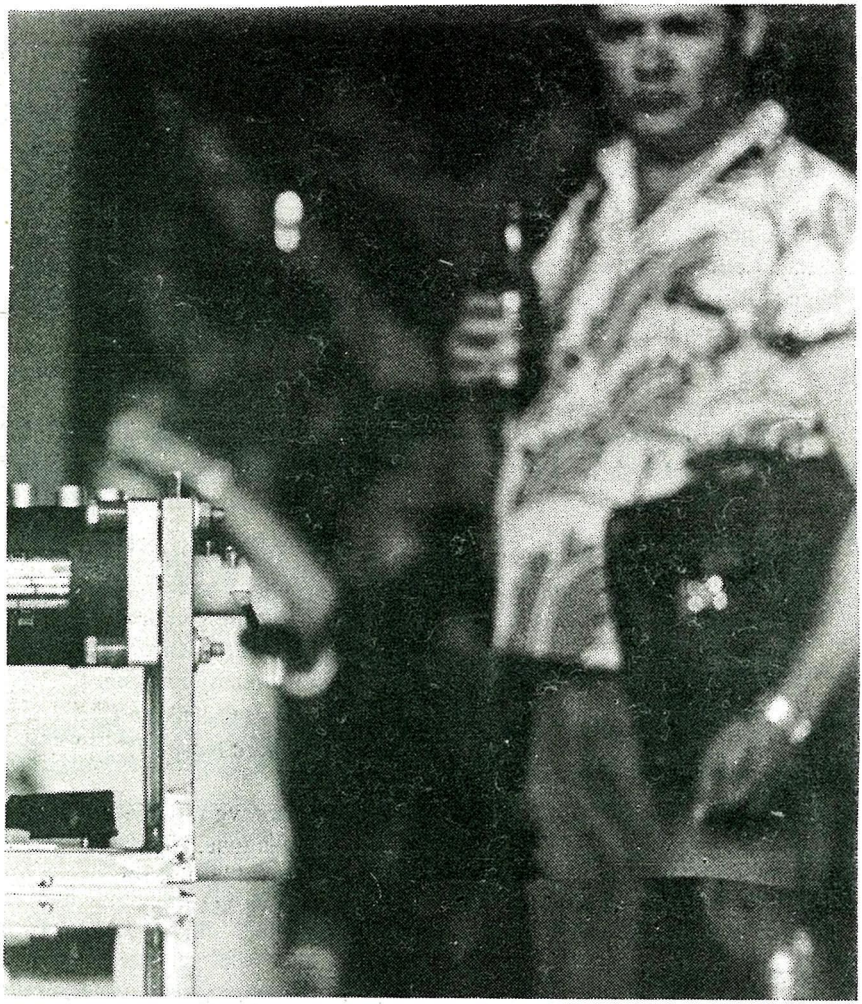
StackPointer

4-1988. Organ för Datorföreningen Stacken, KTH.



PDP – en makalös manick att knäcka julnötterna med
– julklappen till den som trodde han hade allt – sid 14.

STACKPOINTER

STACKPOINTER är organ för Datorföreningen STACKEN på KTH. Vi tänkte försöka oss på en mer regelbunden utgivning under 1989: Manusstopp och redaktionsmöte helgen efter månadsmöte i jämn månad. Citera gärna, men kom ihåg att ange källan och tillskicka gärna föreningen ett exemplar.

Redaktör: Hans Nordström.
I redaktionen: Jan Michael Rynning, Mats O Jansson och Stellan Lagerström.
Foto: Rickard Andersson.
Ansvarig utgivare: Mats O Jansson.
Färdigställd: 5:e december 1988.
Nästa manusstopp och redaktionsmöte: 4:e–5:e februari 1989.

Datorföreningen STACKEN

POST till föreningen skickas till nedanstående adress eller läggs i postfacket på NADA.

Datorföreningen STACKEN
c/o NADA
KTH
100 44 STOCKHOLM

Telefon: 08-791 87 97
Klubblokal: Lindstedsvägen 5
Datorhall: Brinellvägen 32

Ordf: Stellan Lagerström {08-790 78 14 (arb) 08-46 93 93 (hem)
Sekr: Olle Betzén {0758-318 48 (hem)
Kass: Henning Croona {08-799 82 80 (arb) 08-739 17 40 (hem)
Red: Hans Nordström {08-797 80 13 (arb) 0760-405 69 (modem)
Hexm: Johnny Eriksson {08-790 65 17 (arb) 08-27 24 30 (hem)
Övr: Mats O Jansson {08-34 93 20 (arb) 08-711 70 37 (hem)
Övr: Thord Nilson {08-712 00 90 (arb) 08-749 21 92 (hem)

Medlemsavgift för 1989: 89 kronor för studerande, 189 kronor för övriga
Postgiro: 433 01 15-9
Bankgiro: 344-3595

Electronic mail: *stacken@stacken.kth.se*
UUCP alternativt: *...{uunet,mcvax,...}!stacken.kth.se!stacken*
ARPA alternativt: *stacken%stacken.kth.se@uunet.uu.net*
BITNET/EARN alternativt: *STACKEN@SESTAK*
NORDUNET DECnet alternativt: *KICKI::STACKEN* eller *60456::STACKEN*
PSI alternativt: *PSI%0240200101905::KICKI::STACKEN*

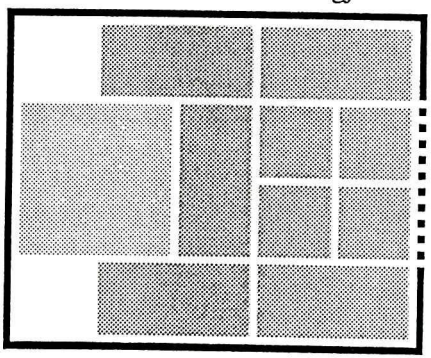
I detta nummer

Julpysselpussel

av Jan Michael Rynning

TIO små plastbitar ligger i en låda. Den största är tunnare än de andra och kan skjutas ut genom springan till höger. Och så var de bara nio... Färre än så blir de aldrig, om man inte lyfter upp dem, men det är fusk.

Hans Riesel gav STACKEN ett sådant pussel vid invigningen av Katia. Vi tog fram det en kväll och fick så småningom ut den stora biten. Sedan började vi diskutera hur man skulle lösa det med dator. Hur kan man representera pusslet i datorn? Hur flyttar man på bitarna? Hur garderar man sig mot att hamna i en loop, där man flyttar runt dem i cirkel? Hur många sätt kan man placera bitarna på, om man tar upp den och sedan lägger ner dem i lådan? Finns det något tal som antalet sätt är jämnt delbart med? Kan man nå alla de lägena från ursprungsläget? Hur många optimala lösningar finns det? Finns det något tal som antalet optimala lösningar är jämnt delbart med? Hur hittar man dem snabbast? Fundera på det och försök lösa det med dator, så kan vi ses på vårmötet och diskutera hur vi gjort!



Höstmötesprotokoll

PROTOKOLL fört vid Datorföreningen STACKENs höstmöte, avhållet torsdagen den 1:a december 1988, i sal E7 på Kungliga Tekniska Högskolan.

Närvarande i alfabetisk ordning:

Eva Albertsson,Olle Betzén, Henrik Björkman, Henning Croona, Sven Erik Enblom, Johnny Eriksson, Håkan Fröjd, Mats O Jansson, Carl-Arne Johannesson, Johan Kjellin (från §12), Mikael Kundel, Stellan Lagerström, Per Lindberg, Christer Lindström (från §12), Lars A Ljungdahl, Thord Nilson, Hans Nordström, Thomas Nyström, Jan Michael Rynning, Peter Svanberg, Bengt Åhlin och Jan Åman.

§1. **Mötets öppnande.**
Stellan Lagerström öppnade mötet nästan en timme försenat, ty närvarande var inte de stadgeföreskrivna tjugo medlemmarna förrän Henrik Björkman hade värvat en ny medlem.

§2. **Val av justeringsmän.**
Mats O Jansson och Johnny Eriksson valdes till justeringsmän.

§3. **Val av mötesordförande.**
Stellan Lagerström valdes till mötesordförande.

§4. **Val av mötessekreterare.**
Olle Betzén valdes till mötessekreterare.

§5. **Tillkännagivande av röstlängd.**
De närvarande prickades av på röstlängden.

§6. **Frågan om mötet ansåg sig stadgeenligt utlyst.**
Mötet ansåg sig vara stadgeenligt utlyst.

§7. **Frågan om dagordningens godkännande.**
Dagordningen godkänndes utan anmärkning.

§8. Val av styrelse.
Valberedningen föreslog förra årets styrelse. Henrik Björkman önskade avgå som hexmästare. Istället föreslogs Johnny Eriksson, som såg entusiastiskt på förslaget.

Till styrelse valdes så:

Ordförande:	Stellan Lagerström
Sekreterare:	Olle Betzén
Kassör:	Henning Croona
Hexmästare:	Johnny Eriksson
Redaktör:	Hans Nordström
Övriga:	Mats O Jansson och Thord Nilsson

§9. Val av firmatecknare.
Stellan Lagerström och Henning Croona valdes till firmatecknare.

§10. Val av revisorer.
Peter Löthberg och Lars S Ljungdal valdes i sin frånvaro till revisorer.

§11. Val av valberedning.
Thord Nilsson och Henrik Björkman valdes till till valberedare.

§12. Fastställande av budget.
Henning Croona meddelade att pengarna är slut och att ingen får dra på föreningen några betalningsplikter förrän mera pengar finns i kassan. Det är den senaste tidens containeraffärer som har tömt STACKENs kassa.

Försäljningen av Amis verkar ha avtagit. Jan Michael Rynning ifrågasatte om förenigen verkligen levererar Amis till de som beställer det och om det sedan faktureras och följs upp att betalning inkommer. Varken ordföranden Stellan Lagerström eller kassören Henning Croona kunde ge besked om den saken.

En uppgradering av Amis till de som har licens skulle kunna ge vissa intäkter. En samtidig försäljning av de Emacs och Amis-häften föreningen sedan flera år har liggande, skulle både ge intäkter och underlätta försäljning av Amis. Hans Nordström åtog sig att göra en sammanställning av befintliga Amis-licenser. Stellan Lagerström åtog sig att göra ett utskick till dem som har Amis-licenser.

Ett utskick av STACKPOINTER skall dock på något sätt finansieras före årets slut.

Budget fastställdes slutligen till att de pengar som kommer in under året får användas till utgifter. Årsavgiften skall i första hand användas till STACKPOINTER.

§13. **Fastställande av årsavgift.**
Årsavgiften bör åtmindstonde täcka kostnaderna för STACKPOINTER. Sex utskick à c:a 2200 SEK fodrar c:a 13200 SEK. Den traditionella årsavgiften på årtalets sista två siffror SEK skulle inte riktigt räcka till det.

