Forth

Radek Hnilica

Book@Hnilica.CZ

Forth

Radek Hnilica

$Header: /home/radek/cvs/forth-book/forth.xml,v 1.31 2005/10/20 19:19:37 radek Exp $ Vydání

Poznámky k programovacímu jazyku „forth“.

Tento dokument pojednává o programování v programovacím jazyku Forth. A to jak obecně tak se zaměřením na PalmOS/Quartus Forth, Osmibitové počítače Atari a další.

Tento dokument je k dispozici v několika různých formátech. Jako vícestránkový HTML dokument (index.html), postscriptový (forth.ps) či PDF (forth.pdf) soubor formátovaný na velikost papíru A4.

Počet stran v Postscriptové a PDF verzi: 417 .

Postscriptová verze má základní velikost písma 12pt.

Radek Hnilica <Book@Hnilica.CZ>

Přehled revizí

Revize 0 2002-02-21
Nultá pracovní revize.
Revize 0.1 2002-02-27
Pracovní revize.
Revize 0.2 2002-03-04
Pracovní revize.
Revize 0.3 2002-03-10
Reorganizace kapitol do částí, zahájena práce na části PPForth.
Revize 0.4 2002-03-17
Další reorganizace, analýza QF, další slova a instrukce MC68k
Revize 0.5 2002-04-01
Drobné úpravy a přidány další slova do slovníku Qurtus Forthu a instrukce MC68k
Revize 0.6 2002-06-25
Drobné úpravy, vytvořena sekce "Databáze"
Revize 0.7 2003-12-27
Zahájeno znovuvytvoření knihy od úplného začátku.
Revize 0.8 2003-12-29
Pracovní výtisk.

Věnování tag dedication/title

tag dedication/para

Tuto knihu věnuji tobě.

FIXME: napsat věnování.

Obsah

Seznam tabulek

FIXME:colophon/title

FIXME:colophon/title

FIXME:colophon/para

Předmluva

* *rcsinfo="$Id: preface.xml,v 1.2 2005/03/02 11:43:44 radek Exp $"*
* ***FIXME:****předmluva pojednávající o okolnostech sepsání tohoto spisku.*

FIXME:

Kapitola 1. Úvod

* *chapter id="introduction" xreflabel="Úvod"*
* *rcsinfo="$Id: ch-introduction.xml,v 1.7 2005/05/03 08:05:01 radek Exp $"*

Všechno jednou skončí.

Já

* *Povídání o věcech kolem Forthu a rozčlenění tohoto spisku.*

Forth je jedním z nejstarších programovacích jazyků. Dle mých znalostí je taky nejjednodušším jazykem vůbec.

Lidé kolem Forthu

Chuck More

Charles H. Moore *The Inventor.*

1.1. Historie jazyka

FIXME: *Všechno jednou začíná.*

Zdroje a odkazy:

- The Evolution of Forth (http://www.forth.com/Content/History/History1.htm)
- The Evolution of Forth (http://burks.boton.ac.uk/burks/language/forth/hist4th/history.html)
- Introduction to FORTH (http://jpb.forth.free.fr/Anglais/introduction.html)

Programovací jazyk Forth vynalezl Charles H. Moore (Chuck Moore). Vyrostl/vyvinul se z Moorovy práce v 60-tých letech. První program nesoucí jméno forth byl napsán okolo roku 1970. Původně se měl jmenovat fourth (čtvrtý) ale protože počítač na kterém byl poprve napsán umožňoval pojmenovat soubory jen nejvýše pěti znaky, byl pojmenován Forth

Když Charles Moor pracoval v *National Look-out post of Radio astronomy of the United States* na začátku 70-tých let.

1.1.1. Počátky

První program s názvem Forth napsal Moore v roce 1970. Byl výsledkem jeho práce v 60-tých letech.

.

1.1.2. Poznámky

Working in the National Look-out post of Radio astronomy of the United States at the beginning of the 70s, Charles Moore was charged to schedule the first minicomputers 16 bits (typify IBM 360) for the purchase of scientific data and the maintenance of equipment.

The high-level languages of time as the FORTRAN (not to confuse with the object of this paragraph) was too heavy for the execution of real-time software packages. Machine language was, as for him, indigestible enough to venture to realize always more complex programs.

And so Charles MOORE had the idea to create a revolutionary language the instructions of bases of which corresponded to a single line of code in machine language.

This allowed not only to write programs in high-level language with almost the performances of machine language but besides reducing at most size memory which was extremely weak at the time.

Charles MOORE wanted to record his language under the name of FOURTH for language of fourth generation but the computer which he used authorizing only names of 5 letters, he called it: FORTH.

Well to understand and to use at most performances inferred by the language FORTH, it is important to master inverted Polish notation used also with calculators made by Hewlett-Packard. The transmission of parameters makes essentially by a last in-first out stack. This notion is what allowed Charles Moore to pull the maximum of performances of its language because it is necessary to know that any processor deserving of this name possesses memory manipulation instructions under shape of stack of this type.

Besides the classic stack of return from subroutines administered intrinsically with most of the processors, the idea to use the same instructions for parameters transfers drove to create a data stack of which some basic instructions are the following ones:

- DUP piles the copy of the number being at the top of the stack: n - > n, n (the summit of the stack is to the right),
- DROP depilates the number being at the top of the stack: n - > - (line means that the stack is empty with regard to the previous level),
- SWAP inverts the 2 numbers of the summit of the stack: n1 , n2 - > n2 , n1
- OVER piles the copy of the number situated in the second rank of the summit of the stack: n1 , n2 - > n1 , n2 , n1
- ...

Operations such as comparisons or additions will directly consume the numbers situated at the top of the stack:

- > Compare the 2 numbers at the top of the stack and leave a boolean: n1 , n2 - >-1 if n1 > n2 , 0 otherwise
- - Complete subtraction such as: n1 , n2 - > n1-n2
- * Signed complete reproduction such as: n1 , n2 - > n1*n2
- ...

The most control basic structure is the set formed with IF, ELSE AND THEN. IF tests the number at the top of the stack by pulling it. If it is different from 0, instructions placed between IF and ELSE are executed then program connects following the word THEN. If it is zero, program connects directly in instructions being following the word ELSE. Several structures of this type can be obviously been linked.

In FORTH, all the instructions can be programs and mutually. They are introduced into the memory according to their compilation. Put in by some instructions which manipulate the return stack as the structures of control for example, all the instructions can be interpreted at any time what gives considerable opportunities of settling for developers. Useless to write another main program and to redo a compilation to see if it is subroutine which works badly . It is enough to launch his execution with adequate parameters.

Instructions or programs or subroutines are inserted into the memory with pointers' system which connect them among them to allow the interpreter (or in the compiler according to the mode of functioning) to find them. Search begins by leaving of the last compiled instruction and by raising until the first which is often DUP. If instruction does not exist, the interpreter / compiler looks if it is a number and piles it at the top of the stack in

interpretation mode or compiles it in the instruction in compilation mode. To create an instruction, one uses 2 instructions (except those that describe the sequence to be executed):

- : Follow-up of the name of the instruction and which spends from interpretation to compilation mode,
- ; Which ends instruction and so goes back in interpretation mode.

Finally, not to be too exhaustive in this presentation, know that FORTH language allows to write recurse procedures what makes a really very evolved language of it. Here is the example of factorial calculation program of an integer appealing to this notion of recursion:

```
: FACTORIELLE
 DUP 1 >
 IF
  DUP 1 - FACTORIELLE *
 THEN
;
```

1.2. Zdroje, literatura odkazy

Na internetu je mnoho zajímavých zdrojů. Některé nejsou aktuální, úspěšně se rozvíjí, jiné zanikají. Zde se pokusím zmapovat alespoň některé z nich. Nejdříve se podíváme na zdroje v jazyce českém či slovenském. Na rootu (http://www.root.cz) vyšla série článků od Pavla Tišnovského (http://www.root.cz/autori/pavel-tisnovsky/), Programovací jazyk Forth (http://www.root.cz/serialy/programovaci-jazyk-forth/).

