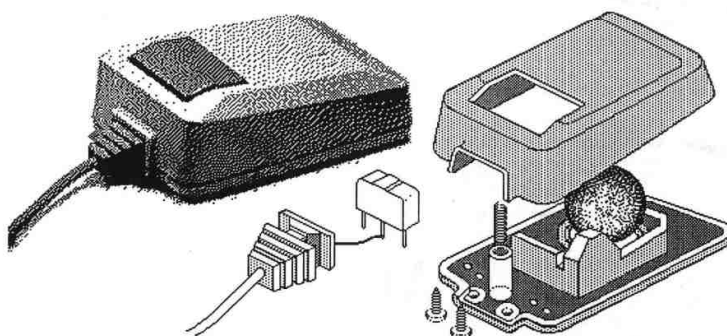




De μ P Kenner

Achttiende jaargang nr. 3
Oktober 1994

87



In dit nummer o.a.:

DOS65: RAM(p)kaart
Cursus C: In de plus!
Toetsenborden gevaarlijk?
Linux: geïnstalleerd, en dan?

Inhoudsopgave

De μ P Kenner

Nummer 87, oktober 1994
 Verschijnt 5 maal per jaar
 Oplage: 250 stuks
 Druk: FEBO Offset, Enschede

De redactie:

Nico de Vries
 Gert van Opbroek
 Geert Stappers

Eindredactie:

Nico de Vries

Vormgeving:

Nico de Vries

Redactieadres:

p/a Nico de Vries
 Van der Waalsstraat 46
 2984 EP Ridderkerk

De μ P Kenner nummer 88 verschijnt op
 17 december 1994.

Kopijsluitingsdatum voor nummer 88 is
 vastgesteld op 3 december 1994.

Vereniging

Uitnodiging voor de clubbijeenkomst	5
Algemene ledenvergadering van de KIM Gebruikersclub	
Nederland	6
Voortgang KGN68k (deel 17)	27
Balans per 31 december 1993	31

Algemeen

Redactioneel	4
Een gevaarlijke hobby?	14
Van de voorzitter	33

Systemen

A.OUT to S37	7
Linux, after install (deel 2)	19

Talen/software/DOS65

DOS65: RAM(P)-kaart?	15
C + + de opvolger van C (deel 1)	22

De μ P Kenner is het huisorgaan van de KIM gebruikersclub Nederland en wordt bij verschijnen gratis toegezonden aan alle leden van de club. De μ P Kenner verschijnt vijf maal per jaar, in principe op de derde zaterdag van de maanden februari, april, augustus, oktober en december.

Kopij voor het blad dient bij voorkeur van de leden afkomstig te zijn. Deze kopij kan op papier, maar liever in machine-leesbare vorm opgestuurd worden aan het redactieadres. Kopij kan ook op het Bulletin Board van de vereniging gepost worden in de redactie area. Nadere informatie kan bij het redactieadres of via het bulletin board opgevraagd worden.

De redactie houdt zich het recht voor kopij zonder voorafgaand bericht niet of slechts gedeeltelijk te plaatsen of te wijzigen. Geplaatste artikelen blijven het eigendom van de auteur en mogen niet zonder diens voorafgaande schriftelijke toestemming door derden gepubliceerd worden, in welke vorm dan ook.

De redactie noch het bestuur kan verantwoordelijk gesteld worden voor toepassing(en) van de geplaatste kopij.

Redactioneel

Daar zijn we weer. Op de HCC-dagen bedoel ik dus. Want de KGN is ook deze keer weer op de HCC-dagen present. We zijn benieuwd naar de afloop van die dagen. Want zo uniek als ze vroeger waren zijn ze niet meer:

- De prijzen van hardware zitten tot op het bot
- Er zijn tegenwoordig ongeveer ieder weekend een soort mini HCC-dagen in de sporthal bij u in de buurt

Tel dat maar even op en je komt tot de conclusie dat die goedkope CD-ROM drive en het nieuwe moederbord dit jaar al in de sporthal zijn gekocht. En daarmee valt een belangrijke reden om naar Utrecht te komen en je in de gigantische drukte te storten een beetje weg. We zullen wel zien.

Voor de leden zijn er echter wel een paar redenen te bedenken om naar de HCC-dagen te komen. Wat dacht u hiervan:

- Op vrijdag (dus niet op zaterdag) korting op de toegangsprijs. Wel even het bonnetje in dit nummer uitknippen...
- Op de stand van de KGN staat de koffie klaar.
- Op diezelfde stand kunt u uw aankoop vast even rustig bekijken en als de betreffende hardware er is ook nog even proberen.
- Geen gesjouw met tassen en (te) warme jassen: mag u op de stand dumpen.
- Verder kunt de resultaten van KGN68k en DOS65 in het echt bewonderen.
- En zo zie je ook nog eens een paar andere leden van onze club.

Kortom: tot ziens op vrijdag 18, of zaterdag 19 november!

Op het front van DOS65 is er een beetje nieuws, dat echter niet door iedereen als positief zal worden ervaren: de RAMkaart wordt veranderd. Gelukkig betreft het alleen de PAL op de kaart, zodat daar de ramp wel te overzien is. In de software is er meer werk aan de winkel, en worden een aantal verandering juist weer ongedaan gemaakt. Verderop in dit nummer leest u er alles over. Verder is de werkgroep nog druk bezig. Het nieuwste projectje is de

combokaart, dat een soort vergaarbak gaat worden van alles wat Elektuur vergeten was of een beetje krom heeft opgelost. Op het kaartje komen waarschijnlijk de volgende zaken: tweede RS-232 chip, de 6818 real time clock met externe 32kHz oscillator en een NiCd accu, fatsoenlijke generators voor reset en NMI, IO65 EPROM (netjes gedecodeerd en gebufferd, want op de originele Elektuur CPU-kaart hangt de arme EPROM rechtstreeks en geheel ongebufferd aan de bus) en als er ruimte overblijft een speakerinterface en een soort systeemthermometer. Dat laatste is een stel LEDs gekoppeld aan een EPROM. De EPROM wordt met de adreslijnen aan de hoogste adreslijnen van de bus gehangen. De LEDs hangen aan de datalijnen van EPROM. Kun je, als de EPROM tenminste een beetje slim geprogrammeerd wordt, precies zien waar de CPU in het geheugen aan het stoken is.

De KGN68k vordert behoorlijk. Alleen met de printen wil het maar niet lukken. Dus als u een keertje Ultiboard wilt leren en de tanden eens op een appetijtelijke multilayer wilt stukbijten ligt hier een unieke gelegenheid. Het resultaat van de werkgroep KGN68k zal op de HCC-dagen te bewonderen zijn.

De bijeenkomst in Amersfoort is een week na de HCC-dagen. Zoals de statuten voorschrijven houden we dan een ledenvergadering, want er staan weer wat wisselingen in het bestuur voorgeschreven. Als zo'n bestuursklus u wat lijkt, even Geert Stappers aanschieten. Het is altijd gezellig, het kost niet veel tijd (minimaal 1 vrijdagavond per 2 maanden) en u bent direct betrokken bij het reilen en zeilen van club en kunt daar nog rechtstreeks invloed op uitoefenen ook.

Verder in Amersfoort een Linux-happening. De bedoeling dat we weer een aantal Linux-wonders aan elkaar gaan knopen zodat er weer wat megabytes heen en weer geslingerd kunnen worden. Kortom, feest genoeg. Tot ziens in Amersfoort.

Nico de Vries

Voor de leden zijn er echter wel een paar redenen te bedenken om naar de HCC-dagen te komen.

KGN op de HCC-dagen!!! Stand L54.6

Uitnodiging voor de clubbijeenkomst

Datum: 26 november 1994
 Lokatie: Bu "Het Klokhuis"
 He Weberstraat 2
 Amersfoort
 Tel: 033-700683
 Thema: Lin Linux / Netwerken

Routebeschrijving

Vanuit Utrecht:

Afslag Woudenberg / Leusden
 Na de eerste twee stoplichten linksaf, Bij het volgende stoplicht wederom linksaf en dan onder het viaduct door. Je bevindt je nu op de Arnhemseweg. Zie verder punt *

Vanuit Amsterdam / Apeldoorn:

Na verkeersplein Amersfoort, derde afslag nemen (Leusden zuid). Onderaan de afslag ga je rechts af. Je zit dan op de Arnhemseweg. Zie verder punt *

Vanuit Arnhem:

Autoweg A2 richting Den Haag. Afslag Maarsbergen nemen, hier richting Woudenberg gaan. Alsmaar op deze weg blijven. Door Leusden heen, in Leusden is dit reeds de Arnhemse weg. Vervolg de weg naar Amersfoort. Zie verder punt *

* In Amersfoort:

Na het bord Amersfoort (stadsbord) komt u eerst een stoplicht tegen (oversteekplaats voor voetgangers/fietsen). De weg vervolgen. De weg komt dan op een soort van "T" splitsing uit. De Arnhemse weg verloopt naar rechts over het Spoor. Direct na de spoorovergang komt u op een 5 sprong. Hier moet je rechts aanhouden (of eigenlijk de tweede straat van rechts nemen). Dit is de Gasthuislaan. Volg daar eventueel het bord ziekenhuis "Eemland". Vervolg deze weg totaan de eerste stoplichten. Ga hier rechtdoor, dit is de Ringweg Rand-

enbroek. Ga dan de derde straat rechts af dit is de Weberstraat. Waarna het buurthuis "Het Klokhuis" na +/- 50 meter links in de bocht staat.

Openbaar vervoer

Vanaf station Amersfoort: met stadsdienst 7 of 8. Uitstappen op het Euterpeplein. Hier bevindt zich ook een winkelcentrum. Men dient dan een stuk terug te lopen richting de Ringweg Randenbroek (+/- 100m). De straat oversteken en men komt dan in de Weberstraat. Na +/- 50 meter bevindt zich dan het buurthuis "Het Klokhuis" in de bocht.

Programma:

9:30 Zaal open met koffie
 10:15 Opening
 10:16 Algemene Ledenvergadering. Voor de Agenda, zie volgende bladzijde
 11:30 Demonstratie van de aanwezige Linux systemen, netwerken en ontwikkeltools t.b.v. KGN68k
 12:30 Lunch, consumpties tegen betaling
 13:30 Forum en Markt
 Aansluitend het informele gedeelte met de mogelijkheid om andermans systemen te bewonderen en Public Domain software uit te wisselen. U en uw systeem zijn uiteraard van harte welkom.
 17:00 Sluiting.