Flera förslag och provomröstningar ledde till att årsavgiften fastställdes till 89 SEK för studerande och 189 SEK för icke studerande. Vidare skall på det förtryckta inbetalningskortet finnas en rad för donation, i hopp om att givmilda medlemmar, medvetna om sin förenigs besvärliga ekonomiska situation, skall donera en summa till föreningen.

§14. **Fastställande av mötesdagar.**
Månadsmöte skall hållas första torsdagen i varje månad, även om det är en helgdag. Vårmöte skall hållas torsdagen den 2:a februari 1989 och höstmöte torsdagen den 7:e december 1989.

§15. **Övriga frågor.**
På Christer Lindströms förslag tillsattes en grupp för att försöka portera Amis till PC.

Stellan Lagerström informerade om att en container från MIT har kommit, innehållandes bl.a. KL-10-maskinen MIT-MX och två Lisp-maskiner. Förrådet vid Provningsanstalten har städats och de saker som finns där är ännu mer upptravade på varandra än tidigare.

Jan Michael Rynning frågade om det fanns fler bidrag till STACKPOINTER. Tid och plats för helgens STACKPOINTER-redaktionsmöte fastställdes.

§16. **Mötets avslutande.**
Mötet avslutades efter 1 timme och 51 minuter. Därefter visade Christer Lindström hur man gör musik med dator. Ett fint initiativ, som vi tackar för.

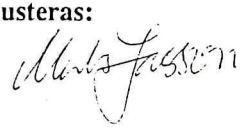
Vid protokollet:

Olle Betzén, mötessekreterare

Justeras:

Stellan Lagerström, mötesordförande

Justeras:



Mats O Jansson, justeringsman

Justeras:

Johnny Eriksson, justeringsman

Japan?

Hej STACKEN!

Om några av er är intresserade att hälsa på i Japan, kan jag hjälpa till med att hitta relativt billig inkvartering, diverse tips, etc. Jag har varit här ett tag och är ganska hemma i Akihabara (Tokyos data- och elektronikdistrikt). Problemet är att jag förmodligen blir tvungen att åka tillbaka till Sverige i slutet på mars 1989. Så ni kommer förhoppningsvis hit innan dess. Var förvarnade om att Japan är mycket mer intressant för dom som hackar hårdvara än dom som hackar mjukvara.

NIL

Martin Nilsson
Hidehiko Tanaka Lab., Dept. of Electrical Engineering,
University of Tokyo, Hongo 7-3-1, Bunkyoku, 113 Tokyo
Tel: 03-812-2111 ext. 7413
nilsson%mtl.u-tokyo.junet%UTOKYO-RELAY.CSNET@RELAY.CS.NET

Hem: Green Coop 405, Hakusan 1-29-3, Bunkyo-ku, 113 TOKYO
Tel: 03-816-5874

SuperMAIL-nytt

av Thomas Nyström

VERKAR nästan som om det här håller på att bli en stående punkt i StackPointer...

Vad har hänt?

SuperMAIL har fått ett nytt användarprogram som de flesta säkert redan har märkt. Jag har tagit Digitals MS (som är deras standard MAIL-program för TOPS-10 och TOPS-20) och anpassat det till SuperMAIL. MS är egentligen menat att köras under TOPS-10 V7.03, så det finns några finnesser som inte fungerar, t ex: Man kan inte innifrån MS anropa en editor för att redigera sitt brev, då ett speciellt systemanrop saknas på TOPS-10 V7.02.

Hur var det förr.

Tidigare låg varje användares brevfil på ALL:programernummer.MAI[3,5]. Detta har ändras i och med MS, brev ligger nu i filen MAIL.TXT på användarens eget konto. Formatet på denna fil är vanligt 7-bitars teckenformat så nu kan envar som så önskar skriva sig ett eget användarinterface och manipulera denna fil. Man måste dock "låsa" filen medan man arbetar med den, annars kan man få problem om MAIRTR lägger ned ett nytt brev i den samtidigt som man själv manipulerar den.

Starta nya programmet.

För att starta MS skriver man .R MS eller ger monitorkommandot .MS, använder man den senare formen kan man ange ett kommando på kommandoraden, t ex: .MS SEND STURE om man vill skicka ett brev. Då MS har utfört önskat kommando kommer MS att avsluta. Om man inte ger något kommando eller skriver .R MS, komman man in i MS:s kommandoläge. MS använder GLXLIB vilket innebär att man kan använda '?' och <ESC> för att få hjälp resp fylla i kommandon (dvs samma som på TOPS-20).

MS har ett flertal kommandon som ni själva får upptäcka, jag ska här berätta om de viktigaste kommandona så ni kan använda programmet.

Skicka brev.

Som jag tidigare nämnde så använder man kommandot "SEND" för att skicka

ett brev till en eller flera mottagare, man kan direkt efter "SEND" ange en lista på mottagare, ger man inte denna lista kommer MS första att fråga efter vanliga mottagre ("To:") och sedan efter kopiemottagare ("Cc:"). Då man anger mottagare kan man använda "." för att ange sig själv som mottagare. Man kommer nu att få frågan "Subject:" precis som tidigare, och då man har svarat på denna fråga får man en uppmaning att skriva in sin text, man får här reda på att man kan använda några control-tecken för att få speciella funktioner (Ctrl-E fungerar alltså inte pga tidigare nämnd anledning). Nu är det bara attskriva in sin text. Om man här trycker på <ESC> kommer man in i ett speciellt kommandoläge (man ser det genom att prompten ändras till "MS Send>" då man kan bl a kan lägga till ytterligare mottagare, lägga in fil eller annat meddelande. Om man inte har sådana önskemål kan man bara skriva in sin text på vanligt sätt och sedan trycka på Ctrl-Z för att skicka brevet, låg man i send-kommandoläget ger man kommandot "SEND" för att skicka i väg brevet. Då brevet köas får man meddelande om detta och till vem/vilka brevet köas.

Läsa brev.

En stor skillnad mot det gamla systemet är att MS håller reda på vilka brev man har läst tidigare och vilka brev som är nya, har man nya brev i sin brevlåda då man startar MS kommer MS att skriva ut en lista över alla nya brev. För att läsa sina nya brev ger man bara kommandot "READ" utan argument. MS skriver nu ut det första brevet på terminalen, man hamnar nu i ett annat kommandoläge (Prompt: "MS read>").

Här kan man nu göra flera saker med det brev som man nyss har läst, det viktigaste är "DELETE" ("UNDELETE" finns!) för att ta bort brevet (egentligen tas inte brev bort ur brevlådan förrän man lämnar MS med kommandot "EXIT") eller "ANSWER" för att svara på brevet, "ANSWER" vill ha ett argument "All" eller "Sender-only", detta anger till vilka svaret ska skickas. Man kan nu skriva in sitt svar på samma sätt som då man skickar ett nytt brev. I brevhuvudet på det brev man skriver kommer MS lägga in information om att vilket brev man har svarat på. För att läsa nästa brev om man hade flera nya brev trycker man bara på RETURN, MS kommer nu att skriva ut nästa brev, hade man inte fler olästa brev kommer man tillbaka till toppnivån. Om man får meddelande om att brev har kommit in medan man kör MS ger man bara kommandot "READ" så får man se detta brev.

För att läsa gamla brev använder man "READ nr" där "nr" är ett nummer, en lista med nummer (kommatecken mellan) eller en sekvens (start:stopp) av brev man önskar läsa. För att få en lista på sina brev använder man kommandot "HEADER ALL", man får nu en lista på alla brev man har i

sin brevlåda. "HEADER" och "READ" tar flera andra argument för att ange vilka brev man vill läsa, använd "?" för att få en lista över dessa.

Avsluta.

Om man har kommit in i ett av de extra kommandolägena (SEND/READ) kan man ge kommandot "QUIT" för att komma till toppnivån igen. I alla lägen kan man använda kommandot "EXIT" för att lämna MS, man måste använda detta kommando för att MS ska ta bort de brev ur brevfilen man har gjort "DELETE" på (eller använda sig av kommandot "EXPUNGE" som gör detta utan att lämnas MS).

Gamla brev.

Gamla brev ligger kvar på [3,5], man kan flytta dessa till sin nya brevfil genom att starta det gamla programmet och ge följande kommandorad: "MOVE * MAIL.TXT", dina brev flyttas (tas bort i den gamla filen) till MAIL.TXT som MS sedan kan läsa, formatet är nämligen kompatibelt i denna riktning, dvs MSGH (gamla programmet) kan inte läsa filer som MS har skrivet.

Initiering.

Man kan ha en fil "MS.INI" på sitt konto med kommandon man vill ska utföras då man startar programmet. Denna fil skapar man med AMIS (använder ni någon annan editor?) för att ange vilka kommandon som ska utföras, t ex kan man lägga sin adress i brevhuvudet, ett exempel på detta kan ni finna på MAI:MS.INI på Kicki.

In Binary language, 00 is none,
Which cannot be said to be new.
Nor is it novel that 01 is one,
But in Binary 10 is two!

If you ponder and strive, perhaps you'll contrive
A matrix from which you will see
That 101 stands for the numeral five,
While the simple 11 equals three.

Computers of course speak Binary perforce,
Though we mortals the language abhor;
We'd sooner endorse the numerical Morse
But we're not who the language is 100.

Francis Cartier

Turerna kring nya datorhallen.

Sagan om ett systemanrop

av Thomas Nyström

DET VAR EN GÅNG en dator som hette Kicki, denna dator var en mycket fin dator som till och med hade två processorer. För att datorn skulle kunna fungera hade de lokala hackarna modifierat standardoperativsystemet, detta nya operativsystem kallade man SuperTOPS. Eftersom man tyckte det var tråkigt att sitta och köra ensam på datorn hade man anslutit datorn till tre olika nätverk, bland annat till Swedish University Network, SUNET kallat. SUNET var ett DECnet.

För att man från denna datorn skulle kunna skriva brev och ta emot brev så skrev man SuperMAIL. Eftersom det vid denna tid hade bestämts att man skulle använda domänadressering på alla datorer i universitetsvärlden som var anslutna till olika nätverk och DECnet inte använde domänadresser så hade man i SuperMAIL ordnat det på så vis att alla brev som skulle till en dator på DECnet och som inte SuperMAIL kände till försågs med domänen ".sunet.se" så att man följde detta med domänadresser.

Detta fungerade mycket bra tills den dag då man bestämde att man skulle koppla ihop detta DECnet med likadana nätverk i grannländerna. Man kunde nu inte lägga till ".sunet.se" på alla brev som skulle till en dator på DECnet, utan SuperMAIL var tvungen att veta vilket land datorn fanns i. En av programmerarna som underhöll SuperMAIL beslutade då sig för att fixa detta: SuperMAIL skulle fråga SuperTOPS om vilket område, eller area som det heter på DECnetsiska, datorn hörde till.