Seznam článků série: Programovací jazyk Forth ROZPRACOVÁNO

1. Programovací jazyk Forth a zásobníkové procesory (http://www.root.cz/clanky/programovaci-jazyk-forth-a-zasobnikove-procesory/) vyšlo 2005-01-11
2. Programovací jazyk Forth a zásobníkové procesory (2) (http://www.root.cz/clanky/programovaci-jazyk-forth-a-zasobnikove-procesory-2/) vyšlo 2005-01-18
3. Programovací jazyk Forth a zásobníkové procesory (3) (http://www.root.cz/clanky/programovaci-jazyk-forth-a-zasobnikove-procesory-3/) vyšlo 2005-01-25
4. Programovací jazyk Forth a zásobníkové procesory (4) (http://www.root.cz/clanky/programovaci-jazyk-forth-a-zasobnikove-procesory-4/) vyšlo 2005-02-01
5. Programovací jazyk Forth a zásobníkové procesory (5) (http://www.root.cz/clanky/programovaci-jazyk-forth-a-zasobnikove-procesory-5/) vyšlo 2005-02-08
6. Programovací jazyk Forth a zásobníkové procesory (6) (http://www.root.cz/clanky/programovaci-jazyk-forth-a-zasobnikove-procesory-6/) vyšlo 2005-02-15
7. Programovací jazyk Forth a zásobníkové procesory (7) (http://www.root.cz/clanky/programovaci-jazyk-forth-a-zasobnikove-procesory-7/) vyšlo 2005-02-22
8. Programovací jazyk Forth a zásobníkové procesory (8) (http://www.root.cz/clanky/programovaci-jazyk-forth-a-zasobnikove-procesory-8/) vyšlo 2005-03-01
9. Programovací jazyk Forth a zásobníkové procesory (9) (http://www.root.cz/clanky/programovaci-jazyk-forth-a-zasobnikove-procesory-9/) vyšlo 2005-03-08
10. Programovací jazyk Forth a zásobníkové procesory (10) (http://www.root.cz/clanky/programovaci-jazyk-forth-a-zasobnikove-procesory-10/) vyšlo 2005-03-16

Dalším zdrojem jsou historické materiály Rudolfa Pecinovského (http://rudolf.pecinovsky.cz/texty/index.htm). Jedná se o seriál Forth publikovaný v časopise Amatérské rádio v letech 1984 (http://rudolf.pecinovsky.cz/texty/Forth_AR_84-85_str_01-12.pdf) a 1985

(http://rudolf.pecinovsky.cz/texty/Forth_AR_84-85_str_13-18.pdf). Pan Pecinovský ještě zmiňuje učebnici programování v jazyku Forth602. Bohužel tuto knihu již nemá (kontrolováno 2005-03-02 na jeho stránce (http://rudolf.pecinovsky.cz/texty/index.htm)). Poslední na co jsem narazil je prezentace Petr Erbena a Petra Koňaříka: FORTH (http://web.quick.cz/kopato/forth/). Asi jsem málo hledal, protože se mi to zdá velmi málo, ale nic dalšího jsem nenašel.

* *Stránky 4TH (http://www.volny.cz/gccomp/) které byly naposled aktualizovány 2000-12-12 a jsou velmi chudé a Forth zmiňují jen okrajově..*

Algicky psaných stránek je již podstatně více.

- comp.lang.forth.repository (http://forth.sourceforge.net/)

Wiki a komunity weby

- #forth (http://forth.bespin.org)

Forth OS

- Forth OS (http://www.forthos.org/)

Tutoriálny

- Forth primer (http://www.xs4all.nl/~thebeez/ForthPrimer/Forth_primer.html)
- . ()

Odkazy a katalogy odkazů

- Forth Research at Institut für Computersprachen (http://www.complang.tuwien.ac.at/projects/forth.html)
- Programming Resources (http://www.stejskal.de/web/computer/forth/_programmers_info.html)

Historie

- The Genesis of Postscript (1981) (http://www.geocities.com/jim_bowery/psgenesis.html)
- . ()
- . ()
- . ()

1.3. Dostupné implementace

Pokusím se čtenáře obeznámit s běžně dostupnými implementacemi jazyka. Z řadou z nich nemám osobní zkušenosti a proto jen zmiňuji jejich existenci. V žádném případě neprovádím srovnání ani hodnocení. Pokud u některého uvedu poznámku, jedná se vždy o záležitos osobní nebo z nějakého důvodu specifickou.

1.3.1. Volné, Open Source implementace

FIXME:

- gForth — je k dispozici v debianu v balíčku `gforth`
- kForth — taktéž v Debianu v balíčku `kforth`
- pForth — taktéž v Debianu v balíčku `kforth`
- sjforth — taktéž v Debianu v balíčku `sjforth`
- yforth — taktéž v Debianu v balíčku `yforth`

gForth

FIXME:

1.3.1.1. kForth

kForth je Open source inplementace od Creative Consulting for Research and Education (http://www.ccreweb.org). Je postavena na modelu ITC.

1.3.2. Implementace pro 8-mi a 16-ti bitové procesory

FIXME:

- Forth pro 8-mi bitové mikropočítače a některé 16/32-bitové (68k, ARM) (http://www.strotmann.de/twiki/bin/view/APG/LangForth)

Kapitola 2. Forth hardware

* *chapter id="forth-hardware" xreflabel="Forth hardware"*
* *rcsinfo="$Header: /home/radek/cvs/forth-book/ch-forth_hw.xml,v 1.2 2005/10/20 05:33:42 radek Exp $"*

Odkazy a zdroje:

- . ()
- . ()

ToDo

1. První úkol.

Technika optimalizovaná nebo přímo konstruovaná pro běh Forthu.

Protože zde není žádný obsah, kapitolu jsem zatím jen otevřel, uvádím odkazy které se k tématu vztahují a chtěl bych je zapracovat.

Odkazy a zdroje:

- Proposal for a 64-bit NOSC Forth CPU (http://www.forthos.org/64-bit.txt)

I. Tutoriál

Učíme se programovat ve Forthu.

V této části se budu učit programovat ve Forthu pěkně od základů.

Kapitola 3. Tutorial

* *chapter id="tutorial" xreflabel="Tutoriál"*
* *rcsinfo="$Header: /home/radek/cvs/forth-book/ch-tutorial.xml,v 1.6 2005/10/20 05:33:42 radek Exp $"*

Odkazy a zdroje:

- Conditional branching (http://www.geocities.com/matteo_vitturi/english/spectraforth7.htm)
- ___ (http://___)

ToDo

1. První úkol.

3.1. Úvod

* *rcsinfo="$Header: /home/radek/cvs/forth-book/ch-tutorial.xml,v 1.6 2005/10/20 05:33:42 radek Exp $"*

Forth je jazyk s velmi jednoduchou syntaxí. Rozeznává jen dva druhy prvků. Slova a znaky slova oddělující. Program je posloupnost slov. Slova mohou obsahovat všechny tisknutelné znaky. Následující znaky se mohou vyskatovat ve slovech:

```
! " # $ % & ’ ( ) * + , - . / <digits> : ; < = > ? @ <ALPHA> [ \ ] ^ _ ‘ <alpha> { | } ~
```

kde označení `<digits>` znamená jakoukoliv číslici 0-9, označení <ALPHA> znamená veliká písmena anglické abecedy a <alpha> malá písmena anglické abecedy.

Příklad 3-1. Příklady slov

```
@   !   ."   dump   <R   +   mirror-table   cell+
```

Ale jak jsem již uvedl, můžeme použít jakékoliv tisknutelné znaky, pokud to konkrétní implemetace forthu dovoluje. Například znaky s diakritikou

```
áčd’ééíll’ňóörřšt’úůýž ÁČĎÉĚÍĽL’ŇÓÖŘŘŠŤÚŮÝŽ
```

Použitím těchto znaků se však můžeme dostat do problémů při přenosu programu/aplikace na jinou platformu či při použití jiného forthu.

```
# $Id: TUT.uvod2.ses,v 1.1 2003/12/31 00:08:29 radek Exp $
Gforth 0.6.2, Copyright (C) 1995-2003 Free Software Foundation, Inc.
Gforth comes with ABSOLUTELY NO WARRANTY; for details type ‘license’
: šiši 15 ;  ok
: ö 23 ;  ok
šiši ö  ok
. . 23 15  ok
BYE
```

Poznámka: Velikost písmen nehraje roli. To znamená že například slova

```
Tak   TAK   tak   taK
```

jsou stejná. Ovšem neznamená to, že neexistuje implementace vekteré by velikost písmen byla důležitá a zmíněná slova nebyla odlišná. Konzultujte s manuálem ke své implementaci jazyka Forth.

Poznámka: Znaky které v jiných jazycích oddělují slova mohou být ve Forthu součástí slov. Neslouží tedy jako oddělovače. Například

```
,    ;    +    :    .    "
```

Program ve Forthu je [posloupnost] slov oddělená bílými znaky (mezerou, tabulátorem a znakem nového řádku). Slovo je pak [posloupnost/řetězec] libovolných tisknutelných znaků, například písmen, číslic, interpunkčních znaků.

Slova která používáme se nacházejí ve slovníku.

Nová slova definujeme pomocí slov již známých.