Algemene ledenvergadering van de KIM Gebruikersclub Nederland

Datum: 26 november
Plaats: Amersfoort

Behandelde agendapunten:

Agenda:

- 1: Opening, vaststelling agenda
- 2: Verslag van de vorige ledenvergadering
- 3: Mededelingen
Status project KGN68k
Status DOS-65
- 4: Concept-begroting 1994
- 5: Verkiezing kascontrole-commissie 1994
- 6: Verkiezing bestuursleden
- 7: Eventuele overig agendapunten
- 8: Rondvraag

- 1: Het verslag van de ledenvergadering d.d. 25-09-1993 zoals afgedrukt in μ P Kenner 83 is goedgekeurd.
- 2: Het jaarverslag werd door de leden, na enige uitleg van de penningmeester en op voorstel van de commissie kascontrole, goedgekeurd.
- 3: Er waren geen overige agendapunten of punten voor de rondvraag.

Een kopie van het goedgekeurde jaarverslag is op te vragen bij de secretaris of de penningmeester.

Toelichting:

Ad. 2:

De vorige ledenvergadering was in eerste instantie uitgeschreven voor 19 maart in Almelo. Het belangrijkste agendapunt was goedkeuring van het jaarverslag over 1994. Helaas kon dit niet doorgaan omdat de kascontrole niet had plaatsgevonden. De ledenvergadering is toen verzet naar de volgende bijeenkomst.

Op 28 mei in Haarlem is vervolgens de ledenvergadering gehouden. Het jaarverslag was ondertussen door de kascontrolecommissie goedgekeurd.

Ad 6:

Dit jaar treden af als bestuursleden:

Jacques Banser (penningmeester)
Gert van Opbroek (secretaris)
Nico de Vries (redactie-secretaris)

Deze personen zijn om persoonlijke redenen niet herkiesbaar. Het bestuur roept de leden op nieuwe kandidaten voor te stellen voor het bestuur. Aanmeldingen van kandidaat-bestuursleden kunnen tot uiterlijk voor de behandeling van dit agendapunt gedaan worden bij de voorzitter Geert Stappers.

De werkgroep KGN68k zoekt:

Een enthousiaste PRINT LAYOUTER

voor het lay-outen van multilayerprinten ten behoeve van het KGN68k-project.

Om het werk te kunnen uitvoeren zal een Ultiboard print-ontwerppakket ter beschikking gesteld worden.

Ervaring met dit pakket is daarom gewenst.

Geïnteresseerd? Neem dan contact op met de projectgroep KGN68k op telefoon 04871-41279

A.OUT to S37

De volgende source is een concreet resultaat van KGN68k. Het programma zet een zogenaamde 'A.OUT' file om in een S37 records file. 'A.OUT' komt de GNU-linker en S37 records worden begrepen door EPROM programmers en door de monitor in KGN68k.

Het stuk software is prachtig geschreven, het laat zich lezen als een roman. Ondanks het feit dat er

nauwlijks commentaar is staat is het toch duidelijk wat er gebeurt.

Voor een andere convertor is deze file een prachtig uitgangspunt. De source staat ook op "The Ultimate".

Geert Stappers met medeweten van Pieter de Visser.

```

/* aout2s37.c - convert a.out file to S37 file
 *
 * Author:
 *      Pieter de Visser
 *
 * Log:
 *      23-Aug-92 Initial version
 *      7-Nov-92 Corrected length of S7-record
 *      7-Nov-92 Last update
 */

#include <stddef.h>
#include <stdio.h>
#include <stdlib.h>
#include <stdarg.h>
#include <string.h>
#include <errno.h>
#include <a.out.h>

#define SEGSIZE      4096      /* Segment size used in a.out file */
#define IBUFLEN      4096
#define OBUFLEN      8192
#define SRLEN        16
#define BCD(x)        ((x) < 10 ? '0' + (x) : 'A' + ((x) - 10))
#define C4TOL(cp)      ((cp)[0] < 24 | (cp)[1] < 16 | (cp)[2] < 8 | (cp)[3])

static char  *ifilename,
             *ofilename;
static FILE  *ifid,
             *ofid;

```

Fig. 1: source file AOUT3S37.C

```
static unsigned char    obuf[OBUFLen + (SRLen + 7) * 2 + 1],
                        *obufp;
static unsigned long    oaddress;
static unsigned         check_sum;
```

```
static void parse_command_line (int, char **);
static void open_files (void);
static void close_files (void);
static void convert (void);
static void read_exec (struct exec *);
static void check_exec (struct exec *);
static void process_segm (unsigned long);
static void process_bss (unsigned long);
static void process_entry (unsigned long);
static void write_S3 (unsigned char *, int);
static void write_zero_S3 (int);
static void write_header (int, int, unsigned long);
static void write_chksum (void);
static void write_byte (unsigned char);
static void flush_output (void);
static void usage (void);
static void die (char *, ...);
```

```
int main (int argc, char **argv)
{
    parse_command_line (argc, argv);
    open_files ();
    convert ();
    close_files ();
    return 0;
}
```

```
static void parse_command_line (int argc, char **argv)
{
    int i;
    ifilename = NULL;
    ofilename = NULL;
    oaddress = 0;
    for (i = 1; i != argc; i++)
    {
        char *p = argv[i];
        if (p[0] == '-' && p[1] != 0)
        {
            if (p[1] == 'b')
```

```
        oaddress = strtoul (&p[2], NULL, 0);
    else
        usage ();
    }
else if (ifilename == NULL)
    ifilename = p;
else if (ofilename == NULL)
    ofilename = p;
else
    usage ();
}
if (ofilename == NULL)
    usage ();
}

static void open_files (void)
{
    if (strcmp (ifilename, "-") == 0)
        ifid = stdin;
    else
    {
        ifid = fopen (ifilename, "rb");
        if (ifid == NULL)
            die ("Can't open %s: %s", ifilename, strerror (errno));
    }
    if (strcmp (ofilename, "-") == 0)
        ofid = stdout;
    else
    {
        ofid = fopen (ofilename, "w");
        if (ofid == NULL)
            die ("Can't create %s: %s", ofilename, strerror (errno));
    }
}

static void close_files (void)
{
    if (ifid != stdin)
        (void) fclose (ifid);
    if (obufp != obuf)
        flush_output ();
    if (ofid != stdout)
        (void) fclose (ofid);
}
```

```
static void convert (void)
{
    struct exec exec;
    read_exec (&exec);
    check_exec (&exec);
    obufp = obuf;
    if (fseek (ifid, SEGSIZE, SEEK_SET) != 0)
        die ("Can't seek to text segment: %s", strerror (errno));
    process_segm (exec.a_text);
    process_segm (exec.a_data);
    process_bss (exec.a_bss);
    process_entry (exec.a_entry);
}
```

```
static void read_exec (struct exec *exec)
{
    unsigned char buf[sizeof (struct exec)];
    if (fread (buf, 1, sizeof (struct exec), ifid) != sizeof (struct exec))
        die ("Error reading header: %s", strerror (errno));
    exec->a_info = C4TOL (&buf[0]);
    exec->a_text = C4TOL (&buf[4]);
    exec->a_data = C4TOL (&buf[8]);
    exec->a_bss = C4TOL (&buf[12]);
    exec->a_syms = C4TOL (&buf[16]);
    exec->a_entry = C4TOL (&buf[20]);
    exec->a_trsize = C4TOL (&buf[24]);
    exec->a_drsize = C4TOL (&buf[28]);
}
```

```
static void check_exec (struct exec *exec)
{
}
```

```
static void process_segm (unsigned long nbytes)
{
    unsigned char buf[IBUFLEN],
                *bufp;
    int nbuf = 0;
    unsigned long nwritten = 0;
    while (nwritten != nbytes)
    {
        int len;
        if (nbuf == 0)
        {
            size_t n = IBUFLEN;
```

```

        if (n > nbytes - nwritten)
            n = nbytes - nwritten;
        if (fread (buf, 1, n, ifid) != n)
            die ("Error reading segment: %s", strerror (errno));
        nbuf = n;
        bufp = buf;
    }
    len = Srlen;
    if (len > nbuf)
        len = nbuf;
    write_S3 (bufp, len);
    bufp += len;
    nbuf -= len;
    nwritten += len;
}
}

static void process_bss (unsigned long nbytes)
{
    unsigned long nwritten = 0;
    while (nwritten != nbytes)
    {
        int len = Srlen;
        if (len > nbytes - nwritten)
            len = nbytes - nwritten;
        write_zero_S3 (len);
        nwritten += len;
    }
}

static void process_entry (unsigned long entry)
{
    write_header (7, 0, entry);
    write_chksum ();
    if (obufp >= obuf + OBUFLen)
        flush_output ();
}

static void write_S3 (unsigned char *data, int len)
{
    int i;
    write_header (3, len, oaddress);
    for (i = 0; i != len; i++)
        write_byte (data[i]);
    write_chksum ();
}

```

```
        if (obufp >= obuf + OBUFLen)
            flush_output ();
        oaddress += len;
    }

static void write_zero_S3 (int len)
{
    int i;
    write_header (3, len, oaddress);
    for (i = 0; i != len; i++)
        write_byte (0);
    write_chksum ();
    if (obufp >= obuf + OBUFLen)
        flush_output ();
    oaddress += len;
}

static void write_header (int type, int len, unsigned long address)
{
    *obufp++ = 'S';
    *obufp++ = '0' + type;
    check_sum = 0;
    write_byte (len + 5);
    write_byte ((address >> 24) & 0xff);
    write_byte ((address >> 16) & 0xff);
    write_byte ((address >> 8) & 0xff);
    write_byte (address & 0xff);
}

static void write_chksum (void)
{
    write_byte (~check_sum & 0xff);
    *obufp++ = '\n';
}

static void write_byte (unsigned char byte)
{
    check_sum += byte;
    *obufp++ = BCD ((byte >> 4) & 0x0f);
    *obufp++ = BCD (byte & 0x0f);
}

static void flush_output (void)
{

```

```

        if (fwrite (obuf, 1, obufp - obuf, ofid) != obufp - obuf)
            die ("Write error: %s", strerror (errno));
        obufp = obuf;
    }

static void usage (void)
{
    fprintf (stderr, "Usage: aout2s37 [-baddr] input-file output-file\n");
    exit (1);
}

static void die (char *mess, ...)
{
    va_list ap;
    va_start (ap, mess);
    vfprintf (stderr, mess, ap);
    fprintf (stderr, "\n");
    va_end (ap);
    exit (1);
}

```



Deze bon is f 5,- waard

HCC Dagen

94

Inleveren bij de kassa van JAARBEURS UTRECHT
op vrijdag 18 november van 10.00 tot 18.00 uur

- ▶ Alléén vrijdag geldig
- ▶ Eén bon per persoon
- ▶ Reductie geldt alleen voor de entreprijs
- ▶ Deze bon mag niet worden verhandeld

HCC • postbus 149 • 3990 DC Houten • telefoon (03403) 7 87 88

Een gevaarlijke hobby?