SuperMAIL hade tidigare använt systemanropet "DNET." för att fråga SuperTOPS om datorn fanns i DECnet. Detta anrop kunde man nu inte använda eftersom det inte talade om vilken area datorn låg i, om den låg utanför den area som Kicki hörde till. Programmeraren som skulle göra jobbet var nu tvungen att se sig om efter ett annat systemanrop, han kunde då efter lite letande hitta ett anrop som hette "NTMAN.".

Då han letade i manualerna efter en beskrivning av detta anrop kunde han så småningom hitta en beskrivning som berättade hur argumentblocket såg ut, men anropet var inte fullständigt beskrivet och i inledningen av beskrivningen stod det att anropet inte skulle användas av användarprogram. Detta brydde sig nu inte programeraren om

eftersom Kicki körde SuperTOPS, som var det egna operativsystemet. Nu var frågan hur man skulle använda anropet.

Programmeraren började med att titta i olika beskrivningar som fanns för att se om det fanns beskrivet någon annanstans, det fanns det inte. Programmeraren visste att det fanns ett program som hette NODNAM, som man använde för att definiera namnet på de olika datorer som fanns anslutna till DECnet. Han letade reda på källtexten till detta program och tog reda på om programmet använde detta anrop för att sätta datornamn; jodå, det gjorde det, men han blev inte klokare av detta.

I beskrivningen som fanns i manualen så stod det att anropet bara skulle användas av systemprogram som skulle övervaka DECnet, han bestämde sig då för att leta reda på anropet i programmet OPR, som man använde för att göra dessa saker. Han fann då att detta program inte gjorde detta systemanrop utan överlämnade detta till programmet NML4, "Jaha, då får jag väl titta i NML4 hur man gör" tänkte programmeraren, och letade raskt reda på källtexten till programmet, han upptäcker då till sin bestörtning att NML4 är skrivet i BLISS, som är ett mycket konstigt språk, programmeraren börjar nu bli alltmer desperat.

Då han håller på med detta får han ett påpekande av en av de andra som har hand om datorn att man måste fixa detta, han talar då om att han håller på och berättar om sitt problem, programmeraren får då förslaget att titta i programmet MM, som körs på samma dator fast under ett annat operativsystem, källtexten till MM ska ligga på datorn Xerxes. Då programmeraren försöker koppla sig till Xerxes så finner han att Kicki just då inte kan koppla sig ut på DECnet eftersom den lokala DECnet-växeln inte fungerar. Han drar en djup suck och konstaterar att han nu kan få gå och lägga sig och sova istället.

Nästa gång som programmeraren försöker koppla sig till Xerxes så går det bättre, han letar nu i källtexten till programmet och hittar så småningom den del som verkar ta reda på datornamn. Återigen har han dragit en nitlott, och kan inte ens hitta någon användning av anropet "NTMAN.". "Hm, om man skulle..." tänker han, "Om man skulle ta och titta i källkoden till operativsystem", sagt och gjort det gör han.

Han kan nu efter lite letande bland operativsystemets alla moduler så småningom hitta den modul som utför anropet "NTMAN.", det var en alldeles egen modul som gjorde allt detta. Programmeraren hade tidigare när han läste den lilla beskrivning som fanns av "NTMAN." kommit på att det var funktionen ".NTMAP" man skulle använda, för den funktion skulle kunna översätta ett datornummer till ett namn eller vice versa. Han började leta i modulen efter denna funktion, "Jodå, där var den,

nu ska vi se... Jaha... 'entity', vad fåglarna betyder det?", tänkte han, "och datafält och bytepointer, hu så hemskt, det här verkar bara konstigt...".

Programmeraren blev nu förvirrad och bestämde sig rask för att prova, han skrev ihop ett litet program i C för att testa detta. Han väljer funktionen ".NTMAP" och stoppar in lite data som verkar rimliga, och kör, ERROR, det gick inte. "Parameter missing" fick han som felkod. "Suck, vad betyder detta jag har ju skickat med en pekare på namnet till en nod...?", tänkte han och kände hur trött han verkligen var, han bestämde sig för att sova!

Nästa dag försökte han igen; "Om jag skulle ta och titta på NODNAM igen hur det programmt använder argumenten?" tänkte han och började leta i det programmet, jaha, "entity" ska vara någon konstig form av nodnamnet och med *åtta-bitars* bytar, jaha, vi gör nytt försök, samma fel igen, "parameter missing".

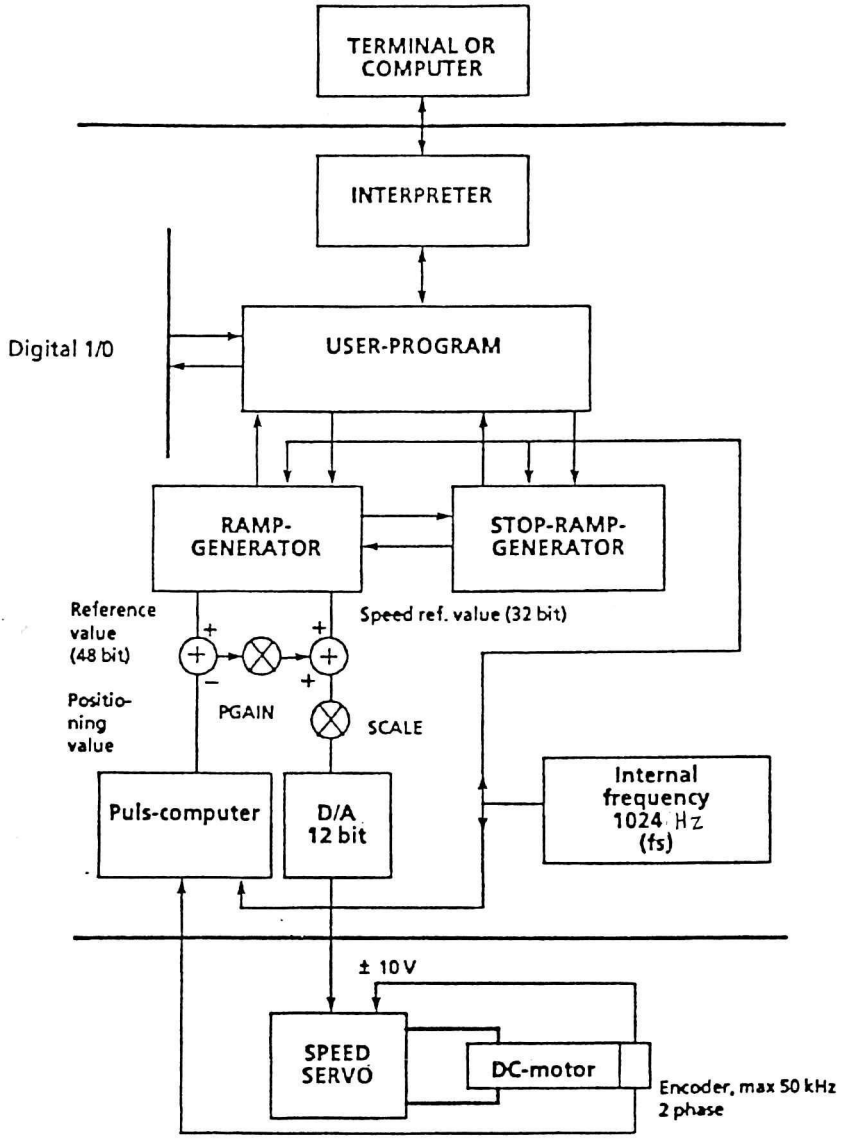
Programmeraren bestämde sig nu för att leta lite mera i källkoden till operativsystemet, "Jaha, där försöker man plocka upp ett nodnummer... och där ett nodnamn, jaha, om man inte anger något nodnamn så översätter den nodnumret till ett nodnamn. Verkligt intressant...", programmeran började känna sig hoppfull igen. "Nu ska vi se hur man använder argumenten, jaha, nodnumret är två 8 bitars bytar först i datafältet, och nodnamnet kommer också från datafältet, och längdbyte ska det ha..."

Programmeraren skrev raskt ett litet testprogram (denna gång i MACRO-10) och provkörde, "Men titta, nu fick jag ett annat fel, 'Argument too long'". Han letade nu reda på varifrån den felkoden returnerades, "Jaha, den kom då operativsystmet ser att längden på nodnamnet är längre än 32 tecken, hm, nu ska vi se... Jaha, nodnumret ska ligga först i datasträngen och *efter* detta ska nodnamnet ligga, vi prövar igen!" tänkte han och ändrade sitt testprogram, "Titta, nu fungerade det, och där i stället för nodnumret har något hamnat, jodå, det är nodnumret".

Programmeraren hade nu tagit reda på hur anropet fungerade och kunde nu lägga in det i SuperMAIL så att SuperMAIL bara lade till ".sunet.se" på brev som skulle till svenska noder.

Så gick det till när jag skulle låta MAIRTR titta på areanumret för att inte skicka brev till Norge med tillägget ".sunet.se".

God Jul och Gott Nytt År!


TERMINAL OR COMPUTER
INTERPRETER
Digital 1/0
USER-PROGRAM
RAMP-GENERATOR
STOP-RAMP-GENERATOR
Reference value (48 bit)
Speed ref. value (32 bit)
Positio-ning value
PGAIN
SCALE
Puls-computer
D/A 12 bit
Internal frequency 1024 Hz (fs)
± 10 V
SPEED SERVO
DC-motor
Encoder, max 50 kHz 2 phase

Figur 1.

PDI—Peanut Defence Initiative

eller

Hur man bygger en jordnötskastare

av Thord Nilson

DET VAR på våren 1988 och NASACON IX närmade sig med stormsteg.

NASACON är en årlig SF-kongress som hålles någonstans i Fisksätra. Förutom det litterära, arrangeras även det så kallade jordnötsloppet.

Jordnötsloppet är en tävling där det gäller att förflytta en jordnöt minst två meter, över ett av tävligsledningen uppmätt område. Alla medel är tillåtna utom att över området tillföra mänsklig energi. Dvs, man får kasta över nöten, men inte bära den. Stilpoäng utdelas. Den som får högst stilpoäng vinner tävlingen.

På NASACON VIII, året innan, hade SAFA (Svenska Arbetsgruppen För Algoritmforskning) ställt upp med PDI i jordnötsloppet.

Den dåvarande PDI-konfigurationen bestod av en voice coil-magnet från en utrangerad RP03:a. Den används för att förflytta läs/skrivhuvudena. En sådan magnet är klart imponerande. Efter transport i bil, har de flesta lösa verktyg man hade i bagageluckan hittat dit. I stället för den medföljande spolen tillverkades en egen av en avsågad 1-liters T-spritflaska, som visade sig ha samma diameter som originalspolen. På toppen av flaskan (vid korken) monterades ett cigarretui, i vilket sedan jordnöten skulle placeras.

För att skjuta iväg spolen anslöts sedan ett kraftigt kondensatorbatteri via en kontaktor. För att öka imponatorcf-

fekten ytterligare styrdes det hela från en portabel dator med talsyntesprogram.