Hlavní datové/paměťové struktury se kterými pracujeme jsou

- zásobník — zde jsou uložena pracovní data
- zásobník návratových adres — **FIXME:**
- slovník — zde jsou uloženy všechy definice slov
- volná paměť — ve forthu máme možnost pracovat s jakoukoliv částí paměti počítače

Další povídání tedy bude o slovech.

3.2. Zásobník dat

* *rcsinfo="$Header: /home/radek/cvs/forth-book/ch-tutorial.xml,v 1.6 2005/10/20 05:33:42 radek Exp $"*

Abychom pochopili, jak pracuje forth s daty musíme si nejdříve něco povědět o zásobníku dat. Zásobník je struktura v paměti ke které se přistupuju jen z jednoho konce. Na zásobník se můžeme dívat jako na sloupeček čísel. Nová pokládáme na vršek a chceme-li nějaká odstranit musíme postupně odebírat shora.

Povídání o kořenech forthu, zásobníku a RPN.

Zásobník je základí paměťová struktura forthu. Kolem něj se všechno točí, čísla se ukládají na zásobník, návratové adresy podprogramů se ukládají na zásobník návratových adres a slovník má také charakter zásobníku.

Zásobník je paměťová struktura FIFO, tedy poslední objekt který na vrchol zásobníku uložíme, odebíráme jako první. Zásobník si můžeme představit jako stožek listů. Vždy když chceme do zásobníku něco uložit, napíšeme to na volný list papíru a položíme na vrchol našeho stožku, zásobníku. Ze zásobníku můžeme odebírat listy jedině shora a v pořadí.

Zásobník tedy implementuje dvě základní operace

ulož číslo

: která složí k uložení čísla na vrchol zásobníku

odeber

: která slouží k odebrání čísla z vrcholu zásobníku

Pro odkazování se na hodnoty v zásobníku se používají tyto zkratky

TOS

Top of Stack -- Vrchol zásobníku, číslo na vrcholu zásobníku.

NOS

Next on Stack -- Následující pod vrcholem zásobníku.

NNOS

Následující pod NOS

Jednoduše namalováno to vypadá takto

```
|       |
| volno |
|       |
+-------+
|  TOS  |  vrchol zásobníku
+-------+
|  NOS  |
+-------+
|  NNOS |
+-------+
| data  |
   ...
|       |  dno zásobníku
+-------+
```

3.3. Čísla

Mimo slova jazyka a slova definovaná programátorem bych na tomto místě zmínil ještě čísla. Tedy poté co interpret prohledá celý slovník a slovo v něm nenajde, pokusí se je interpretovat jako číslo. Jestli uspěje, uloží je na zásobník.

```
# $Id: TUT.cisla.ses,v 1.1 2003/12/31 00:08:29 radek Exp $
Gforth 0.6.2, Copyright (C) 1995-2003 Free Software Foundation, Inc.
Gforth comes with ABSOLUTELY NO WARRANTY; for details type ‘license’
1 2 3  ok
. 3  ok
BYE
```

Tato malá „zvláštnost“, kdy každé slovo je nejdříve vyhledáváno ve slovníku nám dovolí „předefinovat“ čísla. Tedy pojmenovat nově definované slovo číslem jak je vidět na ukázce

```
# $Id: TUT.cisla2.ses,v 1.1 2003/12/31 00:08:29 radek Exp $
Gforth 0.6.2, Copyright (C) 1995-2003 Free Software Foundation, Inc.
Gforth comes with ABSOLUTELY NO WARRANTY; for details type ‘license’
: 15 55 ;  ok
15  ok
. 55  ok
BYE
```

3.4. Základní aritmetika

FIXME:

Pro základní počítání máme k dispozici několik slov: +, -, * a /. Slova očekávají parametry na zásobníku, odsud je vyberou a výsledek uloží na zásobník.

```
# $Id: TUT.arit1.ses,v 1.1 2003/12/31 00:08:29 radek Exp $
Gforth 0.6.2, Copyright (C) 1995-2003 Free Software Foundation, Inc.
Gforth comes with ABSOLUTELY NO WARRANTY; for details type ‘license’
4 9 +  ok
. 13  ok
BYE
```

Všechy výše uvedené oprace očekávají na zásobníku dvě čísla.

3.5. Operace se zásobníkem

V této sekci si vysvětlíme slova DUP, ?DUP, OVER, SWAP, ROT, PICK, ROLL, NIP, TUCK, 2DROP, 2DUP, 2OVER, 2SWAP.

FIXME:

3.6. Definice nových slov

```
# $Id: TUT.definice1.ses,v 1.1 2003/12/31 00:08:29 radek Exp $
Gforth 0.6.2, Copyright (C) 1995-2003 Free Software Foundation, Inc.
Gforth comes with ABSOLUTELY NO WARRANTY; for details type ‘license’
: square DUP * ;  ok
3 square . 9  ok
BYE
```

3.7. RPN

Předem krátká ukázka

```
# $Id: tut-rpn-1.ses,v 1.1 2002/12/26 10:13:27 radek Exp $
Gforth 0.6.2, Copyright (C) 1995-2003 Free Software Foundation, Inc.
Gforth comes with ABSOLUTELY NO WARRANTY; for details type ‘license’
1 2 + . 3  ok
BYE
```

```
# $Id: tut-rpn-2.ses,v 1.1 2002/12/26 10:13:27 radek Exp $
Gforth 0.6.2, Copyright (C) 1995-2003 Free Software Foundation, Inc.
Gforth comes with ABSOLUTELY NO WARRANTY; for details type ‘license’
3 4 * . 12  ok
BYE
```

Jak jste si asi všimli u jednoduchého příkladu na sčítání, Forth používá postfixovou notaci. Ve škole jsme se učili zapisovat jednoduché sčítání takto `1 + 2`. Tedy znak pro operaci sčítání „+“ se píše mezi sčítance (čísla jenž jsou sčítána). Tomuto zápisu se říká *infixový zápis* nebo taky *infixová notace*. Ve Forthu však znak operace píšeme až za operandy. Tedy náš příklad vypadá takto: `1 2 +`. Tomuto zápisu se též říká *RPN* (*reverzní polská notace*). Je základem Forthu a proto je třeba si ji osvojit.

Výhodou postfixového zápisu je, že není třeba závorek k určení priority operací u složitějších zápisů. Výraz `3 * (2 + 4)` se zapíše jako `3 2 4 + *`

3.8. Zásobník

FIXME:

3.9. Zásobník návratových adres

* *Popsat slova: R@, R>, 2R@, 2R>, >R, 2>R*

3.10. Větvení programu

Pro podmíněné větvení můžeme použít konstrukci `IF ... THEN` případně její rozšířenou variantu `IF ... ELSE ... THEN`

```
IF (true part) THEN
IF (true part) ELSE (false part) THEN
```

Slovo IF se rozhoduje podle hodnoty na vrcholu zásobníku zdali se vykoná část kódu za ním uvedená, případně která část kódu. Přesněji pokud je na vrcholu zásobníku hodnota nenulová, je tato považována za logickou 1 tedy `true`, a je vykonán kód který následuja za IF až po THEN, nebo ELSE. Je-li na zásobníku `0` je považována za logickou 0 tedy `false`. V tomto případě se buď to nevykoná kód žádný, varianta IF...THEN, nebo se vykoná kód následující za slovem ELSE, varianta IF...ELSE...THEN. Příkaz IF je definován jen v módu překladu a můžeme jej tudíž použít jen v těle definovaného slova. Ukažme si tedy první příklad

```
# $Id: TUT.if.ses,v 1.1 2003/12/28 18:21:58 radek Exp $
Gforth 0.6.2, Copyright (C) 1995-2003 Free Software Foundation, Inc.
Gforth comes with ABSOLUTELY NO WARRANTY; for details type ‘license’
: iftest  IF ." TRUE" THEN ;  ok
0 iftest  ok
1 iftest TRUE ok
-1 iftest TRUE ok
BYE
```

Jak je vidět `TRUE` se vytiskne jen pro hodnoty `1` a `-1`, a vytiskla by se i pro všechny další nenulové hodnoty. Při zadání `0` se neudělá nic. Rozšiřme si teď větvení o ELSE

```
# $Id: TUT.if2.ses,v 1.1 2003/12/28 18:21:58 radek Exp $
Gforth 0.6.2, Copyright (C) 1995-2003 Free Software Foundation, Inc.
Gforth comes with ABSOLUTELY NO WARRANTY; for details type ‘license’
: iftest  IF ." TRUE" ELSE ." FALSE" THEN ;  ok
0 iftest FALSE ok
1 iftest TRUE ok
-1 iftest TRUE ok
BYE
```

3.11. Větvení

3.11.1. cond IF ... THEN

Nejednodušší větvení je konstrukcí `IF ... THEN`

3.11.2. cond IF ... ELSE ... THEN

*FIXME:

```
bollean_value if then_part else else_part then
```

3.11.3. CASE

```
( n ⟶ )
CASE
    0 OF ." Just a zero!" ENDOF
    1 OF ." All is ONE!" ENDOF
    2 OF WORDS ENDOF
    DUP . ." Invalid Input!"
    ( n )
ENDCASE
```

3.12. Cykly

3.12.1. Konstrukce For-Next

.