In de CM Corporate van 7 september las ik een bericht waarover ik mij enigszins verbaasde. Samengevat luidt dit bericht als volgt:

In verband met de risico's voor de gezondheid van het werken met toetsenborden heeft Compaq besloten op de toetsenborden die naar de Verenigde Staten worden geëxporteerd de volgende mededeling af te drukken:

Warning! To reduce risk of serious injury to hands, wrists or other joints, read safety & comfort guide.

Dit is een gevolg van de drukte die is ontstaan over het gebruik van toetsenborden en de gevolgen die repeterende bewegingen voor de gezondheid kunnen hebben. Nu gaat men in de VS heel ver met het begrip gevolgschade. Daar, en overigens ook in de EEG, is de fabrikant aansprakelijk voor de gevolgen van de kwaliteit van een produkt. Als door een slechte constructie de slang van je wasmachine losschiet en je huis loopt onder, dan is de fabrikant aansprakelijk voor de gevolgen. Tot zover niets aan de hand. In de VS gaat men echter een flinke stap verder. Dat het roken van sigaretten slecht is voor de gezondheid, mag bekend worden verondersteld. Tegenwoordig staat er op elk pakje sigaretten dan ook een waarschuwing. Voor de Amerikaanse rechters is dit echter niet genoeg. Er is een grote groep rokers die ten gevolge van het roken nu leven op kosten van de staat (uitkering). Bovendien kost de behandeling van de medische problemen van deze mensen handen vol geld. De staat probeert nu deze kosten te verhalen op de fabrikanten van sigaretten. Het argument dat gebruikt wordt is dat de fabrikant bewust een artikel op de markt heeft gebracht met risico's voor de gezondheid. Met een verwijzing naar het begrip gevolgschade is de fabrikant dus aansprakelijk voor alle kosten die hieruit voortvloeien.

Hetzelfde geldt, gelukkig in mindere mate, voor het gebruik van typemachines en toetsenborden. Mensen die dag in dag uit op typemachines of toetsenborden zitten te tikken, kunnen hierdoor problemen met de gezondheid krijgen door blessures aan polsen en handen. De kosten die hiermee gemoeid zijn, zouden wel eens op de fabrikant van het toetsenbord verhaald kunnen worden en om die reden zet Compaq de bovengenoemde waarschuwing al vast maar op zijn toetsenborden.

Trekken we dit door, dan zijn er nog wel meer produkten waar een waarschuwing op moet komen:

**Niet
lezen
dus!
Zelfs dit
hokje
niet.
Krijg je
mis-
schien
claustro-
fobie van
of zo...**

- Dit operating system kan uw gezondheid schaden door een tekort aan nachtrust bij het installeren (Linux)
- Aanschaf van dit apparaat kan leiden tot een extreem hoge telefoonrekening (een modem)
- Lidmaatschap van deze computerclub kan leiden tot ernstige brandwonden (van het solderen) en oververmoeide ogen (vanwege het tot diep in de nacht werken achter een beeldscherm)
- Bezoek aan deze beurs leidt vrijwel zeker tot onverantwoorde uitgaven en kan een negatieve invloed op uw gezinsleven hebben (de HCC-dagen)

Ik denk dat we zo nog wel even door kunnen gaan. Let vooral ook op het volgende:

Attentie: De auteur van dit artikel waarschuwt ervoor dat het lezen van artikelen van zijn hand nadelig voor u kunnen zijn en mogelijk schadelijk zijn voor uw gezondheid (vooral als de layouter weer eens van slag af was).

Gert van Opbroek

DOS65: RAM(P)-kaart?

Huidige status

Er nu verschillende kaarten gebouwd. Ze werken allemaal goed. Sommigen hebben moeite met het verkrijgen van het batterijtje, dat gespecificeerd is als een Lithium cel van 3.6 Volt. Die blijken schaars. De standaardwaarde van 3 Volt is ook goed. De Dallas DS1210 battery controller vindt namelijk dat de batterij pas leeg is als de spanning is gezakt tot 2.0 Volt. Marge genoeg dus.

Een NiCd accu is niet echt aanbevolen, ofschoon het wel kan. Als eerste bezwaar geldt dat er op de print geen laadcircuit is voorzien, zodat de accu tijdens bedrijf niet geladen wordt. Een tweede bezwaar is, dat de DS1210 geen hogere spanningen dan 4 Volt op de batterij-ingang mag hebben. Dus als u een NiCd met laadcircuit wilt gebruiken, neem dan een NiCd van 3.6 Volt (dus 3 cellen) en overbrug de accu met een zenerdiode van 3.9 Volt. Dan blijft in ieder geval de DS1210 heel. Het laadcircuit bestaat uit een 1N4148 en een weerstand die in serie tussen de +5V en de plus van de accu worden geschakeld. De waarde van de weerstand wordt bepaald door de laadstroom. Nominaal valt er $5 - 0.6 - 3.6 = 0.8$ V over de weerstand. Bij een accu van 60 mAh is de laadstroom maximaal 6 mA, zodat de weerstand in dat geval minimaal 150 ohm moet zijn.

Een ander probleem blijkt de real-time clock te zijn. Sommigen rapporteerden een hangend systeem. Het vermoeden gaat op dit moment in twee richtingen:

- De interrupt van de real-time clock wordt in IO65 V2.14 niet goed afgehandeld;
- Een residente routine doet iets heel smerigs door direct een routine in IO65 aan te roepen.

Het onderzoek loopt nog. U hoort er nog van.

Adressering

In een van de vorige artikelen is keurig beweerd dat de nieuwe RAM-kaart opwaards compatibel zou zijn met de virtual diskkaart met DRAMs. De opletende lezer en bouwer heeft aan het vorige nummer kunnen zien dat dat eigenlijk niet klopt. Wat is het geval?

Op de versie met DRAMs ontbreekt het taaknummerregister. Het mapping-RAM bestaat uit twee 74S189's die geadresseerd worden op \$FFF0 tot en met \$FFFF. De SRAM-versie heeft op \$FFF0 het taaknummerregister, terwijl het mapping-RAM op

\$FFE0 tot en met \$FFEF te vinden is. Conclusie: niet zo heel erg compatibel. Aan het artikel in het vorige nummer over de virtual disk op de nieuwe kaart was dat dan ook te zien: alle adressen tussen \$FFF0 en \$FFFF moesten \$10 teruggefloten worden.

Nieuwe wegen?

Op de laatste bestuursvergadering is dit onderwerp kort ter sprake geweest. Het bestuur vond dat de ontwerper teruggefloten moest worden, en dat de adressering op de kaart terug moest naar de oude situatie, dus taaknummerregister op \$FFE0 en het mapping RAM op \$FFF0 tot en met \$FFFF. Dat levert de volgende zaken op:

- Nieuwe PAL. Helaas zijn de geleverde PALs niet herprogrammeerbaar.

- Nieuwe IO65. Deze zit in EPROM, kan dus wel worden overgeprogrammeerd.
- Nieuwe utilities. De bij de print gelverde diskette bevat het testprogramma VTEST en ook de in het vorige nummer gepresenteerde gewijzigde virtual disk. De meeste wijzigingen voor de virtual disk moeten dan weer ongedaan worden.

Omdat deze wijzigingen eigenlijk niet zoveel schade opleveren (ik hoop niemand de PAL direct in de print gesoldeerd heeft) is min

of meer besloten om de adressering om te draaien. De bestellers van de printer krijgen daarom binnkort een pakketje thuis met de volgende inhoud:

- Nieuwe PAL
- Nieuwe utility diskette
- Nieuwe IO65 in EPROM

Het pakketje zal kosteloos zijn.

Voor hen die niet kunnen wachten is in de figuur de nieuwe inhoud voor de PAL gegeven. De daar getoonde file is voor het pakket ABEL van Data I/O. De enige wijziging is dat waar !A4 stond nu A4 staat, en andersom. Samen met de gegevens uit het artikel uit het vorige nummer is de klus dan eventueel ook zonder externe steun te klaren.

De DOS65 werkgroep biedt alvast de excuses aan voor het ongemak. Maar we staan nu nog aan het begin, zodat de verandering met relatief weinig moeite doorgevoerd kan worden. Namens de werkgroep,

Nico de Vries

**Vroeger is keurig
beweerd dat de
nieuwe RAM-kaart
opwaards compati-
bel zou zijn.**

```

module _VDISK3B;                                flag '-r0';

title
'VDISK3B 5th VERSION                        N. DE VRIES  01-11-1994
TMM2064 CONTROLLER PAL FOR DOS65 VDISK CARD WITH SRAMS, 1M VERSION
NDV FIRMWARE';

"VDISK3B device 'P20V8S';
VDISK3B device 'P22V10';

"declarations

      H,L = 1,0;
      X,Z,C = .X.,.Z.,.C.;

"power pins
      GND      pin 12;
      VCC      pin 24;

"inputs
      RW      pin 1;   A12      pin 8;
      PHI2    pin 2;   A13      pin 9;
      A0      pin 3;   A14      pin 10;
      A1      pin 4;   A15      pin 11;
      A2      pin 5;   NC1      pin 13;
      A3      pin 6;   NC2      pin 14;
      A4      pin 7;   LS30     pin 23;

"outputs
      G245    pin 15;   Q2      pin 19;
      WE2064  pin 16;   Q1      pin 20;
      G573    pin 17;   Q0      pin 21;
      Q3      pin 18;   G138    pin 22;

equations

!G138      = (PHI2 & !A15                                "access
# PHI2 & !A14                                           "to
# PHI2 & !A13                                           "RAM
# PHI2 & A15 & A14 & !A13 );                          "array

!G245      = (!RW & A15 & A14 & A13 & A12 & !LS30 & A4  "write to $FFF0-$FFFF
# !RW & A15 & A14 & A13 & A12 & !LS30 & !A4           "write to $FFE0