Det hela blev mycket uppskattat och SAFA vann tävlingen.

Nu gällde det alltså att försöka hitta på något ännu häftigare.

På arbetet (Mikrologik AB) satt jag för tillfället och uppdaterade programvaran till GME-systems positioneringssystem Mikropos 300. Det var då jag kom att tänka på möjligheten att montera en sked direkt på motoraxeln och sedan snäppa iväg jordnöten med skeden på något intressant sätt.

Mikropos 300 är styrdatorn i ett generellt positioneringssystem för enklare maskinstyrningar etc. Ett sådant system består i huvudsak av: Servomotor med tachometer för hastighetsåterkoppling och en pulsgivare för positionsåterkoppling. Till detta kopplar man en drive och en Mikropos. En drive är i princip ett effektsteg med en hastighetsregulator. En analog insignal styr motors hastighet, 0 V = stå stilla, +5 V = halv fart framåt, +10 V = full fart framåt, –5 V = halv fart bakåt etc.

Mikropos är kopplad till pulsgivaren och denna styrsignal. Från Mikropos kan man sedan styra motorns position, hastighet och acceleration, se fig 1.

Ett timerinterrupt i Mikropos beräknar varje ms en ny önskad hastighet och position för motoraxeln. Den beräknade positionen jämförs med den verkliga positionen (från pulsgivaren). Denna skillnad kombineras med den önskade hastigheten för att ge ett nytt hastighetsbörvärde till driven.

För att sedan kunna styra det hela på ett effektivt och flexibelt sätt finns det en interpreter som interpreterar ett för positioneringsändamål specialutvecklat språk kallat P. Språket P påminner till utseendet om en blandning av Basic och Assembler, se listningen.

Nu skulle jag alltså testa ifall det skulle vara praktiskt genomförbart att montera en sked eller liknande på motoraxeln, lägga en jordnöt i skeden och sedan på ett konstfullt sätt skicka iväg jordnöten minst 2 meter.

En plastsked monterades provisoriskt och försök med att kasta muttrar påbörjades. Någon timma, ett antal plastskedar och många muttrar senare skickades den första muttern med en kraftfull forehand i väggen. Det fungerade alltså, med serve!

Nu behövde följande göras/byggas:

Ett riktigt fäste för skeden, en ställning för motorn, en laddare av något slag (det vore ju inget roligt att bara lägga jordnöten i skeden), skriva styrprogram i P för att kontrollera skedens rörelser, ett häftigt menyprogram till I*M PC med imponatoreffekt, affischer och ett

föredrag med OH-bilder visandes PDI:s funktion.

Vi delade upp arbetet så att Nils tog hand om menyprogrammet, Jörgen fick fixa föredraget, jag P-programmet, och jag + Jörgen de mekaniska bitarna.

En helg ca 1 månad före NASACON var undertecknad hos Jörgen för att bygga behövliga mek-bitar. En ställning till motorn svetsades samman och slipades till av ett par meter fyrkantsrör. Detta skedde under svåra protester från en finsk barnkör, som övade ute på gräsmattan. Vi fick bara använda slipmaskinen i korta intervaller, medan kören drack saft och åt bullar eller var allmänt religiösa.

Därefter skulle mataren för jordnötterna konstrueras. Vi tänkte oss att nötterna skulle ligga i ett magasin (à la k-pist) som ledde till en ränna och sedan flyttades därifrån med en kolv. Detta visade sig dock inte fungera i praktiken.

Problemet var att toleransen hos jordnötterna var betydligt sämre än hos 9 mm:s ammunition. Efter ett antal timmars provande och testande kunde vi klara max två till tre jordnötter i magasinet utan att de fastnade i varandra. Det är förvånansvärt på hur många olika sätt jordnötter kan trassla ihop sig. Vi gjorde en back-tracking (som det heter i Prolog) och hamnade till sist på den välkända Colt-principen. Den verkade tillräckligt enkel för att kunna fungera med jordnötter.

Alltså, en revolver med plats för tolv jordnötter sågades ut ur en spånplatta, en ställning löddes ihop av kretskort och stagades med tråd och ännu mera kretskort. Nu återstod bara rännan som nötterna skulle glida nedför för att hamna snyggt och prydligt i skeden. Det visade sig vara ett icke-trivialt problem. (För de som inte tror mig: Skaffa en påse oöppnade jordnötter och prova själv!) Någon gång vid midnatt var vi nöjda med resultatet.

En vecka senare fanns en preliminär version av styrprogrammet klart. Det var nu dags för samkörning mellan PC:n och Mikropos, dock fanns det inget menyprogram. Nils S höll på, men var inte klar.

Nils hade hittat en Modula-2-kompilator för PC på Kicki. Som demoprogram för en medföljande multitaskhanterare fanns ett terminalemulatorprogram. Nu höll Nils på att hacka om detta program till ett PDI/2-menyprogram. Detta för att enklare kunna köra seriekommunikationen mot Mikropos.

Medan jag trimmade positioneringsprofilerna till Mikropos skrev Jörgen följande spec till Nils:

Nya specar på de strängar som skall styra PDI/2, the sequel.

PDI Hardware Team har i sin godhet beslutat införa en del nya strängar att styra nötkastaren med. Dessutom har en ny form av kommunikation införts. Allt för att glädja Herr Programmeraren av användarinterfejset.

Ny kommunikation.

Prompten och loginmeddelandet fås som tidigare.

Nu gäller det att mata värden till och läsa värden ur register istället. Läser gör man på följande sätt:

DISP Rnn<CR>

Svaret blir:

Rnn= m

och en ny prompt.

Att stoppa ett värde i ett register går till på följande sätt:

LET Rnn, m<CR>

varvid svaret blir en ny prompt.

Nya strängar.

Alla kommandon enligt tidigare spec. har utgått och ersatts av att man matar in värden i register och skickar till Mikropos istället. Svaren får man genom att läsa andra register. Denna metod bryter inte exekveringen, som ^C gjorde. Värdena och åtgärderna för register R20 kvarstår dock i stort sett oförändrade.

Registren kan pollas när som helst och svar kan fås när som helst under körningen.

Vi har nu fyra register, R20, R21, R22 och R23.

Reg	Läsfunktion	Skrivfunktion
R20	status	N.A.
R21	= 0, kommandot accepterat. Nytt kommando kan ges omedelbart, ett kommando kan ligga pending.	Skjutochladdkommando.
R22	Antal nötter i revolvern.	Antal nötter kvar efter nyladdning.
R23	Hur många nötter kvar att automatskicka.	Antal nötter att automatskjuta.

Detaljer, R20.

Om R20 är	gäller följande status
0	MP måste startas, ge "^C RUN<CR>"
1	Klart att ge startkommando
2	Klart att ge startkommando

Detaljer, R21.

Om R21 läses till noll är det nyss givna kommandot accepterat. Alla andra värden skall ignoreras.

Värden skrivna till R21 är skjutkommandon. Dessa är följande:

Tal	Betydelse
1	Misslyckad laddning av jordnöt, sked för hög.
2	Misslyckad laddning av jordnöt, sked för låg.
3	Lyckad laddning, svagt kast (0.5 meter).
4	Kastar jordnöt och tar den igen, kastas därefter enl 3.
5	Lyckad laddning av jordnöt, men tappar den senare.
6	Vevar skeden fram och åter, winding up spoon power.
7	Enkelt kast av jordnöt. Tävlingskast utan serve.
8	Slag med serve, framåt, tävlingsslag.
9	Mycket hårt slag med serve, bakåt. Jordnöt sannolikt krossad

10	Livsfarligt kommando, skeden går 600 rpm samtidigt som magasinet töms. Jordnötter sprutar åt alla håll.
11	Automateld, tävlingskast, snabbare än om det görs som upprepade 7 eftersom, tja skit i det. Antal jordnötter som skjuts anges i R23, före skottet.
12	Skeden går som sekundvisare, tills annat kommando ges. Skjuter nötter varje minut, tills nötterna är slut.
13	Automateld med krossfunktion, mycket illusoriskt, jordnötsdimma.
14	To be determined.
...	

Detaljer, R22.

Häri skrivs hur många nötter det nu finns i revolvern. Om Mikropos skulle ha en avvikande mening, dvs när man skjutit slut och laddat. Max antal nötter är tolv (12).

Detaljer, R23.

Häri skrivs hur många nötter nästföljande automateldkommando skall skjuta. Automateld genom Mikropos försorg blir snabbare än om du skall ge flera skjutkommandon i serie på grund av att skeden inte behöver återställas efter varje skott.

Värdet som skrivs i R23 får naturligtvis inte vara högre än det som läses ur R22.

Jörgen

Två dagar innan tävlingen:

Nu var det hög tid för samkörning. Nils var dock inte riktigt klar ännu, några detaljer var kvar att fixa, bl a läsning av registervärden från Mikropos.

Samkörningen gick hyfsat tills läsning av registervärden skulle provas. Ett allvarligt problem upptäcktes: Om det blev ett enda överföringsfel på seriekanalen så hamnade alla processerna i PC:n i WAIT, och det fanns ingen timeout-möjlighet.

Olika alternativ övervägdes och förkastades. Det hela skulle gå att lösa snyggt om multitaskhanteraren hade varit styrd från timer-interruptet eller hade haft en funktion motsvarande

Simula Simulations HOLD. Vid midnatt var det dags att ge upp för dagen.

Hemma, med källkoden till multitaskhanteraren försökte jag laga till en HOLD-funktion. Efter någon timmas funderande fanns ett utkast klart.

En dag innan tävlingen:

Den nya HOLD-funktionen provas och visar sig fungera. Meny-programmet skrivs om för att utnyttja den, och allt verkar funka OK. Nils ritar nya skärmbilder, ändrar andra och fixar in nya finesser i programmet som diverse oljud mm.

Under tiden plockar jag och Jörgen ihop de saker som skall användas, fixar kablar till högtalare etc.

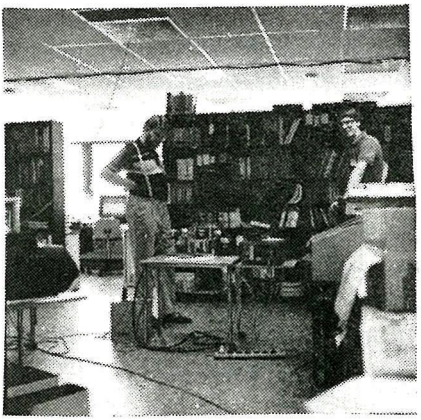


Bild 1: Testuppkopplingen på Mikrologik begrundas av Jörgen och mig.

Tävlingsdagen:

Vi har nu en konfiguration enl fig 2.

På förmiddagen ställs allt upp för en sista testskjutning. Allt fungerar OK. (Se bild 1 och 2) Vi monterade ned grejorna och lastade i två bilar för färd mot Fisksätra skola.