3.12.2. Konstrukce While-Do

.

3.12.3. Konstrukce Repeat-Until

.

3.12.4. Nekonečné smyčky

.

3.12.5. Kontrukce DO ... LOOP

Smyčka s předem daným počtem opakování

```
to from DO ... I ... LOOP
```

3.12.5.1. DO

*FIXME:

```
( to from ⟶ )
```

3.12.5.2. LOOP

*FIXME:

3.12.5.3. I

*FIXME:

```
( ⟶ n )
```

3.12.5.4. Příklad

```
: cyklus 5 1 do i . loop ; Enter
cyklus Enter 1 2 3 4 ok
```

3.12.6. DO ... +LOOP

*FIXME:

3.12.7. Konstrukce BEGIN ... UNTIL

* *FIXME: Popsat cyklus typu BEGIN ... UNTIL*

```
BEGIN
    kód
    podmínka
UNTIL
```

3.12.8. BEGIN ... REPEAT

```
BEGIN ... cond WHILE ... REPEAT
```

3.12.9. BEGIN ... AGAIN

*FIXME:

Kapitola 4. Forth

* *rcsinfo="$Header: /home/radek/cvs/forth-book/ch-forth.xml,v 1.7 2005/10/20 05:33:42 radek Exp $"*
* *Kapitola určena ke zrušení. Její jednotlivé části budou převedeny do jiných, relevantních kapitol.*

Odkazy:

- Russian FORTH Interest Group (http://www.forth.org.ru/)
- AI requirement for a Forth Assembler (http://home.hccnet.nl/a.w.m.van.der.horst/forthassembler.html)
- HP-48 programs written by Chris Heilman (http://chemlab.pc.maricopa.edu/hp48.html)
- Forth Objects (http://c2.com/cgi/wiki?ForthObjects)
- On Standardizing Object-Oriented Forth Extensions (http://www.complang.tuwien.ac.at/forth/objects/opinion.html)
- Sleepless-Night Wiki (http://sleepless-night.com/cgi-bin/twiki/view/Main/WebHome)

Odkazy na INET

- Úvod pro zčátečníky (http://www.albany.net/~hello/simple.htm)
- Krátké příklady (http://www.jwdt.com/~paysan/screenful.html) (Mini-OOF (http://www.jwdt.com/~paysan/mini-oof.html) a A BNF Parser in Forth (http://www.zetetics.com/bj/papers/bnfparse.htm))
- Úvod do Forthu (http://astro.pas.rochester.edu/Forth/forth.html) od J.Kevina McFaddena

Implementace Forthu

- ppforth (http://members.nbci.com/_XMCM/pai123/ppforth.html)

4.1. Tipy

- Start simple. Get it running. Learn what you're trying to do. Add complexity gradually, as needed to fit the requirements and constraints. Don't be afraid to restart from scratch.
- Plan for change (by designing components that can be changed).
- First, and most importantly, the conceptual model should describe the system's interfaces.
- Factor the fruit. (Don't confuse apples with oranges).
- You don't underestand a problem until you can simplify it.
- Decide on error and exception-handling early as part of defining the intarface.
- Generality usually involves complexity. Don't generalize your solution any more than will be required; instead, keep it changeable.
- To simplify, take advantage of what's available.
- The mean time for making "two-hour" addition to an application is aproximatelly 12 hours.
-
-
-

4.2. Extreme Programming

Always implement things when you *actually* need then, never when you *foresee* that you need them. From (http://www.xprogramming.com/Practices/PracNotNeed.html)

4.3. Nejzákladnější slova

+
-
*
/
MOD
MIN
MAX
=
AND
OR
XOR
NEGATE
ABS
NOT
*/
DUP
DROP
SWAP
OVER
DECIMAL
HEX
OCTAL
.
n
.R
CR
EMIT
KEY
:
;
CREATE
,
ALLOT
IF
ELSE
THEN
FOR
NEXT
I

4.4. Čísla a slova

* *Povídaní o číslech a slovech, základních stavebních kamenech programu v jazyce Forth*

4.5. Zásobník a základní operace s ním

* *Co to je zásobník, jak funguje a jaké základní slova pro práci se zásobníkem Forth poskytuje. DROP DUP OVER ROT SWAP*

4.6. Definice nových slov

* *Jak se definují nová slova za pomoci slov již známých.*

Příklad 4-1. Definice nového slova

```
: nové_slovo známá slova ;
```

4.7. TIB (Terminal Input Buffer)

Část paměti sloužící jako vstupní buffer.

Kapitola 5. Základy jazyka Forth

* *rcsinfo="$Header: /home/radek/cvs/forth-book/ch-zaklady.xml,v 1.11 2005/10/20 05:33:42 radek Exp $"*
* *Tato kapitola vysvětlí základní pojmy a uvede nás do problematiky programování v jazyce Forth.*

5.1. Syntaxe jazyka

Syntaxe jazyka je velmi jednoduchá. S trochou nadsázky lze říci, že Forth žádnou syntaxi nemá.

Program v jazyce Forth je posloupnost slov a čísel oddělená bílými znaky (mezerou tabelátorem a znakem konce řádku). Znak konce řádku má ještě jeden význam a to ten že předává řízení forthu. Slovo je pak řetězec tisknutelných znaků, jako jsou například písmena, číslice, interpunkční a speciální znaky. Na velikosti písmen nezáleží. Příklady slov

```
@   !   ."   dump   <R   +   display-hook   cell+
```

Tedy znaky, které v jiných jazycích oddělují slova jsou ve Forthu legální součástí slov.

Interpret/kompilátor forthu, se kterým komunikujeme pak tuto posloupnost zpracuje tak, že slova rovnou vykoná a čísla uloží na vrchol zásobníku. Jak tak činí, k tomu se dostaneme později, teď si ukážeme pár příkladů.

5.2. RPN

Reverzní „polská“ notace, je zápis kdy symbol pro operaci se nachází až za svými operandy. Tedy sčítání `2 + 3` se zapíše jako `2 3 +`

5.3. Sémantika

Slova se zadávají v pořadí, v tomto pořadí se i vykonávají. Nejjednodušší slova jsou definovány v jádře interpretu ve strojovém kódu.

Slovo „+“ se chová takto

```
( POP + POP ) ⟶ PUSH
```

Například posloupnost `1 2 +` vyvolá na zásubníku tyto změny

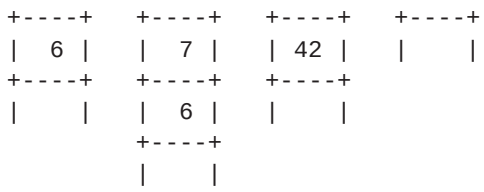
```
1        ( 1 )
2        ( 1 2 )
+        ( 3 )
```

Jak jsme si již řekli dříve, Forth čísla ze vstupu ukládá na zásobník a slova vykonává. Můžeme jej použít jako jednoduchý kalkulátor. Npříklad spočteme kolik je 6*7

```
6  7  *  .
```

Co se děje na zásobníku forthu? Ukážeme si to na obrázku:

```
      6        7        *        .
```



Popsáno slovy, první dvě čísla se uloží na zásobník v pořadí v jakém jsou zadána, tedy posledně vložené bude na vrcholu. „*“ je slovo, které se hned provede a vynásobí dvě čísla odebraná z vrcholu zásobníku a výsledek zase uloží na zásobník. Slovo „.“ pak odebere z vrcholu zásobníku číslo a vytiskne jej.

5.4. Počítání

Základní početní operace které jsou nám k dispozici jsou

+
: Sčítání -- Sečte dvě čísla na vrcholu zásobníku a výsledek uloží zpět do zásobníku.

-
: Odečítání

*
: Násobení

/
: Dělení

MOD
: Zbytek po dělení.