```

Fig. 1: ABEL source file voor de nieuwe PAL

```

& !A3 & !A2 & !A1 & !A0
# !A13                                "access
# !A14                                "to
# !A15                                "RAM
# A15 & A14 & !A13 );                "array

!WE2064 = (PHI2 & !RW & A15 & A14 & A13 & A12 & !LS30
          & A4 );                    "write to $FFF0-$FFFF

G573 = (PHI2 & !RW & A15 & A14 & A13 & A12 & !LS30
        & !A4 & !A3 & !A2 & !A1 & !A0 ); "write to $FFE0

!Q0 = (!A0 & A15 & A14 & A13 & A12 & !LS30 & A4 "write $FFFX: A0 is TMM2064 address
      # !A12 ); "else let A12 through

!Q1 = (!A1 & A15 & A14 & A13 & A12 & !LS30 & A4 "write $FFFX: A1 is TMM2064 address
      # !A13 ); "else let A13 through

!Q2 = (!A2 & A15 & A14 & A13 & A12 & !LS30 & A4 "write $FFFX: A2 is TMM2064 address
      # !A14 ); "else let A14 through

!Q3 = (!A3 & A15 & A14 & A13 & A12 & !LS30 & A4 "write $FFFX: A3 is TMM2064 address
      # !A15 ); "else let A15 through

test_vectors ([PHI2,RW,A0,A1,A2,A3,A4,LS30,A12,A13,A14,A15]
              -> [WE2064,G138,G573,G245,Q0,Q1,Q2,Q3])

"                                     W
"                                     E
"P                                     L      2 G G G
"H                                     S A A A A 0 1 5 2
"I R A A A A A3 1 1 1 1 6 3 7 4 Q Q Q Q
"2 W 0 1 2 3 4 0 2 3 4 5 4 8 3 5 0 1 2 3
"-----
[H,H,H,H,H,H,H,L,H,H,H,H,H]->[H,H,L,H,H,H,H,H]; " 1 $FFFX, reading TMM2064
[L,H,H,H,H,H,H,L,H,H,H,H,H]->[H,H,L,H,H,H,H,H]; " 2 $FFFX, reading TMM2064
[H,H,L,L,L,L,H,L,H,H,H,H]->[H,H,L,H,L,L,L,L]; " 3 $FFFX, reading TMM2064
[L,H,L,L,L,L,H,L,H,H,H,H]->[H,H,L,H,L,L,L,L]; " 4 $FFFX, reading TMM2064
[H,L,L,H,L,H,H,L,H,H,H,H]->[L,H,L,L,L,H,L,H]; " 5 $FFFX, writing TMM2064
[L,L,L,H,L,H,H,L,H,H,H,H]->[H,H,L,L,L,H,L,H]; " 6 $FFFX, writing TMM2064
[H,L,H,L,H,L,H,L,H,H,H,H]->[L,H,L,L,H,L,H,L]; " 7 $FFFX, writing TMM2064
[L,L,H,L,H,L,H,L,H,H,H,H]->[H,H,L,L,H,L,H,L]; " 8 $FFFX, writing TMM2064
"

```

```

[H,H,L,L,L,L,L,H,H,H,H]->[H,H,L,H,H,H,H,H];      " 9 $FFE0, reading tasknumberreg.
[L,H,L,L,L,L,L,H,H,H,H]->[H,H,L,H,H,H,H,H];      "10 $FFE0, reading tasknumberreg.
[H,L,L,L,L,L,L,H,H,H,H]->[H,H,H,L,H,H,H,H,H];      "11 $FFE0, writing tasknumberreg.
[L,L,L,L,L,L,L,H,H,H,H]->[H,H,L,L,H,H,H,H,H];      "12 $FFE0, writing tasknumberreg.
"
[H,H,L,H,L,H,L,L,H,H,H,H]->[H,H,L,H,H,H,H,H,H];    "13 $FFEX, reading
[L,H,H,L,H,L,L,L,H,H,H,H]->[H,H,L,H,H,H,H,H,H];    "14 $FFEX, reading
[H,L,L,L,H,H,L,L,H,H,H,H]->[H,H,L,H,H,H,H,H,H];    "15 $FFEX, writing
[L,L,H,H,H,H,L,L,H,H,H,H]->[H,H,L,H,H,H,H,H,H];    "16 $FFEX, writing
"
[H,H,X,X,X,X,X,X,L,H,H,H]->[H,H,L,H,L,H,H,H,H];    "17 $EXXX, reading IO/video
[H,L,X,X,X,X,X,X,L,H,H,H]->[H,H,L,H,L,H,H,H,H];    "17 $EXXX, writing IO/video
[H,H,X,X,X,X,X,X,H,H,H,H]->[H,H,L,H,X,X,X,X,X];     "19 $FXXX, reading EPROM
[H,L,X,X,X,X,X,H,H,H,H,H]->[H,H,L,H,H,H,H,H,H];    "20 $FXXX, writing EPROM
"
[L,X,X,X,X,X,X,X,L,H,L,H]->[H,H,L,L,L,H,L,H,H];    "21 RAM: PHI2 low
[H,X,X,X,X,X,X,X,L,L,L,L]->[H,L,L,L,L,L,L,L,L];    "22 RAM: CPU address $0XXX
[H,X,X,X,X,X,X,X,H,L,L,L]->[H,L,L,L,H,L,L,L,L];    "23 RAM: CPU address $1XXX
[H,X,X,X,X,X,X,X,L,H,L,L]->[H,L,L,L,L,H,L,L,L];    "24 RAM: CPU address $2XXX
[H,X,X,X,X,X,X,X,H,H,L,L]->[H,L,L,L,H,H,L,L,L];    "25 RAM: CPU address $3XXX
[H,X,X,X,X,X,X,X,L,L,H,L]->[H,L,L,L,L,L,H,L,L];    "26 RAM: CPU address $4XXX
[H,X,X,X,X,X,X,X,H,L,H,L]->[H,L,L,L,H,L,H,L,L];    "27 RAM: CPU address $5XXX
[H,X,X,X,X,X,X,X,L,H,H,L]->[H,L,L,L,L,H,H,L,L];    "28 RAM: CPU address $6XXX
[H,X,X,X,X,X,X,X,H,H,H,L]->[H,L,L,L,H,H,H,L,L];    "29 RAM: CPU address $7XXX
[H,X,X,X,X,X,X,X,L,L,L,H]->[H,L,L,L,L,L,L,L,H];    "30 RAM: CPU address $8XXX
[H,X,X,X,X,X,X,X,H,L,L,H]->[H,L,L,L,H,L,L,L,H];    "31 RAM: CPU address $9XXX
[H,X,X,X,X,X,X,X,L,H,L,H]->[H,L,L,L,L,H,L,L,H];    "32 RAM: CPU address $AXXX
[H,X,X,X,X,X,X,X,H,H,L,H]->[H,L,L,L,H,H,L,L,H];    "33 RAM: CPU address $BXXX
[H,X,X,X,X,X,X,X,L,L,H,H]->[H,L,L,L,L,L,H,H,H];    "34 RAM: CPU address $CXXX
[H,X,X,X,X,X,X,X,H,L,H,H]->[H,L,L,L,H,L,H,H,H];    "35 RAM: CPU address $DXXX

"G245 is the enable for the databusbuffer
"
"G573 is the enable for the task number register
"
"G138 is the enable for the RAM array address decoder
"
"WE2064 is the write enable input for the TMM2064 and the enable for the
"   write data buffer to the TMM2064
"
"Q0..Q3 are the lower addresslines for the TMM2064

end _VDISK3B;

```

Linux, after install (deel 2)

Het doet me goed dat je na het installeren van Linux toch er meer mee wilt doen als alleen installeren. Dat er je er veel over moet lezen en nog meer mee moet werken zal echter altijd blijven.

Bij sommige operating systems moet men op de command line de extra opties of flags na de parameters opgeven, bij weer andere O.S.'s maakt het niet uit of eerst de optie of eerst de parameters komen. Bij Linux is het altijd **command [flag] [parameter [parameter]]**, het voorbeeld daarvan is **ls -l filename**. Ik onthoud de volgorde door de opties te beschouwen als een uitbreiding op het commando.

Voor het handiger gebruik van de shell is er de "tab-toets". Hij kan het commando of de filenaam verder voor je aanvullen. Type maar eens een **g** in vervolgens 1 keer op "tab": je hoort dan een piep. De "tab-hulp-functie" heeft dan gezien dat er verschillende mogelijkheden zijn, maar weet nog niet welke jij wenst. Door dan nog een keer op "tab" te drukken worden de verschillende mogelijkheden getoond. En de prompt verschijnt weer en ook de ingetypte letter "g". Als je dan op een toets drukt, bijvoorbeeld "a", welke het volgende karakter is van het commando en ook nogmaals op "tab", dan verschijnt er mij **gawk**, omdat de "tab-hulp-functie" dan wel weet welk commando je bedoelt.

Mtools

Voor het uitwisselen van informatie op floppy zijn MS-DOS formatted disks te gebruiken. Om zulke zaken te vergemakkelijken zijn er de "mtools". **man mtools** kan je al een heel eind verder helpen. Bij de kleine letters zal dan ergens staan dat er in **/etc** de file **mtools** moet staan. Een voorbeeld daarvan is te

zien in figuur 1. Merk op dat **fd0** twee keer gebruikt wordt.

Super user

/etc is trouwens de directory waar diverse bijzondere files in staan. De meeste files zijn echter van **root**. Om toch tijdelijk super user te worden, is er het commando **su**. Eventueel nog een wachtwoord invoeren en de prompt is veranderd in de prompt van **root**. Je wordt weer jezelf door **exit** of **bye** in te typen.

HOSTNAME/hosts

/etc herbergt ook de file **HOSTNAME**. Dat is dus de plaats waar je de naam van de machine die ook in de prompt wordt gebruikt kunt veranderen. Een andere file is **hosts**. Als deze rechtstreeks van Fred van Kempen komt, is hij behoorlijk leeg. In figuur 2 staat er meer informatie in. **hosts** wordt geraadpleegd als er verbindingen gemaakt worden met andere systemen. Wat ik nu aan het vertellen ben, is namelijk als voorbereiding voor de Linux-dag bedoeld.