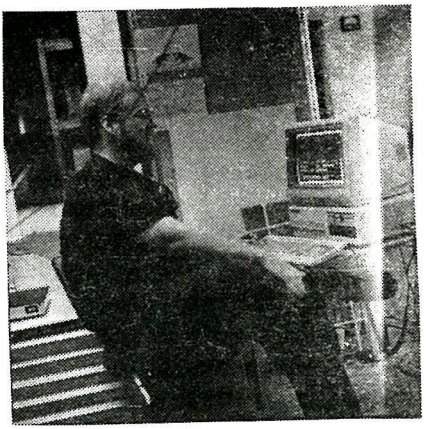
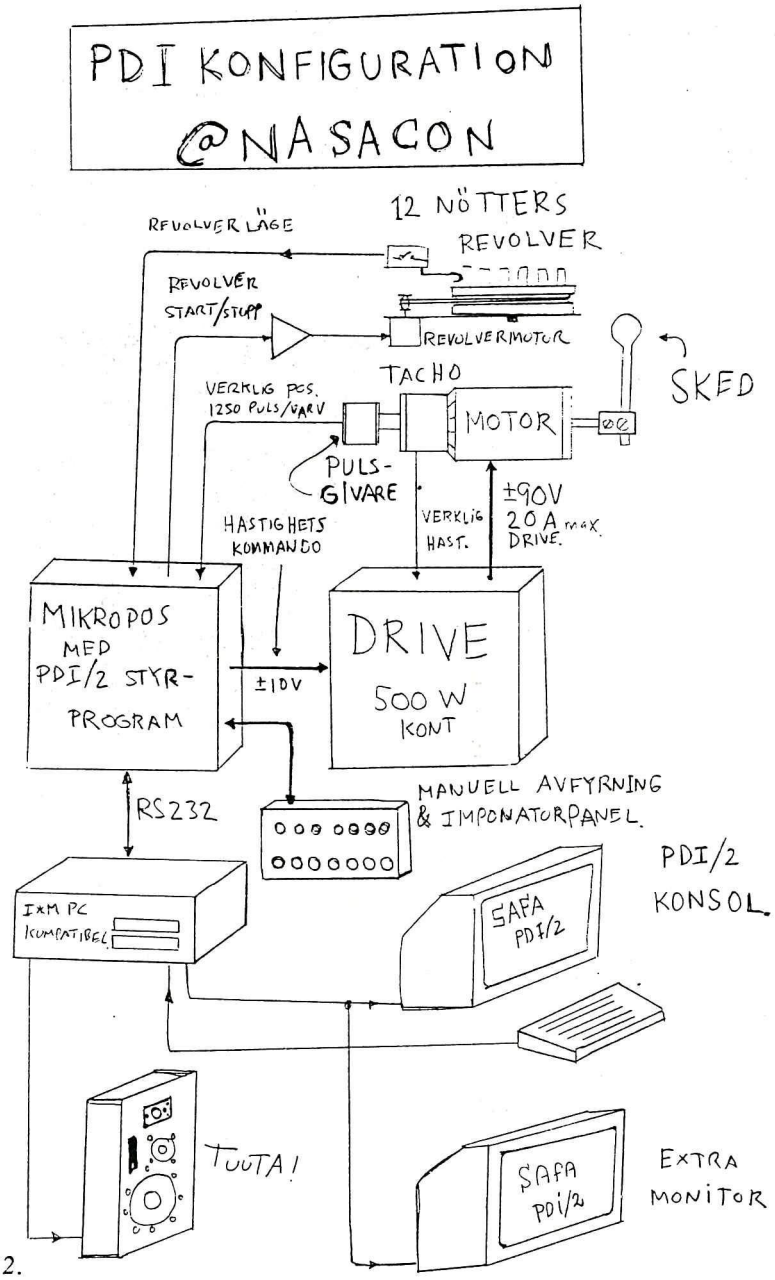


Bild 2: Nils som PDI Commander vid testkörning på Mikrologik.

Allt bars in i stora aulan i väntan på tävlingen. Vi hade ansökt om att bli sista deltagare, vilket tävlingsledningen också beviljade.

Jörgen, iklädd vit rock, och något svamligare än vanligt drog teorin bakom PDI. Han hade faktiskt utarbetat ett program som följdes i huvudsak:


PDI KONFIGURATION
@NASACON
12 NÖTTERS
REVOLVER
REVOLVER LÄGE
REVOLVER START/STOPP
REVOLVERMOTOR
SKED
TACHO
VERKLIG POS. 1250 PULS/VARV
MOTOR
PULS-GIVARE
±90V 20A max DRIVE.
VERKLIG HAST.
HASTIGHETS KOMMANDO
MIKROPOS MED PDI/2 STYR-PROGRAM
±10V
DRIVE 500 W KONT
RS232
MANUELL AVFYRNING & IMPONATORPANEL
PDI/2 KONSOL
I*M PC IKOMPATIBEL
SAFA PDI/2
TUUTA!
SAFA PDI/2
EXTRA MONITOR

Figur 2.

Peanut Defence Initiative
the sequel
på Nasacon 9, 1988

1. Sakerna ställs upp inför publik
2. Overheadpresentation
3. Skjutning
4. Undertecknande av Onlookers Award

2. Overheadpresentation

SAFA och PDI

PDI:s historia

PDI och framtiden

PDI i ett militärhistoriskt perspektiv, redan Lionardo da Vinci...

PDI är naturligtvis helt ofarligt för Sveriges befolkning. Som vi redan har visat med vår förra demonstration, den elektromagnetiska kanonen, är de förstörelsevapen SAFA framställer helt riskfria för den egna nationen. Naturligtvis är eldkraften så stor att ingen annan nation skulle våga ett anfall. Som vanligt framställer SAFA bara vapnen för forskningens skull och skall de sedan användas är det ju politikernas sak. Vi tar inte ställning till den saken. Vi gör bara vad som är bäst för landet.

(Overhead ON!)

Projektet studerades denna gång mycket ingående och under några hektiska veckor i juni färdigställdes och togs hela utrustningen och all programvaran fram. Det var The SAFA Night Hacking Team, vilka alla finns representerade här idag, som arbetat till långt in på nätterna vid terminaler och lödkolvar för att få allt färdigt till denna viktiga demonstration, idag.

För att ni skall kunna föreställa er vilka svårigheter detta projekt bjöd på skall jag ta er på en kort rundvandring genom årets jordnötskanons alla delar.

Gå igenom konstruktionen

Sluta med att utlova en Award

3. Skjutning

Och nu får jag be alla intresserade stiga fram och flockas kring bildskärmen men se upp för kanonen, eftersom den är farlig och kan ställa till med personskador om den vidrörs. Det visade sig bland annat under projektets gång, då skedenheten lossnade från motordelen och flög och studsade genom hela laboratoriet, slog märken i en printer och slutligen hamnade i ett angränsande rum.

Presentera maskinens olika delar

Tala om problemen som uppstått under projektets gång, olika skjutfel, nötterna hamnade på alla andra ställen än där de skulle. Det stod från början helt klart att nötterna måste automatmatas. Inte kunde man hålla på att lägga i dem för hand, inte. Vi hade på gång en helt annan matarteknik, nämligen en matare med liggande nötter, men det visade sig inte fungera så bra, revolvern fick bli lösningen. Många teorier prövades, rörande det optimala skottet och vi ska be att få visa några av dem, senare.

Spoon-enheten var ursprungligen av plast, men detta gav sådana problem med flygande splitter i lablokalen att den snabbt utbyttes mot en i ett mera beständigt material, nämligen ståååååål! Dessutom visade det sig att vi behövde ett specialfäste som tålde höga accelerationer, varför ett dylikt svarvades av Delrin.

Demo av The Spoon Unit. Nu måste fjädern spännas, så vi tar och vevar upp spoonenheten.

Veva skeden fram och åter-demon

Demo av olika laddnings-metoder

Misslyckad laddning, sked för hög LOAD TOO HI
Misslyckad laddning, sked för låg LOAD TOO LO
Tappar nöten LOAD 'N DROP
Släng upp nöten och fånga igen CATCH

Demo av eldkraften, tidigt försök

Svagt kast LOAD OK 50 CM

Tävlingskastet, får vi nu be tävlingsledningen frambära N Ö T E N och ladda den i revolvern!

Enkelt kast — tävlingsjordnöt — ONESHOT 1

Andra, kraftigare skott

Slag med serve ONESHOT 2
Hårt slag bakåt BACKFIRE

"All out superpower confrontation"

kommando 10, slumpmässigt nötsprut, s k asynkron kross LETHAL!!!
kommando 11, lyckad automateld AUTOFIRE

Medan domarna nu bestämmer sig för vilket bidrag som skall vara det vinnande, låter vi klockan gå! Observera med vilken precision vårt urverk spinner, etc.

Klock-kommandot CLOCK

Under tiden kan vi få be alla hugade åskådare att stiga fram och emottaga ett bevis på att ni stått ut med denna demonstration, the PDI Onlookers Award.

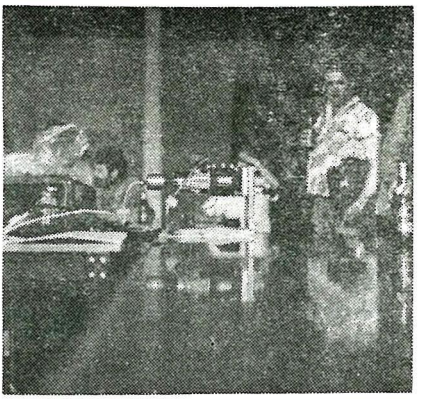


Bild 3 och 4: Vid NASACON, PDI utför CATCH, jordnöten kastas upp i luften, skeden backar ett varv för att sedan fånga nöten när den faller ner.

Det hela vart så lyckat att vi tyckte det var synd att bara deltagare på NASACON fick bevittna det hela, så efter ett tag fanns följande meddelande att läsa på diverse ställen:

SAFA visar PDI/2 på STACKEN-möte

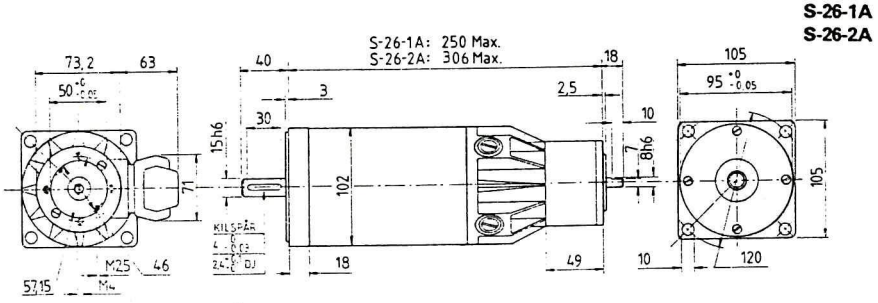
På STACKEN-mötet torsdagen den 7/7-1988 kl 19:15 i sal E7 kommer SAFA att demonstrera PDI/2, årets vinnare av jordnötsloppet på NASACON IX.