5.5. Základní operace nad zásobníkem

Základní operace nad obsahem zásobníku jsou tyto

Tabulka 5-1. Základní manipulace se zásobníkem

dup	(x1 ⟶ x1 x1)	zdvojí prvek na vrcholu zásobníku
drop	(x ⟶)	odstraní prvek z vrcholu zásobníku
swap	(x1 x2 ⟶ x2 x1)	prohodí prvek na vrcholu zásobníku s prvkem pod ním
over	(x1 x2 ⟶ x1 x2 x1)	zkopíruje prvek pod vrcholem na vrchol zásobníku

rot	(x1 x2 x3 ⟶ x2 x3 x1)	„rotace“ tří prvků na vrcholu zásobníku
nip	(x1 x2 ⟶ x2)	odstraní prvek pod vrcholem zásobníku
tuck	(x1 x2 ⟶ x2 x1 x2)	ekvivalntní `SWAP OVER`

DUP

Zdvojení obsahu na zásobníku. Například po vykonání

```
1 DUP
```

bude obsah zásobníku vypadata takto, TOS je vpravo

```
1 1
```

DROP

Odebere číslo z vrcholu zásobníku a zahodí jej.

```
2 6 DROP
```

```
2
```

SWAP

Prohodí číslo na vrcholu zásobníku s číslem pod ním. Tedy po

```
9 4 SWAP
```

vypadá zásobník takto

```
4 9
```

OVER

Na vrchol zásobníku přidá číslo z pod vrcholu, tedy

```
5 2 OVER
```

zanechá na zásobníku

```
5 2 5
```

ROT

FIXME:

5.5.1. DUP

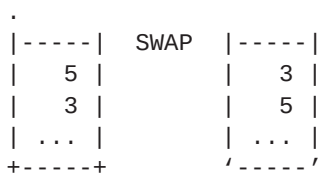
Zdvojí číslo na vrcholu zásobníku.

5.5.2. DROP

Odstraní číslo z vrcholu zásobníku.

5.5.3. SWAP

Vymění (prohodí) číslo na vrcholu zásobníku s číslem pod ním



5.5.4. OVER

.

5.6. Definice nových slov

Nyní již toho známe dostatek, abychom se mohli naučit definovat nová slova. Postup při definování je velmi jednoduchý

```
: název_slova definice_slova ;
```

Slovo „:“ zahajuje definici slova a je následováno jménem tohoto nového slova. Poté následuje vlastní definice která je ukončená znakem/slovem „;“. Předvedem si to na několika příkladech. Slovo **double** které zdvojnásobí číslo na vrcholu zásobníku si můžeme nadefinovat takto

```
: double 2 * ;
```

nebo s využitím základního matematického faktu že 2*x = x+x takto

```
: double dup + ;
```

5.7. Stack-Comment

Zásobníkové komentáře píšeme ke každé definici slova. Popisujeme v nich jaký efekt má vykonání slova na obsah zásobníku. Zapisujeme

```
( stav před vykonáním -- stav po vykonání )
```

Například k dříve definovanému slovu **double** si pozamenáme zásobníkový komentář

```
( n1 -- n2 )
```

který nám říká že slovou **double** očekává na zásobníku jedno číslo a po ukonční je nahrazeno jiným číslem.

n
: číslo v jednoduché přesnosti (zabírá jednu buňku) a se znaménkem

u
: číslo v jednoduché přesnsoti bez znaménka

d
: double -- číslo ve dvojté přesnosti (zabírá dvě buňky) se znaménkem

ud
: unsigned double -- číslo ve dvojité přesnosti bez znaménka

c
: znak (7 nebo 8 bitů), zabírá ale celou buňku.

b
: byte -- bajt, rovněž zabírá celou buňku

a nebo adr
: adresa

5.8. Základní aritmetika

.

5.8.1. +

.

5.8.2. -

.

5.8.3. *

.

5.8.4. /

.

5.9. Konstanty a proměnné

5.9.1. Konstanty

Definice konstanty

```
number constant name
```

příklady

```
5 constant five
-257 constant byeThrow
```

5.9.2. Proměnné

```
variable name
```

Pro užití proměnných máme dvě slova **!** a **@**. První z nich, slovo **!** čteme jako „store“ slouží k ukládání obsahu zásobníku na danou adresu. Použijeme jej takto

```
variable X
3 X !
```

Výsledkem je uložení čísla 3 do paměťové buňky proměnné X. Hodnotu uloženou v proměnné zase získáme příkazem **@** (fetch).

```
X @
.
```

5.9.3. Zásobník návratových adres

5.9.3.1. >R

TOS ⟶ rstack

5.9.3.2. R>

rstack ⟶ TOS

5.9.3.3. R@

rstack copy ⟶ TOS

5.9.3.4. RDROP

Zahození tos hodnoty na zásobníku návratových adres rstack

Kapitola 6. Řízení toku

* *rcsinfo="$Header: /home/radek/cvs/forth-book/ch-control_flow.xml,v 1.5 2005/10/20 05:33:42 radek Exp $"*

epigram

Text kapitoly

Příklady:

```
10 < if do-a else do-b then  \ je li hodnota menší než 10, tak DO-A jinak DO-B
10 0 do i do-think loop         \ pro čísla od 0 do 9 volej DO-THING
begin do-thin again             \ nekonečná smyčka, pořád se volá DO-THING
begin do-test while do-thing repeat \ volej DO-THING dokud je DO-TEST pravdivý
exit                            \ návrat z definice slova
```

II. Implementace

* *part*
* *rcsinfo="$Header: /home/radek/cvs/forth-book/forth.xml,v 1.31 2005/10/20 19:19:37 radek Exp $"*
* *print="**psselect -p47-164 forth.ps|foldprn -s24**"*

V této části se chci zabývat implementací jazyka Forth. Tedy způsoby jakými jej lze implementovat a rozeberu i konkrétní implemantace.

Kapitola 7. Implementace

* *chapter id="Implementace" xreflabel="Implementace"*
* *rcsinfo="$Header: /home/radek/cvs/forth-book/ch-implementace.xml,v 1.15 2005/11/09 23:11:35 radek Exp $"*

Odkazy a zdroje:

- Build Your Own Forth (http://www.figuk.plus.com/byof.htm)
- Moving Forth (http://www.zetetics.com/bj/papers/moving1.htm)

Forth je více než programovací jazyk, je to princip. Z toho vyplývá že jej můžeme implementovat více způsoby. Některé implementace jsou natolik odlišné, že se ani nenazývají Forth.

FIXME: Základem mě známých implementací je slovník jednotlivých slov. Tento je možno realizovat mnoha způsoby.

FIXME: Program je slovo definované pomocí jiných slov, již známých. Tato jsou pak definována pomocí jiných slov, atd. až skončíme u slov jenž jsou v základní výbavě daného „Forthu“.

7.1. Virtuální procesor

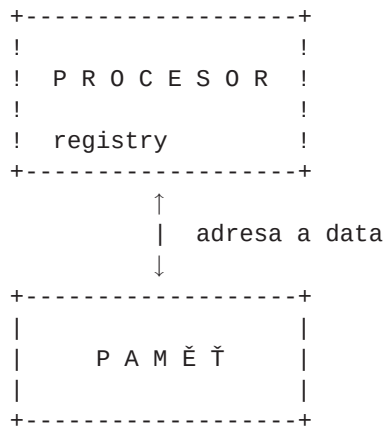
* *section id="virtual-processor" xreflabel="Virtuální procesor"*
* *rcsinfo="$Header: /home/radek/cvs/forth-book/ch-implementace.xml,v 1.15 2005/11/09 23:11:35 radek Exp $"*

Vysvětlovat si jak funguje Forth „zevnitř“ tak, abychom pochopili a poté implementovali Forth pro reálný procesor, je bez toho reálného procesoru velmi těžké. Proto k popisu fungování Forthu použiji virtuální procesor, jak se v literatuře používá.

* ***To be done:****Doplnit odkazy na virtuální procesor.*

Náš virtuální procesor **FIXME:**(von Neumanovy architektury|udělat odkaz např do Wikipedie) obsahuje sadu vnitřních registrů, a má spojení s pamětí (RAM/ROM) obsahující oba zásobníky a slovník.

Obrázek 7-1. Virtuální počítač



Pro jednoduchost o našem virtuálním procesoru nemáme příliš mnoho předpokladů. Předpokládáme pouze:

- paměť je adresovaná po slovech (buňkách)
- slovo (buňka) má dostatečnou velikost aby mohla obsahovat celou adresu

Vysvětlovat jak Forth funguje uvnitř, jak je implementován, je bez ukázky (praktického příkladu) na konkrétním procesoru téměř nemožné. Proto byl vytvořen model procesoru, virtuální procesor, jenž je k tomuto popisu vhodný a je základem implementace na skutečné procesory.

FIXME: Vysvětlovat jak forth funguje, bez ukázky na konkrétním procesoru je téměř nemožné. Proto vznikl v průběhu času virtuální forth procesor. Je to myšlený stroj který je dostatečně blízko běžnému hardwéru a přitom dostatečně obecný. Na něm ukážeme funkčnost forthu a tento virtuální procesor také bývá základem konkrétní implementace.