/etc/rc.d/rc.inet1

Een andere noodzakelijke voorbereiding is dat je de file in figuur 3 hebt en dat je hem kunt aanpassen. Alle genoemde files

zullen trouwens ook op The Ultimate komen te staan. Zelf met de hand die files editten is misschien net zo makkelijk.

Kernel compileren / NET2

Wat ik deze keer NIET uitleg is het opnieuw compileren van je **kernel**. We gaan er namelijk van uit dat 26 november in je Linux systeem een netwerkkaart hebt zitten die door de **kernel** herkend wordt. Als dat niet het geval willen we daar ook wel naar kijken, maar eerst komen de mensen die we zou aan kunnen

De "tab-hulp-functie" heeft dan gezien dat er verschillende mogelijkheden zijn.

```
# a '/etc/mtools' file
A /dev/fd0H1440 12 80 2 18      # A: 3 1/2 High density
B /dev/fd0H720 12 80 2 9       # A: 3 1/2 Double density
C /dev/hda1 16 0 0 0
```

Fig. 1: het bestand MTOOLS

```

#
# hosts                                This file describes a number of hostname-to-address
#                                     mappings for the TCP/IP subsystem. It is mostly
#                                     used at boot time, when no name servers are running.
#                                     On small systems, this file can be used instead of a
#                                     "named" name server. Just add the names, addresses
#                                     and any aliases to this file...
#
# Version:                            @(#)/etc/hosts2.0004/30/93
#
# Author:                             Fred N. van Kempen, waltje@uwalt.nl.mugnet.org
#
# filled by Geert Stappers, 30/10/94
#

# For loopbacking.
127.0.0.1                            ds40.kgn.nllocalhost

# For TCP/IP
194.11.26.101 ds40.kgn.nl ds40
194.11.26.102 r2d2.kgn.nl r2d2
194.11.26.103 c3po.kgn.nl c3po
194.11.26.199 gmac.kgn.nl gmac
# End of hosts.

```

Fig. 2: het bestand HOSTS

sluiten. Ik ga er ook vanuit dat die mensen de NET2 software geïnstalleerd hebben.

Ping / ifconfig

ping hostname verstuurt pakketjes naar *hostname* met het verzoek deze weer terug te sturen. Als de pakketjes niet terug komen, (afbreken met control-C) weet je dat er wat fout is met de verbinding. Komen de pakketjes wel terug, dan wordt gerapporteerd hoelang ze onderweg zijn geweest. **ping** is een verdraaid makkelijk hulpmiddel om TCP/IP verbindingen elementair te testen. Een ander hulpmiddel is **ifconfig**, deze verschaft je informatie over het netwerk interface. Het is overigens een *root* commando.

Reboot

Als de instellingen niet goed blijken te zijn, dienen deze veranderd te worden. Vervolgens moet *root*

dan het commando **reboot** uitvoeren. Dat is een speciale vorm van *shutdown*, die na de "halt" een herstart teweeg brengt.

ftp

Als de installaties volbracht zijn, is het mogelijk om te "ftp-en". Je kunt thuis met 1 machine al wat testen. Na **ftp** een ? voor help. Wordt tijdens je test **pwd** beantwoord met **Not connected.**, dan moet je eerst een connectie maken. Dat doe je met **open eigen hostname** of met **open ipadres (aa.bb.cc.dd)**

RG58

Oh ja, zou je voor de aansluiting van 26 november zelf ook een stuk coaxkabel met een 50ohms impedantie, dus RG58, mee willen brengen?

Geert Stappers

```
#!/bin/sh
#
# rc.inet1                This shell script boots up the base INET system.
#
# Version:                @(#)/etc/rc.d/rc.inet11.0230/10/94
#

HOSTNAME='hostname'

# Attach the loopback device.
/etc/ifconfig lo 127.0.0.1
/etc/route add 127.0.0.1

# IF YOU HAVE AN ETHERNET CONNECTION, use these lines below to configure the
# eth0 interface. If you're only using loopback or SLIP, don't include the
# rest of the lines in this file.

IPADDR="194.11.26.101"
NETMASK="255.255.255.0"
NETWORK="194.11.26.0"
BROADCAST="194.11.26.255"
#GATEWAY="194.11.26.20"

# Uncomment ONLY ONE of the three lines below. If one doesn't work, try again.
# /etc/ifconfig eth0 ${IPADDR} netmask ${NETMASK} broadcast ${BROADCAST}
# /etc/ifconfig eth0 ${IPADDR} broadcast ${BROADCAST} netmask ${NETMASK}
# /etc/ifconfig eth0 ${IPADDR} netmask ${NETMASK}

# Uncomment these to set up your IP routing table.
/etc/route -n add ${NETWORK}
#/etc/route add default gw ${GATEWAY} metric 1

# End of rc.inet1
```

Fig. 3: de file RC.INET1

C++ de opvolger van C (deel 1)

Inleiding

Natuurlijk denkt u "Hé, moet Gert van Opbroek nu ook al een C-cursus schrijven?" Het antwoord hierop is een beetje Ja maar ook een beetje Nee. Zoals gebruikelijk komen we voor de μ P Kenner weer een hoop kopij tekort; dat is dus alleen al een reden wat nieuws aan te pakken. Bovendien wil ik zelf nu eindelijk eens C++ leren en tenslotte lijkt het me zeer nuttig verder te gaan waar Hans van Boheemen in zijn uitstekende serie is gestopt.

Deze cursus bestaat uit ongeveer 6 afleveringen en volgt zo'n beetje het spoor van een vergelijkbare cursus uit een Duits computerblad. Omdat ik zelf ook een beginner ben op het gebied van C++, zijn er in deze cursus ook een groot aantal persoonlijke ervaringen met de taal opgenomen (de leermeester ligt 14 dagen voor op de leerlingen, namelijk precies de tijd tussen dat ik dit schrijf en dat het blad bij u in de bus ligt). Ik ga er vanuit dat de lezer in ieder geval wat basiskennis van programmeren heeft en bij voorkeur in de taal C of in Pascal. Ik zal trachten de cursus zo goed mogelijk aan te laten sluiten bij de cursus van Hans van Boheemen zodat de cursussen samen een goede opleiding tot C++ programmeur zullen vormen.

Wat is C++ en hoe is het familie-verband

Zoals Hans al heeft aangegeven is de oerversie van C ontwikkeld uit de taal B. Het zou dus in de lijn ge-

egen hebben als de opvolger van C de naam D zou krijgen. Dat is niet gebeurd, op C heeft men de C operator voor opvolger losgelaten waardoor men de taal C++ kreeg.

In figuur 1 zijn vier versies van C++ als Venn-diagram getekend. Een Venn-diagram is een schematische voorstelling van een verzameling. We zien dat elke versie in het plaatje de vorige versie omvat. Hiermee wordt aangegeven dat de volgende versie upwards compatible is met de vorige versie. Dit betekent dat je een C-programma vrijwel zonder problemen door een C++ compiler kunt laten vertalen.

Kijken we even naar figuur 1, dan zien we in het hart de oerversie, het Kernigan en Ritchie C dat uit 1978 stamt. Op deze versie zijn in principe twee uitbreidingen gekomen. Dit is in de eerste plaats de versie die door de ANSI gestandaardiseerd is en die wordt aangeduid met ANSI C. Over deze versie is in 1988 door Kernigan en Ritchie een boek geschreven. De tweede uitbreiding noem ik Uitgebreid C; in het engels Extended C. In deze versie zitten een aantal uitbreidingen van het moderne C++, maar nog niet het zogenaamde Object Oriented deel. De laatste versie is het moderne C++ dat langzaam maar zeker een standaard-programmeertaal begint te worden.

C++ is oorspronkelijk ontwikkeld door B. Stroustrup en werd gepubliceerd in 1986. De taal is nog niet gestandaardiseerd, maar meestal wordt versie 3.0 van AT&T als referentie gebruikt. Een specificatie van C++ is wel in behandeling bij de ANSI om tot standaard te worden.

De belangrijkste uitbreiding van C++ ten opzichte van ANSI C is de mogelijkheid om Object geOrienteerd (OO dus of Object Oriented) programmatuur te ontwikkelen. Dit maakt C++ buitengewoon geschikt om voor de moderne operating systems (Windows!) en hulpmiddelen software te ontwikkelen.

Een leuke opmerking over C++ die ik bij de research voor dit artikel ben tegengekomen is dat C en dus ook C++ ontwikkeld is voor programmeurs door programmeurs. Hierbij werd rekening gehouden met het feit dat programmeurs over het algemeen te lui zijn om veel in te toetsen en dat

**Programmeurs
zijn over het algemeen
te lui om
veel in te toetsen.**

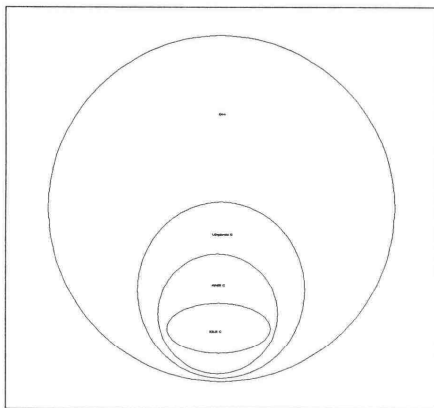


Fig. 1: van oerversie tot C++

1

asm	auto	break	case
char	class	const	continue
default	delete	do	double
else	enum	extern	float
for	friend	goto	if
inline	int	long	new
operator	overload	private	protected
public	register	return	short
signed	sizeof	static	struct
switch	template	this	typedef
union	unsigned	virtual	void
volatile	while		

Tabel 1: gereserveerde woorden in C++

daarom een groot aantal dingen heel compact in C++ geprogrammeerd kunnen worden. Uiteraard heeft dit als nadeel dat voor niet-programmeurs (of minder ervaren programmeurs) C++ maar moeilijk te lezen is.

De eerste verschillen

Voordat ik ook maar één voorbeeld in C++ kan laten zien, moet ik het eerste verschil tussen C en C++ al vertellen. Commentaar in C wordt aangegeven d.m.v. de tekens /* (start commentaar) en */ (einde commentaar). In C++ bestaan deze tekens ook (compatibiliteit!) maar er is nog een tweede commentaarteken bijgekomen die zeer handig is. Dit is het teken //. // betekent dat alles vanaf dit teken tot het einde van de regel opgevat moet worden als commentaar, dus zo'n beetje hetzelfde als de ; in assemblers. Omdat veel C compilers ook (een aantal van) de uitbreidingen van C++ kennen, is dit commentaarteken meestal ook in C programma's wel toegestaan; dit is echter niet volgens de ANSI standaard.