PDI står för "Peanut Defence Initiative" och PDI/2 består i huvudsak av en datorkontrollerad motordriven sked med vilken man på ett antal konstfulla sätt kan slunga iväg en jordnöt. Till denna sked är ett revolvermagain för 12 nötter anslutet. Maximala eldhastigheten är omkring 1 jordnöt/sekund.

Eftersom PDI/2 blev synnerligen uppskattad på NASACON och är kraftigt datoriserad så är det något som en sann hacker bara måste se.

Missa inte detta unika tilfälle!!!

Detta skedde också på utlovad tid, samt på kåren och vid ett senare tillfälle i STACKENs lokal.



Faktaruta:

Motor: ElectroCraft S-26-1A.

Kont moment:	2.5 Nm
Toppmoment:	20.7 Nm
Max varvtal:	3000 rpm
Max kont uteff:	700 W
Mek tidskonst:	6 ms
El tidskonst:	1.7 ms
Max kont ström:	7.5 A
Max toppstöm:	60 A
Vikt:	7 kg

Pulsgivare: Leine Linde, 1250 pulser/varv.
Detta pulstal multipliceras sedan med 4 internt i Mikropos vilket ger en effektiv upplösning på 5000 pulser/varv.

Drive: GME System PWE 6010.

Styrsystem: GME System Mikropos 300.

Och till sist, för de verkliga entusiasterna: Styrprogrammet i P för PDI/2:

```
;*****************************************************************
;*
;*      PDI/2 -- Peanut Defence Initiative, the sequel.
;*
;*      Peanut Throwing Program.
;*
;*      Motor/Drive:            Max RPM: 2500
;*      Pulse transducer:       1250 pulses/turn * 4
;*                              ==> 5000 pulses/turn.
;*
;*****************************************************************

.define status = r20    ; Ok for PC to restart prog if status=0
.define command = r21   ; Set by PC to execute a command.
.define peanuts = r22   ; Set by PC to indicate # of peanuts in revolver.
.define stsidle = 0     ; Program not running, PC must send ^C RUN
.define okstart = 1     ; Ok for PC to send command.
.define running = 2     ; Command in progress.

        .org    1               ; Start of program.

restart:
        serial mode 16          ; Enable XON/XOFF.
        let     status, running
        vector  10,ctrlc        ; Control-C trap....
        gosub   initpdi         ; Initialize.
        let     status, okstart
        goto    wcmd            ; Then  ready for command.
```

```
        .org    25
;       .linestep 2
        .linestep 1
;
;       This is the main lupe. -- Top level -- Waiting for command.
;

wcmd:   let     status, okstart
        gosub   rblink          ; Twinkle twinkle little led's.
        gosub   swdec           ; Decode switches.
        if      0 = command, wcmd ; Continue waiting if no command.

        gosub   cdec            ; go and decode command.
        goto    wcmd

;*****************************************************************
;*
;*      Decode input switches.
;*
;*****************************************************************

swdec:  if lo in and 1, wave    ; wave spoon.
        if lo in and 2, throw1  ; swing, mode 1.
        if lo in and 4, throw2  ; swing, mode 2.
        if lo in and 8, throw3
        if lo in and 16, throw4
        return

;*****************************************************************
;*
;*      Routines to take care of input switches.
;*
;*****************************************************************

wave:   goto    wavef

throw1: gosub   gohome          ; Start at home pos.
throw1a:
        gosub   swing1
        if lo in and 2, throw1a ; Do again if button pressed.
        return

throw2: gosub   swing2
        return

throw3: gosub   swing3
        return

throw4: gosub   swing4
        return

;*****************************************************************
;*
;*      Decode input command.
;*
;*****************************************************************

cdec:   sub     command, 1
        if      0 = command, cmd1               ; load point to high
        sub     command, 1
        if      0 = command, cmd2               ; load point to low
        sub     command, 1
        if      0 = command, cmd3               ; right but to slow.
        sub     command, 1
        if      0 = command, cmd4
        sub     command, 1
        if      0 = command, cmd5
        sub     command, 1
```

```
        if      0 = command, cmd6
        sub     command, 1
        if      0 = command, cmd7
        sub     command, 1
        if      0 = command, cmd8
        sub     command, 1
        if      0 = command, cmd9
        sub     command, 1
        if      0 = command, cmd10
        sub     command, 1
        if      0 = command, cmd11
        sub     command, 1
        if      0 = command, cmd12
        sub     command, 1
        if      0 = command, cmd13      ; This is the destructor command.

        return                          ; Noo good, wait for new command.

        .org    100

;*****************************************************************
;*
;*      Routines to take care of the individual commands.
;*
;*****************************************************************

cmd1:   let     r2,150          ; offset position -- load to hi.
        goto    faicmd

cmd2:   let     r2,-200         ; load to low.
        goto    faicmd

cmd3:   let     r2,0            ; load ok.
        goto    faicmd

faicmd: acc     50
        ret     50
        pos speed 30 000
        pos abs r2
        wait    rdy
        gosub   loadnut
        clr     tmr
        wait tmr > 1000
        gosub   xthrow
        return

;
;       CATCH.
;

cmd4:   gosub   gohome                  ; Goto home position.
        gosub   loadnut                 ; Load a nut.
        gosub   jiggle                  ; Jiggle it in position.
        pos     speed 40 000
        acc     50
        ret     50
        pos     rel -145                ; Backup to launch postion.
        wait    rdy
        acc     200
        ret     300
        pos     rel 500                 ; Throw nut into the air.
        wait    rdy
        pos     speed 60 000
        acc     600
        ret     400
        pos     abs -5000               ; Back spoon one turn when nut
        wait    rdy                     ; is in the air.
        clr     tmr
```

```
        wait    tmr > 1500              ; Wait for nut to land on spoon
        gosub   xthrow                  ; then lazy throw away.
        return

;
;       Load And Drop.
;

cmd5:   gosub   gohome                  ; Load nut and then drop it.
        gosub   loadnut
        acc     1                       ; slow backwards until drop.
        ret     1
        pos     rel -600
        wait    rdy
        return


cmd6:   goto    wave

cmd7:   goto    swing1

cmd8:   goto    swing2

cmd9:   goto    swing3

cmd10:  goto    swing4

cmd11:  gosub   gohome
cmd11n: if      0 = r23, cmd11e
        if      0 > r23, cmd11e
        gosub   swing1                  ; Eject it...
        sub     r23,1
        goto    cmd11N
cmd11e: return

;
;       CMD12 -- Second hand of watch, eject peanut every minute,
;                 until out of peanuts or other command.
;

cmd12:  gosub   gohome
        clr     tmr

cmd12l: let     r0,20/2         ; One minute.... in 30 seconds.
cmd12m: pos     rel - 83*2
        wait    tmr > 1000
        pos     rel -83*2
        wait    tmr > 2000
        pos     rel -84*2
        wait    tmr > 3000
        dec     tmr     3000
        if      0 < command, cmd12e     ; Exit if new command.
        loop    r0, cmd12m
        gosub   swing1                  ; Throw a nut.
        if      0 = command, cmd12      ; Start all over again if no new
cmd12e: return                          ; command.

;
;       CMD13 - Automateld med krossfunktion.
;

cmd13:  if      0 = r23, cmd13e
        if      0 > r23, cmd13e
        gosub   swing3                  ; Eject it...
        sub     r23,1
        goto    cmd13
cmd13e: return
```

```
;*****************************************************************
;
;       Perform complete swing, mode 1.
;

swing1: let     status, running
        gosub   loadnut
        gosub   fthrow
        return

;
;       Perform complete swing, mode 2.
;

swing2: let     status, running         ; Indicate status is running.
        Gosub   gohome                  ; Goto home position.
        Gosub   loadnut                 ; Load peanut.
        Gosub   jiggle                  ; Jiggle peanut in position.
        Gosub   ffire                   ; Fire!
        return                          ; Wait for next command.

;
;       Perform complete swing, mode 3.
;

swing3: let     status, running
        gosub   gohome
        gosub   loadnut
        gosub   jiggle
        gosub   bfire                   ; Backwards fire.
        return

;
;       Perform complete swing, mode 4 -- This empties revolver.
;

swing4: let     status, running
        acc     50
        speed   50 000
        wait    spd = 50 000

swi41:  if      0 = peanuts, swi4e              ; Exit when no more peanuts.
        if      0 > peanuts, swi4e
        gosub   loadnut
        goto    swi41

swi4e:  speed   0
        pos     speed   50 000
        pos abs 0
        wait    rdy
        return


;*****************************************************************
;*
;*      Wave the spoon and look silly...
;*
;*****************************************************************

wavef:  Acc      50             ; Just 80 turns forward.
        Ret      10
        Pos Speed  50000        ; This is 600 RPM.
        POS ABS 400000
        Wait    RDY

        clr     tmr
        wait    tmr > 1500      ; Pause...
```

```
        Acc       100             ; Then 80 turns back again.
        Ret       500
        POS ABS   0
        Wait      RDY
        CLR TMR
        Wait   TMR >  1000        ; Pause again.

        Acc       1500            ; 50 fast movements to impress audience.
        Ret       1500
        POS ABS   6250            ; To start postion...
        Wait   RDY
        CLR TMR
        Wait   TMR >  1000        ; 1 sec wait.
        LET       R1, 350         ; Inital time between movements.
        LET       R0, 50          ; Number of movements.

wavef1: POS REL   2500            ; 1/2 turn forward.
        CLR TMR
        wait      rdy             ; (to make sure it is not too fast)
        Wait   TMR >  R1          ; Wait
        SUB       R1, 5           ; Decrement wait time.

        Loop      R0,wavef1       ; Do it again.... 50 times.....

        wait      tmr > 1000      ; Puh!!!

        Acc       50              ; New parameters, for silly wave....
        Ret       50

wavef2: Push POS                  ; Get current position.
        Pop       R0
        DIV2      R0              ; Div by 8.
        DIV2      R0
        DIV2      R0

        CLR TMR                   ; 100 ms wait...
        Wait   TMR >  100
        ABS       R0,-1           ; Make r0 negative.
        POS REL   R0              ; Go 1/8 of distance from current pos
                                  ; towards 0.
        Wait   RDY
        If 0 >    R0,wavef2       ; If more to go..

        POS ABS   0               ; Then the last movement....
        wait      rdy
        return

;*****************************************************************
;*
;*      Forward FIRE with SERVE.
;*
;*****************************************************************

ffire:  acc       20              ; Backup to serve position.
        ret       20
        pos rel -130
        wait rdy

        Acc       200             ; Setup...
        Ret       250
        Pos Speed  30000
        CLR TMR                   ; Reference time.
        POS REL 500               ; Throw peanut into air.
        Wait      RDY

        Acc       1000            ; Setup for SMASH!
        Ret       1500
        Pos Speed  35000          ; This is the soft smash.
```

```
        POS ABS -1000           ; Backup....
        Wait  RDY

        Wait  TMR >  380        ; Wait until time to SMASH!
        POS ABS  4000           ; SMASH!!!!!
        Wait  RDY
        Return                  ; This is it!

;*****************************************************************
;*
;*      Backward FIRE with ZAPPING SMASH!
;*
;*****************************************************************

bfire:  acc     20              ; Goto serve position.
        ret     20
        pos rel 75
        wait rdy

        acc     200             ; Parameters for "throw into the air"
        ret     250
        pos speed 30 000

        clr     tmr             ; Ref-Time
        pos rel 500             ; Serve
        wait rdy

        acc     1000            ; Parameters for SUPER SMASH!
        ret     1500
        pos speed 100 000       ; This is 1200 RPM.

        wait    tmr > 350       ; Ready... Steady... GO!!!!
        pos abs -12000          ; SMAAAASHHHHH!
        wait    rdy
        return                  ; So, the world has one broken peanut more.

;*****************************************************************
;*
;*      Lazy throw away.... (forward)
;*
;*****************************************************************

fthrow: Acc     500             ; Good parameter for simple throw away.
        Ret     250
        Pos Speed  30000
        POS REL  5000           ; Go exactly one turn.
        Wait  RDY
        Return                  ; Then ready.

;*****************************************************************
;*
;*      eXtra Lazy throw away.... (forward)
;*
;*****************************************************************

xthrow: Acc     50      ; Good parameter for simple throw away.
        Ret     50
        Pos Speed  30000
        POS REL  2500           ; Go exactly 1/2 turn
        Wait  RDY
        pos rel -2500           ; and back again.
        wait rdy
        Return                  ; Then ready.
```

```
;*****************************************************************
;*
;*      Load peanut.
;*
;*****************************************************************


loadnut:
;
;       First we must check if there is any penuts in the revolver.
;
        if      0 = peanuts, nonuts     ; Error if nope.
        if      0 > peanuts, nonuts     ; Just to make sure....
        sub     peanuts, 1              ; Take a nut.
;
        SET Out  64             ; Start peanut revolvermotor.
ldn1:   If lo IN and   64,ldn1  ; Wait while sensor active
ldn2:   If hi IN and   64,ldn2  ; Wait while sensor inactive.
        CLR Out  64             ; Got active, stop motor.
        CLR TMR                 ; Timer = 0
        Wait  TMR >  150        ; Wait 150 ms for peanut to begin to fall.
        SET Out  64             ; Start motor again
        Wait  TMR >  350        ; Wait 200 ms more.
        CLR Out  64             ; Stop motor.
        Wait  TMR >  500        ; Wait 150 ms more for peanut to stabilize.
        Return
;
;       No Nuts, blink with the lamps....
;
nonuts: let     r0,63
nonu1:  clr out 63
        set out r0
        clr     tmr
        wait tmr > 15
        loop    r0, nonu1
        clr out 63
        goto    wcmd            ; Wait for new command.


;*****************************************************************
;*
;*      INIT PDI -- Find home position.
;*
;*****************************************************************

initpdi:
        Ref   POS    0          ; Setup system parameters etc...
        Scale    145
        Zero     192
        Pole     0
        P Gain   13
        vector  8, fposerr      ; Setup vector for position error.
        pos err 1250            ; 1/4 turn is fatal.....
        Enable REF              ; Want to look for ref-pulse.
        Acc      50
        Ret      50
        Speed    1000           ; Slow forward.
initp1: If lo IN and 128,initp1 ; Wait util i find it.
        Ref   POS    700        ; Yep, this is pos 700 rel to launch pos.

        Pos Speed  10000                ; Goto launch psition.
        POS ABS  0
        Wait  RDY
        Return
```

```
;*****************************************************************
;*
;*      GOHOME -- Goto launch position and setup standard ACC & RET.
;*
;*****************************************************************

gohome: Acc       100
        Ret       100
        Pos Speed 10000
        POS ABS  0                      ; go away...
        Wait  RDY
        Return

;*****************************************************************
;*
;*      JIGGLE -- Jiggle peanut in position.
;*
;*****************************************************************

jiggle: LET       R0, 40                ; Number of times to jiggle.
        LET       R1, 15000             ; Jiggle force.
        LET       R2,-15000

jloop:  Profile ACC  440                ; Jiggle forwards
        Wait  RDY
        Profile ACC -440                ; and then back again.
        Wait  RDY
        CLR TMR
        SUB       R1, 375               ; Decremnt jiggle force.
        ADD       R2, 375
        Loop      R0,jloop              ; Then ready for next jiggle.
        clr      tmr                    ; Wait 0.5 seconds more for
        wait     tmr > 500              ; panut to stablize.
        Return

440     P Data    R1,  5                ; Data for jiggle.
441     P Data    R2, 10
442     P Data    R1,  5
443     P Data    0,  0

;*****************************************************************
;*
;*      FPOSERR -- Fatal positioning error, close down shop.
;*
;*****************************************************************

fposerr:
        scale    0              ; Emergeny stop!

        pos      abort          ; Stop internal ramp gen etc...
        acc      1000
        speed    0

fposer1:
        let      status, stsidle ; Flag that we are offline.
        clr out 127
        clr      tmr
        wait     tmr > 200
        set out 63
        wait     tmr > 400
        goto     fposer1        ; Loop here until user aborts...
```

```
;*****************************************************************
;*
;*      Control-C trap, go here when user types Control-C.
;*
;*****************************************************************

ctrlc:  vector   10,ctrlc       ; Setup Control-C trap again.
        speed    0
        acc      500
        pos      abort          ; This stops main motor gracefully
        Out       0             ; This makes sure revolver motor is off.
        let      status, stsidle
        STOP                    ; Then we can stop program.
        Goto      restart


;*****************************************************************
;*
;*      RANDOM BLINK ROUTINE.
;*
;*****************************************************************

rblink: push     tmr
        pop      r0
        sub      r0,200
        if       0 > r0, rblret

        add      r10, 13611     ; This is a 16 bit random generator.
        mul      r10, 271
        and      r10, 65535     ; Do only keep the 16 lsb's
        let      r11, r10
        div2     r11
        div2     r11
        div2     r11
        div2     r11
        and      r11, 63        ; Want to random blink with 6 LED's
        clr out 63
        set out r11
        let      r12,r10
        and      r12,127
        add      r12,50
        dec      tmr r12        ; Decrment the timer.

rblret: return                  ; No change, return.
;
;       SLUT!!!!
;
=================================================================
```