FIXME: Na tomto stroji si ukážeme ?funkčnost? forthu a později jej použijeme při implemetacích na konkrétní, skutečné procesory.

FIXME: Model FORTHu obsahuje řadu registrů, jejichž výčet je uveden dále. Tyto tvoří virtuální procesor. V praxi, při realizaci jádra FORTHu na konkrétním procesoru pak přidělíme funkci jednotlivých virtuálních registrů skutečným registrům použitého procesoru a popíšeme implementaci základních operací instrukcemi tohoto použitého procesoru.

FIXME: Abychom se domluvili, definujeme si předem několik pojmů. Zásobníky jenž procesor používá na/pro ukládání hodnot velikosti buňky (`CELL`). Tato velikost je volena s ohledem na zpracovávané údaje a velikost adresního prostoru.

FIXME: Tedy ... protože velikost buňky != byte, je třeba ji znát. Konstanta `CELL` tedy bude dále označovat počet bytů/slabik ze kterých se buňka skládá.

Jednotlivé virtuální registry jsou tak veliké, aby se do nich vešla jedna buňka. To u 16-ti bitového FORTHu znamená 16-ti bitové registry, a u 32-bitového 32-bitové registry. To je třeba mít na zřeteli při přidělování funkcí těchto registrů skutečným registrům.

REMOVE: Abychom si mohli názorně a jednoduše vysvětlit princip Forthu, potřebujeme k tomu procesor na kterém jej předvedeme. Pro tyto účely si zavedeme „virtuální procesor“ jehož struktura, orientovaná na forth, je natolik jednoduchá, že nás nebude zatěžovat specifickými detaily.

7.1.1. Registry virtuálního procesoru

* *section id="virtual-processor.register-set"*
* *rcsinfo="$Header: /home/radek/cvs/forth-book/ch-implementace.xml,v 1.15 2005/11/09 23:11:35 radek Exp $"*

Nyní k registrům jenž náš virtuální procesor obsahuje. Uvedu jejich zavedená jména a zároveň popis jejich významu.

W — pracovní registr (*Working Register*)

Pracovní a pomocný registr. Je používán k řadě věcí včetně dočasného uchovávání adresy. Je třeba aby tento registr bylo možno použít jako adresový registr při práci s pamětí. Tento registr se používá v každém slově. Často používané operace mimo aritmetických a logických je přístup do paměti (LOAD, STORE).

IP — ukazatel instrukcí (*Instruction Pointer / Interpreter Pointer*)

Ukazatel na instrukci. Podle implementace a průběhu programu ukazuje na aktuálně vykonávanou instrukci / slovo, nebo o jednu instrukci dopředu, tedy na následující instrukci. Je používán každým slovem. Používá jej implementace slov **NEXT**, **ENTER** a **EXIT**. Musí to být opět adresový registr. Nejčastější operace nad tímto registrem ja inkrementace (INC, DEC, LOAD).

PSP — *Parameter Stack Pointer / Data Stack*

Ukazatel zásobníku parametrů, někdy označován také jen SP (*Stack Pointer*). Je lepší jej označovat PSP protože označení SP často používá některý z registrů skutečného procesoru. PSP musí být adresový registr s charakterm ukazatele zásobníku. Potřebujeme jej inkrementovat, dekrementovat a použít jako adresu do paměti při čtení a zápisu (PUSH, POP). Podle způsobu realizace ukazuje na první volnou buňku na vrcholu zásobníku, na TOS, NOS či NNOS, podle toho jsou-li některé z těchto hodnot drženy

pro rychlost zpracování v registrech procesoru. Samotný zásobník, či jeho zbytek je pak v paměti (u speciálních Forth procesorů se nachází přímo v jádře). Nároky na zásobník parametrů nejsou veliké, postačí několik desítek slov.

RSP — *Return Stack Pointer*

Ukazatel zásobníku návratových adres někdy nazývaný RP. RSP musí mít charakter ukazatele zásobníku nebo adresového registru. Opět operace které budem používat jsou obdobné jako u PSP, tedy inkrementace, dekrementace a přístup do paměti (PUSH, POP).

X — *Working Register*

Pracovní registr. Pro některé operace budeme potřebovat další pomocný pracovní registr. Použití záleží na implementaci. Je důležitý pro procesory jenž nemohou použít jako jeden z operandů buňku v paměti.

UP — *User Pointer*

Ukazatel na uživatelský proces. Je použit u víceúlohového forthu. Ukazuje na důležité struktury jenž jsou pro každý proces samostatné a dovolují tak „paralelní“ běh několika úloh/aplikací, přesněji konkurentní přepínání mezi několika uživatelskými prostředími.

TOS — *Top Of Stack*

TOS není skutečným registrem ale v registru může být. Jedná se o vrchol zásobníku parametrů. Tedy je to označení pro buňku/slovo na které „ukazuje“ PSP. Slovo ukazuje je v úvozovkách, protože realizace zásobníku parametrů může implementovat ukazatel PSP jako ukazatel na první volnou buňku a nikoliv buňku na vrcholu zásobníku. Operace ve forthu pracují se zásobníkem a jeho vrcholem velmi často. Z důvodů výkonu se TOS implementuje v registru procesoru a zásobník obsahuje až další buňky pod vrcholem. V některých případech je i další hodnota NOS, případně NNOS implementována v registrech procesoru.

NOS — *Next Of Stack*

NOS stejně jako TOS není skutečným registrem ale v registru může být. Jedná se o buňku pod vrcholem zásobníku parametrů, tedy pod TOS.

NNOS — *Next Next Of Stack*

NNOS stejně jako TOS není skutečným registrem ale v registru může být. Jedná se o druhou buňku pod vrcholem zásobníku parametrů, tedy pod NOS.

Je-li to možné je potřeba s ohledem na výkon udržovat W, IP, PSP a RSP přímo v registrech procesoru.

Poznámka: ANS Forth vyžaduje aby zásobník parametrů adresovaný PSP měl alespoň 32 buňek (slov) a zásobník navratových adres adresovaný RSP alespoň 24 buňek (slov).

Následující tabulka je beze změny převzata z MOVING FORTH Part1: Design Decisions in the Forth Kernel (http://www.zetetics.com/bj/papers/moving1.htm).

	W	IP	PSP	RSP	UP	TOS	
8086[1]	BX	SI	SP	BP	memory	memory	[LAX84]
8086[2]	AX	SI	SP	BP	none	BX	[SER90]
68000	A5	A4	A3	A7=SP	A6	memory	[CUR86]
PDP-11	R2	R4	R5	R6=SP	R3	memory	[JAM80]
6809	X	Y	U	S	memory	memory	[TAL80]
6502	Zpage	Zpage	X	SP	Zpage	memory	[KUN81]

```
Z80         DE    BC    SP    IX    none   memory  [LOE81]
Z8          RR6   RR12  RR14  SP    RR10   RR8     [MPE92]
8051        R0,1  R2,3  R4,5  R6,7  fixed  memory  [PAY90]
```

7.1.2. Instrukce

FIXME:

Abychom rozuměli zápisu kódu pro náš virtuální procesor, musíme si definovat jeho jazyk, instrukce. Jazky který používám je podobný jazyku symbolických adres, assembleru, a jazyku C. Zápis je řádkově orientovaný. Každý řádek může obsahovat návěští, label, oddělený od zbytku řádku dvojtečkou. Za tímto nepovinným návěštím následuje pole instrukcí. Povoluji zápis více než jedné instrukce, v tomto případě jsou od sebe vzájemně odděleny středníkem `;` ;. Za polem instrukcí může být kometář ve stylu komentářů v shellu či perlu. zápis tedy vypadá takto:

```
label: instrukce; instrukce  # komentář
```

Instrukce přiřazení, tedy převodu hodnoty či hodnoty výrazu do registru nebo paměti je značena znakem ⟶ nebo ⟵, směr šipky označuje směr přesunu/zápisu hodnoty. Například přiřazení hodnoty 2 do registru `W` zapíši

```
2 ⟶ W
```

nebo

```
W ⟵ 2
```

Aritmetické a logické operace značím jak jste zvyklí z jiných jazyků:

```
A + 4 ⟶ B;
```

Čísla označují číselné hodnoty, konstanty. Názvy proměnných označují hodnoty v těchto proměnných uložené.