Als een taal groter wordt, dan neemt over het algemeen ook het aantal gereserveerde woorden toe. In tabel 1 staat een overzicht van alle gereserveerde

woorden in C++ waarbij de woorden die niet in ANSI C zitten vet zijn afgedrukt. Gereserveerde woorden zijn woorden die onderdeel zijn van de taal. Dit betekent dat deze woorden niet als naam van bijvoorbeeld een variabele gebruikt mogen worden.

Omdat C een case-sensitive taal is, moeten woorden altijd op precies dezelfde wijze, met hoofd- en kleine letters, geschreven worden. Merk op dat alle gereserveerde woorden met kleine letters worden geschreven. Verder is de lijst niet compleet. Elke C++ compiler heeft nog een aantal extra gereserveerde woorden die specifiek voor die implementatie van C++ zijn. Mocht je problemen hebben met je compiler kijk dan in de documentatie of je misschien een gereserveerd woord als naam van een variabele hebt gebruikt.

Naast sleutelwoorden, kent de taal ook een aantal operatoren. In tabel 2 zijn deze afgedrukt waarbij de nieuwe operatoren weer vet staan vermeld.

Hoewel C++ een aantal verschillen kent met C, zijn de twee versies toch door elkaar te gebruiken. Je kunt dus van een bestaand systeem de oude in C ge-

[]	()	.	.*	->	->*
++	--	&	*	+	-
~	!	sizeof	/	%	<<
>>	<	>	<=	>=	=
!=	^		&&		?:
=	*=	/=	%=	+=	-=
<<=	>>=	&=	=	,	::
#	##				

Tabel 2: operators in C++

programmeerde delen blijven gebruiken en de nieuwe delen in C++ schrijven. Afgesproken is dat als de source extensie .cpp heeft, de source door de C++ compiler wordt vertaald en als de source extensie .c heeft de ANSI C compiler gebruikt wordt. Wil je nu in een C++ source een in C geprogrammeerde functie gebruiken, dan moet je dit in de C++ source aangegeven. Ik zal nu eerst een voorbeeld geven hoe je functies uit andere source files aanroept in ANSI C. Vervolgens laat ik zien hoe je vanuit een C++ source een C functie aanroept.

Meerdere source files

Stel je hebt een source file bollen.c met daarin de volgende inhoud:

```
float opperv(float strl)
{
    return (4 * 3.14 * strl * strl);
}
```

en uiteraard het hoofdprogramma in een andere source file:

```
#include <stdio.h>
extern float opperv(float);

void main()
{ float straal;
  printf("Geef de straal van een bol: ");
  scanf("%f",&straal);
  printf("De oppervlakte van de bol is %.2f",
        opperv(straal));
}
```

Dit is hetzelfde voorbeeld als in aflevering 6 van de C-cursus van Hans van Boheemen, met dit verschil dat de functie in een andere source file zit. Als je de C-cursus er bij pakt, dan zie dat er in de declaratie van opperv iets is veranderd. In de declaratie staat nu dat de functie niet in deze source file staat maar "ergens anders". Je moet nu aan de linker niet één source file opgeven maar twee, de source file met de functie main() en de source file met de functie opperv(). Hoe je dat doet, is afhankelijk van het systeem waarop je werkt en het type compiler. Op mijn systeem (MS-DOS met Turbo C++), doe ik het door een Project te definiëren en daar de beide sources in op te nemen. Onder LINUX kun je de files beide aan de Linker opgeven, maar je kunt beter een makefile maken en de opdracht "make" gebruiken. Misschien kan een LINUX gebruiker eens een artikelje schrijven hoe je make gebruikt?

Het is gebruikelijk voor een groot programma een heleboel aparte source files aan te maken en die dan vervolgens tot één programma samen te bouwen. Als

je dan functies hebt die in meerdere sources gebruikt worden, dan is het niet echt handig dat je iedere keer weer alle declaraties op moet nemen in de source. In plaats daarvan kun je ook de declaraties in een aparte file zetten die je de extensie .h geeft. Dit wordt dan een header file genoemd en die wordt m.b.v. de #include opdracht in de source file(s) ingelezen. Neem maar weer hetzelfde voorbeeld:

1) De header file bollen.h bevat de volgende regels:

```
extern float opperv(float);
extern float inh(float);
```

2) De source file bollen.c bevat de functies:

```
float opperv(float strl)
{
    return (4 * 3.14 * strl * strl);
}

float inh(float strl)
{
    return (4 / 3.0 * 3.14 * strl * strl);
}
```

3) De source file main_b.c met de regels:

```
#include <stdio.h>
#include "bollen.h"

void main()
{ float straal;
  printf("Geef de straal van een bol: ");
  scanf("%f",&straal);
  printf("De oppervlakte van de bol is %.2f",
        opperv(straal));
  printf("\t en de inhoud %.2f\n",
        inh(straal));
}
```

Merk op dat de twee #include opdrachten verschillende tekens gebruiken waartussen de header file staat. Dat is geen toeval! Met de tekens om stdio.h heen wordt bedoeld dat we een header file zoeken die onderdeel is van de compiler en op een daarvoor gereserveerde directory staat. Met de tekens om bollen.h heen wordt bedoeld de actuele directory.

Zo, laten we nu het hoofdprogramma eens in C++ gaan schrijven waarbij we de functies in C laten staan. Je moet nu de file de extensie .cpp geven. Verder moet je aangeven dat je C functies wilt gebruiken. Dat kan op twee manieren. Je kunt de declaratie schrijven als:

```
extern "C" opperv(float);
```

Of je schrijft:

```
extern "C"
{
#include "bollen.h"
}
```

NB: de hash (#) moet altijd in de eerste positie op de regel staan. Dus ondanks het feit dat de regel tussen accolades staat, mag je niet inspringen. Verder mogen er tussen de accolades meerdere header files staan mits die natuurlijk verwijzen naar in C geprogrammeerde functies.

Het hoofdprogramma in C++ ziet er nu als volgt uit:

```
#include <iostream.h>
#include <iomanip.h>

extern "C"
{
#include "bollen.h"
}

void main ()
{ float straal;
  cout << "\ngeef de straal van de bol :";
  cin >> straal;
  cout << setiosflags(ios::fixed | ios::showpoint)
        << setprecision(2)
        << "De oppervlakte van de bol is: "
        << opperv(straal)
        << "\ten de inhoud " << inh(straal);
}
```

We zien dat er ook geen gebruik meer gemaakt is van `scanf` en `printf`. In deze versie wordt het zogenaamde "stream io" gebruikt. Dit is een uitbreiding die er in Extended C bij is gekomen. Hoewel ook het gewone standaard io (`scanf`, `printf`) gebruikt kan worden, wordt in C++ over het algemeen de stream io gebruikt. Dit lijkt op het eerste gezicht wat omslachtiger, maar later zal blijken dat er toch ook grote voordelen aan deze manier van werken zitten. Ik kom daar in aflevering 2 of 3 nog op terug.

Datatypes

Elke taal heeft een aantal standaard datatypes. Ook C++. In tabel 3 zijn deze opgesomd.

Uit het overzicht blijkt dat de lengte van een int of een short int niet vast ligt. Deze lengte is machine en/of compiler afhankelijk. Verder ligt ook de lengte van een pointer (*) niet vast. Ook deze is machine afhankelijk en wordt in principe bepaald door de

Datatype	Breedte (bit)
char	8
unsigned char	8
int	16 of 32
unsigned int	16 of 32
long int	32
unsigned long int	32
short int	8 of 16
unsigned short int	8 of 16
float	32
double	64
long double	80
*	16 of 32
void	-

Tabel 3: datatypes in C++

breedte van de adresbus of de geheugenruimte die je in het programma wilt kunnen gebruiken.

Het is toegestaan het sleutelwoord `int` weg te laten als dit vooraf wordt gegaan door `unsigned`, `short` of `long`. Verder geldt dat bij berekeningen met float getallen de berekening altijd uitgevoerd wordt in de nauwkeurigheid van double waarna het resultaat in float wordt omgezet. Dit heeft als gevolg dat berekeningen waarin alleen double voorkomt en met een double resultaat sneller uitgevoerd worden dan een berekening met float's.

Een * geeft een pointer aan. Dit datatype is niet door Hans behandeld. We zullen hem echter wel in deze serie tegen gaan komen. Een pointer is niets meer of minder dan een adres. Als we een pointer naar een int hebben, dan betekent dit dat we een variabele hebben met daarin het adres van een int. In C noteer je dat als volgt:

```
int *p,a;

p = &a;
*p = 17;
```

dit betekent dat we een variabele `a` van het type `int` gedefinieerd hebben en een pointer naar een `int`. Vervolgens hebben we, met de `&` operator, in de pointer `p` het adres van `a` geschreven en daarna via de pointer de waarde van `a` op 17 wordt gezet. Pointers zijn de basis van C en C++ en we zullen daar ook nog uitgebreid op terugkomen.

Het void datatype betekent eigenlijk "niets". C++ kent alleen functies, geen procedures of subroutines. Dat wil zeggen dat een functie altijd een resultaat terug moet geven. Mensen die wel eens in Pascal hebben gewerkt, weten dat je daar functies hebt en procedures. Een functie geeft een antwoord terug, een procedure niet. Welnu, door in C++ het datatype void te gebruiken, geef je aan dat je niets terug wilt geven. Aangezien zelfs main() een functie is, moet je om geen waarschuwingen van de compiler te krijgen ook voor main() een datatype, bijvoorbeeld het datatype void neer zetten.

Een bijzonder geval waarin void wordt gebruikt, is bij pointers. Door een pointer naar een void te declareren, krijg je een pointer die naar "niets" wijst. Dit betekent dat je een adres hebt zonder dat je je druk maakt over het datatype op dit adres. Een zeer bekend voorbeeld is de functie malloc met behulp waarvan een stuk geheugen gereserveerd kan worden. Deze functie heeft de volgende declaratie:

```
void *malloc(size_t num)
```

De functie levert een pointer naar een blok vrij geheugen met een afmeting van num byte. Door middel van een typecast operator kun je vervolgens aangeven naar welk datatype de pointer verwijst:

```
char *p;
```

```
p = (char *) malloc(1024);
```

Zo krijgen we een pointer naar een buffer van 1024 byte van het type char. De operator met de haakjes is de typecast operator. Deze wil zeggen "Maak van het resultaat een pointer naar een char".