A boy scout was out doing his bob-a-job stint one Saturday in Farmborough. He walked up to the front door of one house and rang the doorbell. The owner appeared.
"Yes?"
"Bob a job week, sir!"
At first the man didn't want anything to do with the kid, but eventually he agreed to give him a job. "You can paint my porch for me", he said. "The paint and brushes are in the garage—here's the key." The boy scout toddled off to do the job. Two hours later, he rang the doorbell.
"Job's done", he said, his palm outstretched.
"Harumph. Took your time, didn't you ? Well, okay, here's your FIVE PENCE!"
The boy took the money and started to walk away, but after a few paces he turned around and said, "Oh, thanks for the donation, but by the way, it wasn't a porch, it was a Ferrari!"

Relationsdatabaser

av Mats O Jansson

EFTER ett månadsmöte i våras stod jag och pratade om relationsdatabasen 4th Dimension som jag har på min MacIntosh. Någon ville då veta lite mer om ämnet och jag blev lite förvånad. Anledningen till att jag blev förvånad var nog att jag inte varit med i något projekt på över fyra år som *inte* använt relationsdatabaser. Jag började berätta om grunderna och lovade att skriva en artikel om det i StackPointer. Den är den du nu håller på att läsa. Det jag skriver i den här artikeln är ingen absolut sanning utan snarare verkligheten enligt mig.

Tabell, kolumn och nyckel.

Vad är då en relationsdatabas? Jo, en relationsdatabas består av en eller flera tabeller som innehåller en eller flera kolumner med information. Varje rad i en tabell måste kunnas nås av en nyckel. Nyckel kan bestå av en eller flera kolumner.

Man kan jämföra en tabell med en indexerad fil (ISAM-fil) och kolumnerna med de olika elementen i posterna i filen. Där slutar dock likheterna, för i en relationsdatabas är kolumnerna namnsatta och man har även berättat vilken typ av data kolumnerna innehåller. Men precis som med en indexerad fil så vet man vad som är nyckel.

Operationer.

Det finns ett antal operationer som kan göras på en tabell. Dessa är t.ex. READ, INSERT, UPDATE och DELETE. Vad dessa gör behövs väl inte förklaras. SELECT används för att begränsa vilka poster i en tabell man är intresserade av. PROJECT används för att koppla databasen mot variabler.

Frågespråk.

Alla relationsdatabaser har någon typ av frågespråk. Detta frågespråk ger en användare möjlighet att söka efter information i en databas på ett godtyckligt sätt. Det kan vara frågor av typen: Hitta alla bilar av 1987 års modell som har två dörrar och är röda och vars ägare bor i Sundbyberg. Detta kräver dock att databasen innehåller den information som man använder för att begränsa frågan.

View och join.

En view är en virtuell tabell som består av hela eller delar av en eller flera tabeller. Om det är flera tabeller är dessa sammankopplade genom en join. Joinen beskriver vilka fält som skall utgöra kopplingen mellan tabellerna.

Transaktionshantering.

Ett problem med databaser är att man måste försäkra sig om att databasen alltid är konsistent, dvs man vill inte att innehållet i en tabell refererar till en post i en annan tabell som inte finns. Detta gör att man infört ett transaktionsbegrepp.

Tanken är att allt man vill göra mot databasen görs inom en transaktion. Genom att påbörja en transaktion så håller databasen reda på vilka poster som lästs, uppdaterats, lagts till och tagits bort. När man sedan är färdig så avslutas transaktionen. Man kan i princip avsluta transaktionen på två sätt, COMMIT och ROLLBACK. COMMIT är nog den vanligaste och innebär att man utför de ändringar man gjorde inom transaktionen. ROLLBACK betyder att man inte vill utföra transaktionen utan backar ut.

Hur vet man nu att ingen har varit på någon av de poster jag har inom min transaktion? Normalt sker detta genom att man låser de poster man använder sig av. Detta leder till att man kan råka ut för DEADLOCKs. Detta är något som då databasen på något sätt måste försöka att lösa upp. En annan metod är den som en stor svensk databas använde sig av Optimistic Concurrency Control.