Odkaz na paměť se zapisuje do hranatých závorek. Zápis `[W]` tedy označuje hodnotu jenž se nachází v paměti na adrese jenž je uložena v registru `W`. Zápis `[[W]]` je pak označení hodnoty jenž se nachází na adrese uložené v paměti na adrese jenž je uložena v registru `W`. Zápis těchto dvou případů v jazyce C by vypadal takto `*W` pro první a `**W` pro druhý případ.

```
[A] [[W]]
(A) ((W))
```

7.2. Implementace slovníku

* *section*

* *rcsinfo="$Header: /home/radek/cvs/forth-book/sec-dictionary_implementation.xml,v 1.5 2005/10/21 13:36:02 radek Exp $"*

Odkazy a zdroje:

- Threaded code (http://www.complang.tuwien.ac.at/forth/threaded-code.html)

Základním kritériem podle kterého posuzujeme danou implementaci forthu je způsob implementace (organizace) slovníku slov. Rozeznáváme čtyři základní modely:

- nepřímo zřetezený kód
- přímo zřetězený kód

- podprogramy zřetězený kód
- tokeny zřetězený kód

Tyto základní modely si dále popíšeme. Mimo ně existuje ještě řada, ne tolik významných modelů, či jejich modifikací. Zde uvedu aspoň jejich výčet, tak jak jsem je našel na Internetu, bez dalšího popisu.

- STCI — Subroutine Threaded Code with Inlining
- Native Code
- BTC — Bit Threaded Code

Historicky nejstarším a původním modelem je nepřímo zřetezený kód. Jeho pochopení je důležité pro pochopení ostatních modelů ktere z něj vycházejí a různým způsobem rozšiřují.

Slovník sestává ze záznamů / definic jednotlivých slov. Tyto jsou za sebou lineárně uloženy v paměti. Jednotlivá slova mají dvě základní části: tělo a hlavičku.

```
+----------+----------+
| Hlavička |   Tělo   |
+----------+----------+
```

Hlavička obsahuje jméno slova, jeden bajt s přiznaky a velikostí jména a jeden ukazatel na předchozí slovo ve slovníku. Tento ukazatel slouží k vyhledání počátku předcházejícího slova jenž by jinak nebylo možno nalézti postupujeme-li od posledně definovaného slova ke slovům dříve definovaným až slovům základním.

```
+-----+------+-------+
| flg | link | jméno |
+-----+------+-------+
```

Bajt příznaků `flg` obsahuje ve spodních bitech délku jména a horní bity jsou příznaky se speciálním významem. Bit `b7` bývá 1. Následující hodnota má velikost ukazatele paměti, a ukazuje na předchozí slovo ve slovníku. Přesněji na jeho tělo, začátek těla. Za ní následuje `jméno` slova v 7-bit ASCII kódování. Poslední písmeno má nastaven sedmý bit na 1.

Tělo pak obsahuje některé z následujících polí:

CFA *Code Field Address*

: Adresa strojového kódu. Toto pole obsahuje spojovací adresu jenž ukazuje na strojový kód slova. V případě nízkoúrovňových slov které mají stroový kód v poli `PF` ukazuje na toto pole, v případě vysokoúrovňových slov ukazuje na vstupní bod interpretu `ENTRY`.

PF *Parameter Field*

: Pole parametrů, obsahuje strojový kód slova v případě nízkoúrovňových slov, nebo adresy slov v případě vysokoúrovňových slov.

7.2.1. Nepřímo zřetězený kód (*Indirect Threaded Code*)

* *section id="indirect-threaded-code" xreflabel="Nepřímo zřetězený kód"*

* *rcsinfo="$Header: /home/radek/cvs/forth-book/sec-indirect_threaded_code.xml,v 1.7 2005/11/09 23:11:35 radek Exp $"*

Prvotní implemetace forthu, používající nepřímo zřetězený kód je historicky nejstarší a původní implementací. Je také jednou z často používaných implemetací. Základní struktura slova ve slovníku vypadá následovně:

```
+----------+-----+-------+
```

```
| hlavička | CFA | PF    |
+----------+-----+-------+

STATUS:     .BYTE
NAME:       .STRING
LINK:       .WORD
CFA:        .WORD
PF:         .WORDS
```

Tomuto vzoru odpovídají všechna slova. Základní (nízkoúrovňová) slova ve strojovém kódu mají v poli `CFA` hodnotu ukazující na pole parametrů `PF`, které obsahuje přímo strojový k slova.

```
              .---.
             /    v
+----------+-----+-------------+
| hlavička | CFA | PF          |
+----------+-----+-------------+
```

Vysokoúrovňové slovo, tedy slovo překládané kompilátorem `:`, obsahuje v `CFA` ukazatel na strojový kód interpretu vysokoúrovňových slov jenž se obvykle nazývá `DOCOLON`, `DOCOL` nebo `ENTER`. V poli `PF` je pak řada ukazatelů na těla slov. Tedy na `CFA` pole slov pomocí nichž je toto slovo definováno.

```
+----------+-----+---------------------------------+
| Hlavička | CFA | adr1  adr2  adr3  ...  adr_exit |
+----------+-----+---------------------------------+
             |
             v
            ENTER
```

Tento mechanismus nám umožňuje snadno rozšířit implementaci forthu o nové druhy interpretů. Například můžeme mít interpreter 4 bitových čí 8-mi bitových instrukcí. Tím můžeme uspořit paměti.

Nyní, když víme, jak vypadá struktura slov ve slovníku, můžeme si ukázet definice základních slov/operací. Jedná se o slova/procedury/funkce `next`, `enter` a `exit`. Ukážeme si nyní jak tyto operace implementovat na našem virtuálním procesoru.

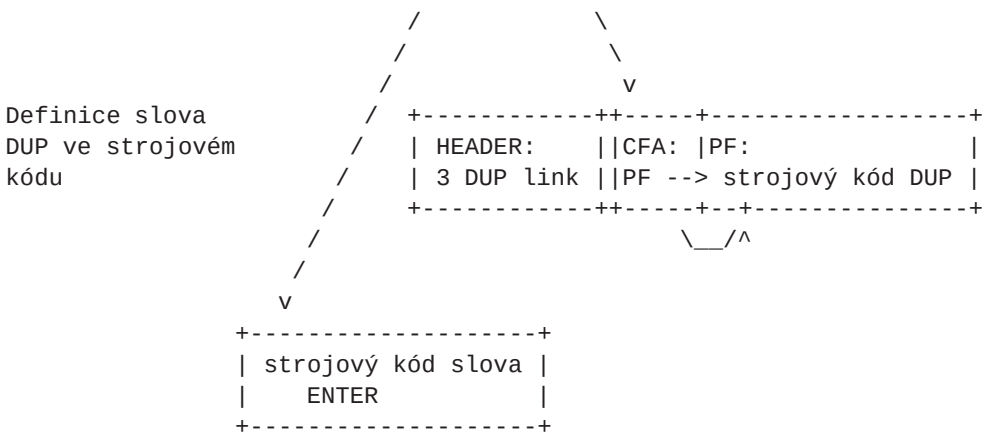
Představme si, že máme slovo `SQUARE` které je definováno takto:

```
: SQUARE DUP * ;
```

Na následujícím obrázku je vidět jak jsou jednotlivá slova ve slovníku a kód provázány ukazateli.

Obrázek 7-2. Provázaní slov v ITC modelu

```
                                  IP
                                   |
                                   v
PF slova FOO      ----+--------+--------+--------+--------+----
jenž používá      ... | GETNUM | SQUARE | GROK   | ...
slovo SQUARE      ----+--------+--------+--------+--------+----
                               /
                              /
                             v
Definice   +---------------++------+------+------+------+
slova      |HEADER:        ||CFA:  |PF:                 |
SQUARE     | 6 SQUARE link || ENTER| DUP  |  *   | EXIT |
           +---------------++------+------+------+------+
```



FIXME: Rutina `next` provádí přechod od jedné instrukce (adresy) v seznamu `PF` k následující instrukci (adrese). `IP` ukazuje na následující instrukci, tedy tu jenž se má vykonat.

V průběhu vykonávání slova **FOO**, na obrázku úplně nahoře, ukazuje registr IP na instrukci **SQUARE**. Procedura `NEXT` začne získáním adresy instrukce **SQUARE** a uložením této do registru W. Registr W tedy obsahuje adresu `CFA` pole slova **SQUARE**. Obsah registru IP je poté zvětšen o velikost buňky aby ukazoval na následující instrukci za instrukcí **SQUARE**. Následuje vykonání slova **SQUARE** skokem na adresu uloženou v jeho `CFA`. Formálně si to zapíšeme takto:

```
        # Vykonání dalšího slova v definici slova FOO.
        # Tedy vykonej slova SQUARE ne které ukazuje IP
        # IP ukazuje na slovo které se má vykonat (SQUARE)
        # W  nedefinováno
next:   [ip] ⟶ w      # W obsahuje adresu CFA slova SQUARE
        ip + cell ⟶ ip # Posun IP na GROK

        # continue [w]     Nepřímý skok na adresu v CFA SQUARE
        [w] ⟶ w       # dereference w
        continue w      # Pokračuj ve vykonávání programu na adrese CFA SQUARE

        # Před převedením řízení na CFA slova SQUARE mají
        # registry následující hodnoty:
        # IP ukazuje na následující slovo GROK, jenž se bude
        #    vykonávat po ukončení slova SQUARE
        # W  ukazuje na CFA slova SQUARE
```

Protože je slovo **SQUARE** definováno kompilátorem **:**, je v jeho `CFA` adresa interpretu `ENTER`. Poslední instrukce rutiny `NEXT` tedy končí skokem na strojový kód interpretu `ENTER`.