Tenslotte nog een opmerking over de sleutelwoorden const en volatile. Deze sleutelwoorden worden voor de declaratie gezet. Het sleutelwoord const betekent dat de betreffende variabele niet gewijzigd mag worden bijvoorbeeld:

```
const int drie = 3;
```

In C++ zijn constanten echte constanten; dit in tegenstelling tot C waar een constante een geheugenplaats is die vooraf gevuld werd. In C++ kun je constanten ook als volgt gebruiken:

```
const int lengte = 25;
char a[lengte];
```

Waarbij je dus de lengte van een array als een constante gedefinieerd hebt. In C lukt je dit alleen met een macro:

```
#define lengte 25
char a[lengte];
```

Het sleutelwoord volatile betekent het tegenovergestelde. Volatile wil zeggen dat de inhoud van de variabele spontaan kan wijzigen bijvoorbeeld omdat dit het register van een klok-ic is. Het is nodig om de compiler dit te vertellen omdat hij anders wel kan besluiten om in een lange berekening de variabele in een register te kopiëren. D.m.v. van volatile dwing je de compiler de variabele elke keer als hij nodig is opnieuw uit het geheugen op te halen.

Plaats van de declaraties

In C moeten declaraties altijd aan het begin van een blok, bijvoorbeeld een functie staan. Je krijgt zo in een blok een deel met declaraties, gevolgd door een deel met statements. In C++ mogen declaraties overal staan. Dit houdt in dat de volgende constructie correct C++ is:

```
for (int i = 0; i < 99; i++)
{
    .....
}
```

De variabele i wordt in dat geval gedeclareerd op het moment dat hij voor het eerst nodig is. Je hebt dan een beter overzicht van welk type een variabele is. Hoewel de variabele i in het for statement gedeclareerd wordt, blijft hij toch tot het eind van de functie bestaan.

Afsluiting

In deze aflevering zijn we voorzichtig begonnen met wat te ruiken aan C++. In de volgende aflevering ga ik het hebben over verschillende manieren van parameter-overdracht naar functies. Bovendien zal ik dan ook ingaan op het verschil tussen macro's, functies en inline functies. Ook pointers zullen in het volgende artikel uitgebreid aan bod komen. Tot de volgende keer.

Gert van Opbroek

Referenties:

- 1: Leen Ammeraal: C++ 2^e herziene druk; Academic Service (1991)
- 2: Turbo C++ User's Guide; Borland 1991
- 3: Turbo C++ Programmer's Guide; Borland 1991
- 4: Heiner Högel: C plus einhalb, Umstieg von C auf C++; DOS International 7/94
- 5: Dirk Schaun: C++ - eine Sprache mit Know-how; DOS International 8/94
- 6: Hans van Boheemen: Cursus C; µP Kenner 81 t/m 86

Voortgang KGN68k (deel 17)

Op het moment van schrijven is het nog maar 7 weken voor de bijeenkomst van november 1994, de "opleverdatum" van KGN68k. Het is met opzet dat "opleverdatum" tussen aanhalingstekens staat, want dat is de datum dat de machine uitgeleverd moet kunnen worden.

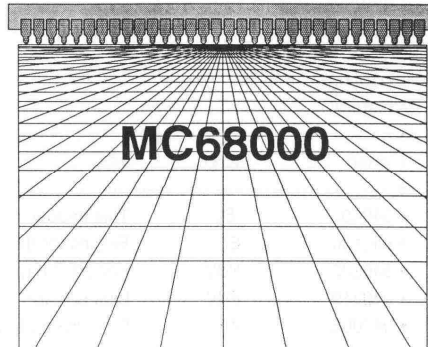
Hardware

Even is er paniek geweest. Eerst dachten we dat de problemen met de busy-lijn op de SCSI-bus in software op te lossen waren. De SCSI-controller chip (53C80) zelf bleek kapot, vermoedelijke oorzaak is de gebruikte harddisk of te dunne ground draden. Bestelling bij dezelfde plaats bleek niet meer mogelijk. De second source fabrikant had de chip obsoleete verklaard! Toen zaten we dus met de handen in het haar. Via via zijn we toen aan het adres van NCR Europe gekomen. Van de oorspronkelijke fabrikant hebben we nu een sample gekregen. Met die chip draait de KGN68k nu, maar wel met dikkere ground lijnen en die twijfelachtige harddisk is er nog niet aan geweest.

Dat het programmeren van de PI/T timer niet goed ging was te wijten aan het feit dat de 68230 een conflict had met een buffer aan Centronics-zijde die de verkeerde richting in stond.

Een probleem dat maar sporadisch voorkwam, en ik daarom voor jullie verzwegen heb, is ook verholpen. De Smartwatch die we gebruiken had er toch moeite mee dat de SRAM met 125 nanoseconde access time werd aangesproken, met een toegangstijd van 187,5 ns (3 periodes van 16MHz frequentie) blijven de "spontane" buserrors achterwege.

Ondertussen heeft de KGN68k ook een NMI circuit. Met een schakelaar is die schakeling te activeren. NMI is een interrupt die altijd doorkomt. Je kunt dan de interrupt routine dan diverse gegevens over het programma op dat moment laten vertellen. Vooral voor de programmeur is dat handig. Hetzelfde circuit kan ook een powerfailure detecteren, je hebt dan nog net even tijd om het write commando naar je diskdrives terug te fluiten. Ook nieuw is een LED is gaat branden als de processor in "fail", te vergelijken met "Halt", terecht komt. Het nut ervan is dat je kunt zien of de processor in software of hardware "hangt".



Je hebt dan nog net even tijd om het write commando naar je diskdrives terug te fluiten.

Dat er zoveel is gebeurd is aan Ad Brouwer te danken. Ad zelf was liever niet bij dat verkeersongeval betrokken, dan had hij aan zijn nieuwe huis kunnen werken, van de zomer eventueel nog kunnen wielrennen en van de winter kunnen schaatsen. Maar omdat hij nu een paar maanden niet kan lopen, had hij wat meer tijd voor de KGN68k. Bij mijn bezoeken aan Ad met een intervaltijd van ongeveer een week heb ik de KGN68k met sprongen zien verbeteren.

Software

Ook daar is het een en ander gebeurd. De grootste verandering heeft plaatsgevonden in de (boot-)ROM. Deze is nu zo goed als helemaal in C geschreven. Daar waar het echt niet in C kan, (zie "entries.asm") wordt gebruik gemaakt van de KGN68k assembler. De vorige versie van de (boot-)ROM was aan elkaar geplakt met diverse assembly routines. Hij werkte wel, maar nieuwe zaken toevoegen, zoals breakpoint routines, zou een hels karwei worden. De nieuwe ROM is echt 'geconstrueerd' met C-functies. Er zijn zelfs voorzieningen aanwezig om nieuwe monitorcommando's te kunnen downloaden. Tijdens het testen van deze nieuwe commando's wordt al gebruik gemaakt van monitor routines zoals "getline" & "tohex". Hebben deze routines de test goed doorstaan dan kunnen ze bij de volgende versie in ROM gezet worden.

```

* entries.asm
* some entry points in rom
* authors : Geert Stappers
*          Pieter de Visser
*          Ad Brouwer
* copyright : K.G.N.
* history:
*
* when      who      why
* -----
* 940102     St       This header
* 940130     St       Entries for disasm, rtcget & rtcset
* 940222     PdV      Added 'bin gnu' - and really proud of it!
* 940319     PdV      Last update
* 941006     AB       Connection to monit.c
*
*          bin       gnu
*          cpu       030
*
* external functions
*
*          global    _monitor
*          global    _start
*          global    _getchar
*          global    _putchar
*          global    _chkchar
*          global    _putline
*          global    _sectorread
*          global    _sectorwrite
*          global    _rtcget
*          global    _rtcset
*          global    _setxentry
*          global    _excreturn
*
* functions to be exported
*
*          global    _doreturn
*          global    _doreset
*          global    _setsp
*          global    _setsr
*          global    _setvbr
*          global    _getcacr
*          global    _setcacr
*          global    _geta6
*          global    _seta6

```

Fig. 1: ENTRIES.ASM

	global	_getusp	
	global	_setusp	
	global	_disableint	
	global	_saveregs	
	global	_retfromexc	
	jmp	_monitor	; warm start monitor
	jmp	_start	; cold start monitor
	jmp	_getchar	
	jmp	_putchar	
	jmp	_chkchar	
	jmp	_putline	
	jmp	_sectorread	
	jmp	_sectorwrite	
	jmp	_rtcget	
	jmp	_rtcset	
	jmp	_setxentry	
_doreturn			
	jmp	_execreturn	
_doreset			
	reset		; activate reset by cpu
	rts		
_setsp			
	move.l	(sp),a0	; return address
	move.l	4(sp),sp	; new stack top
	move.l	sp,a6	; link a6 to top of stack
	subq.w	#4,sp	; at return argument is removed
	jmp	(a0)	
_setsr			
	move.w	6(sp),sr	; set status register
	rts		
_setvbr			
	move.l	4(sp),a0	
	movec	a0,vbr	; set vector base register
	rts		
_getcacr			
	movec	cacr,d0	; get cache control register
	rts		
_setcacr			
	move.l	4(sp),d0	
	movec	d0,cacr	; set cache control register
	rts		
_geta6			
	move.l	a6,d0	; get a6
	rts		
_seta6			
	move.l	4(sp),a6	; set a6
	rts		

_getusp	move.l	usp,a0	
	move.l	a0,d0	; get usp
	rts		
_setusp	move.l	4(sp),a0	
	move.l	a0,usp	; set usp
	rts		
_disableint	or.w	#\$0700,sr	; disable all interrupts
	rts		
_saveregs	move.l	a0,-(sp)	
	move.l	8(sp),a0	; d[0..7], a[0..7]
	movem.l	d0-d7/a0-a7,(a0)	
	move.l	(sp),32(a0)	; a0 saved on top of stack
	addq.l	#4,sp	
	rts		
_retfromexc	move.l	4(sp),a6	; restore registers
	movem.l	(a6),d0-d7/a0-a5	
	move.l	56(a6),a6	; restore a6
	move.l	8(sp),a7	; position sp to stackframe
	rte		
	end		

Voor de disassembler is het verzoek gekomen om ook de eventuele ASCII tekens af te drukken. Eigenlijk was hij af, maar door deze nuttige verfraaiing, duurt het toch weer even voordat hij compleet is.

