
END;

FUNCTION divoct(val2,val1):BYTE;
BEGIN
  n:=val2/val1;
  divoct:=round(16*n);
END;

PROCEDURE wrkpnt;
BEGIN
  n:=memread(sc_rps);
  IF n=0
  THEN BEGIN part:=0;row:=0 END
  ELSE
    BEGIN
      i:=0;
      REPEAT
        n:=memread(sc_rps+i);
        i:=i+1;
      UNTIL n>RPSH OR i=16
      IF i=16
      THEN BEGIN part:=0;row:=15 END
      ELSE
        BEGIN
          RPS:=256*RPSH + RPSL
          x1:=memread(sc_rps+i-1)-RPS;
          x2:=memread(sc_rps+i-1)-memread(sc_rps+i)
          n:=divoct(x1,x2);
          IF n=16 THEN i:=i+1;
          i:=i-1;                   {omdat de counter een verder is dan de rij}
          row:=i;part:=n;
        END;
    END;
  rpsrow:=row;rpsprt:=part;
  pinrow:=Pin div 16;
  pinprt:=Pin mod 16;
  pinrps:=16*pinrow + rpsrow;                                    {werkpunt}
  memwrite(pin_rps,pinrps);                    {naar postbox voor 'commands'}
END;
```

```
FUNCTION div0_2(val1,val2):BYTE;
BEGIN
  IF val2=0
  THEN n:=255
  ELSE
    BEGIN
      n:=0;i:=8;
      x:=val1 div val2;y:=val1 mod val2;
      IF x>1                                                    {resultaat > 2}
      THEN n:=255                                          {=1.111 1111 binair}
      ELSE
        BEGIN
          REPEAT
            n:=2*n+x;
            val1:=y*2;
            x:=val1 div val2;y:=val1 mod val2;
            i:=i-1;
          UNTIL i:=0
          n:=n+x;                                                  {afronding}
        END;
  div0_2:=n;
END;

FUNCTION allFF(tabadr,pinrps):BOOLEAN;
BEGIN
  allFF:=FALSE;
  IF memread(tabadr+pinrps)=255
  THEN
    IF memread(tabadr+pinrps+1)=255
    THEN
      IF memread(tabadr+pinrps+16)=255
      THEN
        IF memread(tabadr+pinrps+16+1)=255
        THEN allFF:=TRUE;
END;

FUNCTION allNFF(tabadr,pinrps):BOOLEAN;
BEGIN
  allNFF:=FALSE;
  IF memread(tabadr+pinrps)<>255
  THEN
    IF memread(tabadr+pinrps+1)<>255
    THEN
      IF memread(tabadr+pinrps+16)<>255
      THEN
        IF memread(tabadr+pinrps+16+1)<>255
        THEN allNFF:=TRUE;
END;

FUNCTION tabval(tabadr,pinrps,rpsprt,pinprt):BYTE;
BEGIN
  x1:=getVAL(tabadr+pinrps,rpsprt);          {interpoleer rps in rij pinrow}
  x2:=getVAL(tabadr+pinrps+16,rpsprt);     {interpoleer rps in rij pinrow+1}
  tabval:=inpltn(x1,x2,pinprt);
END;

FUNCTION getVAL(adr,part):BYTE;
BEGIN
  x1:=memread(adr);
  x2:=memread(adr+1);
  getval:=inpltn(x1,x2,part);
END;
```

```
FUNCTION inpltn(val1,val2,part):BYTE;
BEGIN
  x:=val2-val1;
  subflg:=(x<0);
  x:=(x*part) div 16;
  IF subflg THEN inpltn:=val1+x
            ELSE inpltn:=val1-x;
END;
```