Optimistic Concurrency Control bygger på att man antar att antalet konflikter är litet, och att man därför inte behöva låsa posterna. I och med att man inte låser posterna måste man utföra ett större arbete vid COMMIT för att avgöra om allt är i sin ordning. Först kontrollerar man om det gjorts någon COMMIT, som inte bara innehåller läsningar, sedan transaktionen påbörjades. Om så är fallet kontrollerar man om någon av de poster man läst har förändrats. Om inte så kan man nu uppdatera databasen med sina förändringar. Om något hade förändrats så har transaktionen misslyckats.

Normalisering.

Ett annat viktigt begrepp är normalisering. I och med att databaser ofta är stora vill man undvika att dubbel lagra information. Hur långt man har gått beskrivs ofta genom på vilken normalform (NF) databasen befinner sig.

Nedan kommer ett exempel på hur man transformerat en tabell på första normalformen till fyra på tredje normalformen. Alla nycklar är utmärkta med en asterisk.

Första normalformen.

Första normalformen betyder att alla poster i en tabell måste kunna nås av en unik nyckel.

TILLVERKNING (1NF)

fid*	pid*	kvantitet	fadress	flän	pkat
F1	P2	30	SMÅSTAD	F	A
F2	P1	65	SMÅSTAD	F	B
F2	P3	35	SMÅSTAD	F	B
F3	P1	75	STORSTAD	AB	B
F3	P2	55	STORSTAD	AB	A
F3	P3	50	STORSTAD	AB	B
F3	P4	90	STORSTAD	AB	C

Andra normalformen.

Om tabellen är på första normalform och man vill upp till andra så måste man se till att alla kolumner måste vara beroende av *hela* nyckeln.

TILLVERKNING (2NF)

fid*	pid*	kvantitet
F1	P2	30
F2	P1	65
F2	P3	35
F3	P1	75
F3	P2	55
F3	P3	50
F3	P4	90

PRODUKTER (2NF)

pid*	pkat
P1	B
P2	A
P3	B
P4	C

Tredje normalformen.

Om tabellen är på andra normalform och man vill upp till tredje så måste man se till att alla kolumner måste enbart vara beroende av nyckeln

FÖRETAG (2NF)

fid*	fadress	flän
F1	SMÅSTAD	F
F2	SMÅSTAD	F
F3	STORSTAD	AB

Fjärde normalformen.

Om tabellen är på tredje normalform och man vill upp till fjärde så måste man se till att alla endast en förekomst av varje kolumn, inga tabeller i tabeller.

TILLVERKNING (3NF)

fid*	pid*	kvantitet
F1	P2	30
F2	P1	65
F2	P3	35
F3	P1	75
F3	P2	55
F3	P3	50
F3	P4	90

PRODUKTER (3NF)

pid*	pkat
P1	B
P2	A
P3	B
P4	C

FÖRETAG (3NF)

fid*	fadress
F1	SMÅSTAD
F2	SMÅSTAD
F3	STORSTAD

ADRESSER (3NF)

adr*	län
SMÅSTAD	F
STORSTAD	AB

Hur långt skall man gå? Är det önksvärt att man kommer så långt som till fjärde normalform i sin databas? Det är nog en smaksak om man går så långt. Ofta är det nog så att man försöker få så stora delar som möjligt att ha en så hög normalform som möjligt. Att dela upp ett postnummer i en adress så att bara postnummret och inte orten finns i adressen, och en tabell med postnummer och ort är nog inte många som gör. Därmed skulle en tabell med adresser inte kunna komma längre än till andra normalformen.

Codd´s 12 Rules for Relational DBMS

hese rules outlines a technique that should help users determine how relational a DBMS really is.

Rule Zero:
For any system that is advertised as, or claimed to be, a relational data base management system, that system must be able to manage databases entirely through its relational capabilities.

The Information rule.
Rule 1:
All information in a relational database is represented explicitly at the logical level and in exactly one way - by values in tables.

Guaranteed access rule.
Rule 2:
Each and every datum (atomic value) in a relational data base is guaranteed to be logically accessible by resorting to a combination of table name, primary key value and column name.

Systematic treatment of null values.
Rule 3:
Null values (distinct from the empty character string or a string of blank characters and distinct from zero or any other number) are supported in fully relational DBMS for representing missing information and inapplicable information in a systematic way, independent of data type.

Dynamic on-line catalog based on the relational model.
Rule 4:
The data base description is represented at the logical level in the same way as ordinary data, so that authorized users can apply the same relational language to its interrogation as they apply to the regular data.

Comprehensive data sublanguage rule.
Rule 5:
A relational system may support several languages and various modes of terminal use (for example, the fill-in-the-blanks mode). However, there must be at least one language whose statements are expressible, per some well-defined syntax, as character strings and that is comprehensive in supporting all of the following items:

- Data definition.
- View definition.
- Data manipulation (interactive and by program).
- Integrity constraints.

- Authorization
- Transaction boundaries (begin, commit and rollback).

View updating rule.
Rule 6:
All views that are theoretically updatable are also updatable by the system.

High-level insert, update and delete.
Rule 7:
The capability of handling a base relation or a derived relation as a single operand applies not only to the retrieval of data but also to the insertion, update and deletion of data.

Physical data independence.
Rule 8:
Application programs and terminal activities remain logically unimpaired whenever any changes are made in either storage representations or access methods.

Logical data independence.
Rule 9:
Application programs and terminal activities remain logically unimpaired when information - preserving changes of any kind that theoretically permit unimpairment are made to the base tables.

Integrity independence.
Rule 10:
Integrity constraints specific to a particular relational data base must be definable in the relational data sublanguage and storable in the catalog, not in the application programs.

Distribution independence.
Rule 11:
A relational DBMS has distribution independence.

Nonsubversion rule.
Rule 12:
If a relational system has a low-level (single-record-at-a-time) language, that low level cannot be used to subvert or bypass the integrity rules and constraints expressed in the higher level relational language (multiple-record-at-a-time).

Jingle bells,
Santa smells,
Of reindeer shit and hay.
He had a fight,
With the Elves one night,
And Rudolph wants more pay!

Mac@KTH

Protokoll fört vid arbetslunch 88 12 02

Närvarande:	Peter Graham	(6515) graham@nada.kth.se
	Lennart Johansson	(8471) lelle@lnm.kth.se
	Jan Michael Rynning	(6288) jmr@nada.kth.se
	Mikael Magnusson	(9104) mima@nada.kth.se
	Jesper Lindström	d85-jli@nada.kth.se
	Jan-Erik Mångs	d85-jem@nada.kth.se
	Peter Svanberg	(7146) psv@nada.kth.se
	Kjell Krona	krona@tds.kth.se
	Hans Nordström	(08-797 80 00)sib@stacken.kth.se
	Roswitha Graham	(6525) roswitha@admin.kth.se

- *Interrimsstyrelse tillsatt.* Arbetsgruppen fattade beslut om att fungera som en interimsstyrelse till nästa möte då vi väljer styrelse och konstituterar oss. Till ordförande i interimsstyrelsen valdes Peter Graham. Vi fastslog att i första hand grunda en ideell förening utan medlemsavgifter, öppen för KTH-anställda och studerande. Medlemskap erhålls genom att närvara vid aktivitet och anteckna sig.

- *Arbetsuppgifter för användarföreningen inom den närmaste tiden:* Seminarieverksamhet — program för våren (Roswitha lovade samordna), kontakter med Apple (Peter Graham), stadgar, databas över medlemmar (Peter Svanberg åtog sig att ansvara och lägga upp register, ev i Ordo), elektronisk brevlåda för föreningen (Peter Sv & Peter Gr får fundera på förslag var och hur man lämpligen ordnar), MacKaffe/Te en gång i månaden för att träffas kring allmänna frågor (bestämdes till seminarierum Drottning kristinas väg 31), ordna med tillgång till rum, tidskrifter, gratis programvara för Mac (bör samordnas med programvarubiblioteksprojektet).

- *Stadgar* behöver förberedas på lämpligt sätt. JMR hade skaffat fram Stackens stadgar samt underlag från naturvetenskapliga föreningen på US, Lelle hade tagit med verksamhetsberättelsen från Royal Mac där man inte formaliserat sig ifråga om stadgar. Kjell Krona kunde berätta om Mac in Lund och menade att deras stadgar hade fungerat vettigt så när som ifråga om några små punkter som man borde ändra. Kjell skickar till Roswitha och vi kan börja med att utgå från Lunds stadgar.

Förslag har lagts att föreningen arbetar i olika aktivitetsgrupper. D-linjen med Jan-Erik och Jesper är villiga att engagera sig i programmering, Kjell från arkitektur kunde tänka sig vara engagerad i Hypercardgrupp (med förbehåll att det var mindre lyckat om gruppen därmed skulle fungera som konsultansvariga på KTH, grönt däremot med något användarseminarium). Ur aktivitetsgrupperna skulle man kunna utkristallisera lämpliga personer som vid behov kan anlitas för arbetsuppgifter inom resp område. Lelle menade att ersättning mycket väl kan utgå från institution för sådana uppdrag.

Program för våren: Seminarier i större hörsalar (typ E7) första torsdagen i varje läsperiod med början torsdagen den 19 januari kl 15. Programförslag var OCR-program ochg utrustning. Vi kontaktar distributör för program som Staffan Romberger tidigare tagit upp som intressant. Dessutom lite allmän genomgång av lämplig utrustning och erfarenheter.— Därefter paus och formell konstitutering av förening och styrelse. Kallelse utgår till alla institutioner (Roswitha ordnar). På motsvarande sätt ordnas den 3:e torsdagen kl 15 i varje ny läsperiod en enklare Mac-träff kring program, nyheter och användarfrågor, förslagvis Drottning Kristinas väg 31 (MacKaffe).

Scanner och mera högupplösande laserutskrift hör till det som man skulle vilja ha mera allmänt tillgänglig p KTH — något att tänka på i diskusioner med Apple om utrustning som de lånar ut. Traditionellt brukar de ställa upp med en MacSE och Imagewriter samt återhållsamt tillgång till MacLink för senaste nytt och kontakt med Apple.

Nästa möte som vi hoppas att alla bokar in: 19 januari kl 15 i E7 om inget annat meddelas

Stockholm 88 12 02

Roswitha Graham

Roswitha Graham

Jan Michael Rynning

Jan Michael Rynning

BBS för Macdiggare

018	-30 93 85	Boa the Board	gratis
0470	-221 83	SIX	gratis
0640	-109 29	MacUnderground	gratis demo–485:-/år
0753	-233 31	MacSoftLine	gratis demo–500:-/år
08	-665 34 64	MacUnderground	gratis demo–485:-/år

Välkommen
till
STACKENS Gym

Varje torsdag framöver.

Nya övningar utarbetas för varje gång.

WORKOUT

Färgstretch
Datorpush
Plattlyft
Kylpress
Kabelskruv
Diskmover

Skynda på och deltag. Gymet finns inte hur länge som helst. Det skall bli datorhall där.

Avgiften för HELA 1989 är för studerande endast SEK 89 och för övriga SEK 189. Passa på och få lite fysisk övning.