Bij het porteren van Linux/68k gaan we nu uit van versie 09 patchlevel 3. Versie 09 houdt in dat dezelfde Kernel ook op Atari's draait met een 68030. Momenteel wordt er door ons aan de SCSI device driver gewerkt. Het SCSI stuk is bij Linux in drie lagen opgebouwd. In de bovenste laag worden hard-disk, CD-ROM & tapestreamers behandeld. De middelste is specifiek SCSI, zoals readblock & release bus. De onderste laag is pas hardware-afhankelijk.

Printed Circuit Board

"In het PCB ontwerp zit nauwelijks Voortgang."

In het PCB ontwerp zit nauwelijks Voortgang. En dat is het punt waar nu het meeste moet gebeuren. Dit blijkt echter moeilijk te zijn. Ik heb het lay-outen op mij genomen, toen mijn voorganger zijn vrije tijd elders nodig had. Een volledige baan en Projectleider KGN68k en voorzitter KGN en zelf leven i.p.v. geleefd worden is toch wat veel. Vandaar dat je elders in De μ P Kenner een oproep vindt voor een PCB-layouter.

Marketing

Op het terrein van de-machine-op-de-markt-bren-gen gebeurt er zelfs NIETS. Het enige voorbereide werk wat er nu gebeurt is dat we een plaats op de HCC-dagen hebben. En natuurlijk een werkend proto-type. Maar bijdrages zijn toch nog steeds welkom.

Geert Stappers

Balans per 31 december 1993

De kascontrolecommissie heeft zijn werk gedaan. Dus dit had u nog tegoed.

AKTIVA	1993	1992
(Inventaris)		2.086,35
Vorraden	pm	pm
Te ontvangen posten	0,00	0,00
Geldmiddelen	5.778,96	7.978,59
	-----,-	-----,-
TOTAAL	5.778,96	10.064,94

PASSIVA	1993	1992
Vrije reserve	5.778,96	9.805,86
(vooruit ontvangen contributie)	0,00	195,00
Te betalen posten	0,00	64,08
	-----,-	-----,-
TOTAAL	5.778,96	10.064,94

VRIJE RESERVE	1993	1992
Stand per 1 januari	7.978,59	12.039,50
Toevoeging batig saldo verdeling	-2.199,63	-2.233,64
	-----,-	-----,-
Totaal per 31 december	5.778,96	9.805,86

TE BETALEN POSTEN (in volgend boekjaar)	1993	1992
Redactie	0,00	39,08
Bestuurskosten	0,00	25,00
	-----,-	-----,-
Totaal per 31 december	0,00	64,08

Opm.: Vrije reserve is door de jaren heen onterecht overgeboekt nl. met inventaris. Dit gaat eruit om een goed beeld van de vrije reserve te krijgen.

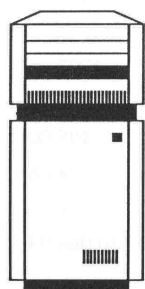
Kascommissie controle,

Voor accoord,

H. Speksnijder en R. Vleesch-Dubois

The Ultimate

The BBS for all systems



Telefoon:
053-303902, 053-328506 of
053-327457

- 053-303902 (2 lijnen!) -
V22, V22bis, V23, V32bis, HST/14k4, V42bis, MNP5

- 053-328506 -
V21, V22, V22bis, V23, V32bis, V.Fast (28k8),
V42bis, MNP5

- 053-327457 -
V21, V22, V22bis, V23, V32bis, V.Fast (28k8),
V42bis, MNP5

Ik heb interesse in de KGN en wil

☐ Lid worden van de KGN

☐ Meer informatie over de KGN

Naam : _____

Adres : _____

Postcode en Woonplaats : _____

Datum : _____ Handtekening : _____

Dit strookje kunt u ingevuld opsturen aan het secretariaat van: **KIM Gebruikersclub Nederland**
Postbus 1336
7500 BH Enschede

Van de voorzitter

In het vorige nummer is er gesproken over eventuele deelname aan de HCC-dagen. De prijs was toch nog omhoog te krijgen. En nu kan ik jullie mededelen dat we er dit jaar dus wel staan. Dan kunnen we te minste aan de buitenwereld laten zien dat we bestaan.

Het weekend er na is er een bijeenkomst/ledenvergadering van de KGN in Amersfoort. De ochtend is bedoeld voor de vergadering en de middag voor Linux. Een van de agenda punten is de "oplevering" van KGN68k. De ledenvergadering van 1993 vond het goed dat we nog een jaar doorgingen met KGN68k, mits we staande de ledenvergadering 1994 maar klaar waren voor uitlevering. Dat de printplaat layout niet op tijd af zullen zijn is wel jammer.

Een ander agendapunt is de verkiezing van een nieuw bestuur. Vooral nieuwe bestuursleden zijn hard nodig. Het gaat er naar uit zien dat het nieuwe jaar gaat beginnen met maar drie bestuursleden. Ik vraag aan jullie om **niet met lede ogen aan te zien** of die drie het voor elkaar kzullen rijgen. Help

alsjeblieft mee de technische computerclub te laten bestaan.

De doelstellingen van de KGN mogen duidelijk zijn. De concrete invulling kan gebeuren door bezig te zijn met operating systemen zoals OS/2 & Linux. KGN68k is een mooi voorbeeld van onze doelstellingen. Dat er bescheiden mensen met microcontrollers bezig zijn is ook prachtig. Liever zou ik zien dat die mensen wat vertellen/schrijven over hun technische computer hobby.

Het TCP/IP gebeuren op de KGN/Linux dag in Amersfoort geeft inhoud aan het datacommunicatie gedeelte.

We weten als bestuur welke kant we op willen, waarschijnlijk willen jullie ook die kant op. Maar zijn er mensen die samen met het bestuur voor op willen lopen?

Jullie Voorzitter,

Geert Stappers

KGN promotie team

Jullie als lid van de KGN mogen deel uitmaken van het KGN promotie team.

Het is gelukkig geen full-time job, want we hebben het waarschijnlijk toch wel druk genoeg. We zoeken een groep mensen die bereid zijn eventueel een stand te bemannen tijdens een beurs of ander evenement. Als je nu ja zegt betekent dat niet dat je straks nog aan die verplichting vastzit. Pas als de KGN naar een namelijk in het aansluiten van de te demonstreren computer(toepassing). Tijdens de beurs zelf promotiemateriaal uitdelen, eventueel wat vertellen. In overleg met andere KGN promotie team leden wie er bij de stand blijft en wie er even over de beurs wandelt. Aan het einde van de dag de zaken weer netjes afbreken.

Hoe wordt je KGN promotie team lid?

Een briefje, bericht of telefoontje naar Geert Stappers!

Informatie

De μ P Kenner (De microprocessor Kenner) is een uitgave van de KIM gebruikersclub Nederland. Deze vereniging is volledig onafhankelijk, is statutair opgericht op 22 juni 1978 en ingeschreven bij de Kamer van Koophandel en Fabrieken voor Hollands Noorderkwartier te Alkmaar, onder nummer 634305. Het gironummer van de vereniging is 3757649.

De doelstellingen van de vereniging zijn sinds 1 januari 1989 als volgt geformuleerd:

- Het vergaren en verspreiden van kennis over componenten van microcomputers, de microcomputers zelf en de bijbehorende systeemsoftware.
- Het stimuleren en ondersteunen van het gebruik van micro-computers in de meer technische toepassingen.

Om deze doelstellingen zo goed mogelijk in te vullen, wordt onder andere 5 maal per jaar de μ P Kenner uitgegeven. Verder worden er door het bestuur per jaar 5 landelijke bijeenkomsten georganiseerd, beheert het bestuur een Bulletin Board en wordt er een softwarebibliotheek en een technisch forum voor de diverse systemen in stand gehouden.

Landelijke bijeenkomsten:

Deze worden gehouden op bij voorkeur de derde zaterdag van de maanden januari, maart, mei, september en november. De exacte plaats en datum worden steeds in de μ P Kenner bekend gemaakt in de rubriek Uitnodiging.

Bulletin Board:

Voor het uitwisselen van mededelingen, het stellen en beantwoorden van vragen en de verspreiding van software wordt er voor de vereniging een Bulletin Board (BBS) beschikbaar gesteld. De telefoonnummers zijn: 053-328506, 053-303902 of 053-327457. Berichten sturen (Mail) via Internet naar het BBS is tegenwoordig ook mogelijk. Het Internet adres van het BBS is: ultima.iaf.nl

Software Bibliotheek en Technisch Forum:

Voor het beheer van de Software Bibliotheek en technische ondersteuning streeft het bestuur ernaar zgn. systeemcoördinatoren te benoemen. Van tijd tot tijd zal in de μ P Kenner een overzicht gepubliceerd worden. Dit overzicht staat ook op het BBS.

Correspondentie adres

Alle correspondentie betreffende verenigingszaken kan gestuurd worden aan:

KIM Gebruikersclub Nederland
Postbus 1336
7500 BH Enschede

Het Bestuur

Het bestuur van de vereniging wordt gevormd door een dagelijks bestuur bestaande uit een voorzitter, een secretaris en een penningmeester en een viertal gewone leden.

Geert Stappers (Voorzitter, KGN/68k coördinator)
Engelseweg 7
5825 BT Overloon
Telefoon 04781-41279
Internet: stappers@knoware.nl

Jacques H.G.M. Banser (penningmeester)
Haaksbergerstraat 199
7513 EM Enschede
Telefoon 053-324137
Internet: jacques@ultima.iaf.nl

Gert van Opbroek (secretaris)
Den Del 16
5071 TT Udenhout
Telefoon 04241-3795

Jan Veninga
Klimopstraat 51
7601 SJ Almelo
Telefoon 0546-827910

Henk Speksnijder
Albert Cuijpsstraat 43
2902 GA Capelle aan den IJssel
Telefoon 010-4586879

Nico de Vries (redactie μ P Kenner)
Van der Waalsstraat 46
2984 EP Ridderkerk
Telefoon 01804-29207

Ereleden:

Naast het bestuur zijn er een aantal ereleden, die zich in het verleden bijzonder verdienstelijk voor de club hebben gemaakt:

Erevoorzitter:
Siep de Vries

Ereleden:
Mevr. H. de Vries-van der Winden
Anton Müller
Rinus Vleesch Dubois