Prvním krokem interpretu `ENTER` je tedy uschování hodnoty ukazatele instrukcí `IP` do zásobníku návratových adres `RS`. Je to proto, abychom mohli pokračovat v interpretaci slova **FOO** vykonáním další instrukce v pořadí, **GROK**. Poté uložím do IP obsah W které ukazuje na `CFA` slova **SQUARE**. Rutina končí voláním procedury `NEXT`. V tomto cyklu vykonání `NEXT` ukazuje IP na `CFA` **SQUARE**. Dojde tedy k získání adresy `ENTER` uložené v tomto `CFA` a ke skoku na tuto adresu. **FIXME:**

```
        # ENTER (DOCOL or DOCOLON)
        # IP ukazuje na slovo GROK, jenž se má vykonat
        #    po ukončení interpretace slova SQUARE
        # W  ukazuje na CFA slova SQUARE
enter:  #push ip to rp    Uschování IP do zásobníku RS (návratových adres)
        ip ⟶ [rp]; rp+cell ⟶ rp
        w + cell ⟶ ip # IP ukazuje na PF slova SQUARE
```

```
        continue on next   # Vykonej slovo SQUARE
```

Poslední slove v definici **SQUARE** je `EXIT`. To provede návrat k bodu zpraování slova **FOO**.

```
        # EXIT called ;S in fig-Forth
exit:   #pop ip from rp  #  Obnov IP ze zásobníku, to nyní ukazuje na GROK
        [rp] ⟶ ip
        rp - cell ⟶ rp
        continuee next  #  Pokračuj vykonáním slova GROK
```

Ve zkratce jsou tedy procedury `NEXT`, `ENTER` a `EXIT` definovány takto:

```
next:  [ip] ⟶ w; ip + cell ⟶ ip; jmp (w)
enter: rp - cell ⟶ rp; ip + cell ⟶ [rp]; w ⟶ ip; next
exit:  [rp] ⟶ ip; rp + cell ⟶ rp; next
```

Pro implementaci kde buňka má velikost jednoho slova a základní adresovatelnou jednotkou je jedno slovo, můžeme zápis za pomocí pre decrement (--) a post increment (++) operátorů zjednodušit takto:

```
next:  [ip++] ⟶ w; jmp (w)  //
enter: ip + 1 ⟶ [--rp]; w ⟶ ip; next
exit:  [rp++] ⟶ ip; next
```

Pro implemetaci kde velikost buňky je dvojnásobkem základní paměťové jednotky, typickým příkladem je implementace 16-ti bitového Forthu na 8-mi bitovém procesoru, to bude vypadat následovně:

```
next:  [ip] ⟶ w; ip + 2 ⟶ ip; jmp (w)  //
enter: rp - 2 ⟶ rp; ip + 2 ⟶ [rp]; w ⟶ ip; next
exit:  [rp] ⟶ ip; rp + 2 ⟶ rp; next
```

* *Zkontrolováno podle MOVING FORTH, Part1: Design Decisions in the Forth Kernel by Brad Rodriguez (http://www.zetetics.com/bj/papers/moving1.htm), a důkladně promyšleno.*

> **Poznámka:** Všiměte si, že při vstupu do funkce `ENTER` ukazuje W na CFA slova jenž se má interpretovat. Proto před vastní interpretací (`NEXT`) je nutno do IP zapsat hodnotu o jednu buňku (CELL) větší, aby IP ukazoval na PF, tedy na první instrukci/adresu. Toto posunutí na následující buňku by se dalo přemístnit do funkce `NEXT` a to tak že by se instrukce `jmp (w)` nahradila instrukcí `jmp (w)+`. Výsledný kód by tedy vypadal:
>
> ```
> next:   (ip) ⟶ w
>         ip+
>         jmp (w)+
> ```
>
> Po rozepsání složitějších obratů za užití pomocného registru X:
>
> ```
> next:   (ip) ⟶ w
>         ip + cell ⟶ ip
>         (w) ⟶ x
>         w + cell ⟶ w
>         jmp x
> ```
>
> Důvod proč to tak učinit je jeden. Pokud budem mít alespoň dva interprety, nebudeme operaci W+cell provádět dvakrát, v každém vstupním bodu `ENTRY` a `ENTRY2`.

FIXME: Zbytek probrat a vyřadit.

Teď si je trochu blíže rozepíšeme.

Vnitřní interpret `NEXT` pro virtuální procesor

```
NEXT:   (IP)    ⟶ W     ; úschova ukazatele do pracovního registru
        next IP ⟶ IP    ; posunutí na další adresu v definici (ip++)
        (W)     ⟶ X     ; dereference adresy
        JUMP X              ; vykonání slova v definici
```

K vnitřnímu interpretu je třeba dodat kód interpretu vysokoúrovňových slov `ENTER`

```
ENTER:  ; PUSH IP TO RSP
        prev RSP ⟶ RSP
        IP ⟶ (RSP)
        next W ⟶ IP      ; uložení adresy PFA do IP
        JUMP NEXT
```

definice slova je ukončena adresou procedury `EXIT` ukončení definice

```
EXIT:   ; POP IP FROM RSP
        (RSP) ⟶ IP
        next RSP ⟶ RSP
        JUMP NEXT
```

Each code field contains a machine code fragment. So, in the case of a primitive (machine code) definition, the xt is the address of the machin code itself.

Struktura slova ve slovníku

```
+----------+-----+--------------------------------+
| Hlavička | CFA | PF: adr1 adr2 adr3 ... adr_exit |
+----------+-----+--------------------------------+

+----+-----+-----+-------------+
| NF | LF^ | CF^ | PF          |
+----+-----+-----+-------------+
```

```
; DOCOLON někdy nazývaná ENTER
DOCOLON:    PUSH IP             ; IP⟶-(RSP)
            W + 2⟶IP       ; PFA(W)⟶IP
            ; IP ukazuje na první adresu v PF
            JUMP NEXT           ; Skok do interpreteru adres

; DOSEMI někdy nazývaná EXIT poslední adresa v PF
DOSEMI:     POP IP              ; (RSP)+⟶IP
            JUMP NEXT

NEXT:       (IP)⟶W
            IP + 2⟶IP      ; posun na další adresu
            (W)⟶X          ;
            JUMP (X)
```

7.2.2. Přímo zřetězený kód (*Direct Threaded Code*)

* *section id="direct_threaded_code" xreflabel="Přímo zřetězený kód"*
* *rcsinfo="$Header: /home/radek/cvs/forth-book/sec-direct_threaded_code.xml,v 1.4 2005/10/21 13:36:02 radek Exp $"*

Odkazy a zdroje:

- . ()

ToDo

1. úkol

Přímo zřetězený kód je modifikací ITC. Zatímco v ITC obsahuje tělo dvě pole, `CFA` a `PF`, v DTC je pole jen jedno, `PF`. Jak tedy Forth rozpozná kde se nachází strojový kód, jednoduše ten je uložen přímo v poli `PF`. Nízkoúrovňová slova obsahují tedy strojový kód přímo v `PF`. Vysokoúrovňová slova obsahují jako první strojovou instrukci v `PF` instrukci skoku (`JUMP`) či instrukci volání podprogramu (`CALL`) na strojový kód interpretu `NEXT`.

Co získáme v DTC oproti ITC? V první řadě je to snížení nároků na kód interpretu o jedno nepřímé adresování. V DTC se v kódu interpretu `NEXT` nahradí instrukce `jmp (w)` instrukcí `jmp w`. Kód nízkoúrovňových slov ve strojovém kódu se zkrátí o velikost pole `CFA`, u vysokoúrovňových slov může dojít ke zvětšní o kód instrukce skoku na interpret `NEXT`.

Implementace:

```
next:   (ip) ⟶ w
        ip + 2 ⟶ ip
        jump w          ; skok na adresu v registru W
```

Struktura slova v slovníku:

```
+----------+----------+------------------------+
| Hlavička | JMP NEST | adr1 adr2 adr3 ... exit |
+----------+----------+------------------------+
            CF         PF
```

DOSEMI (EXIT) je stejný jako v ITC. NEXT je jednodušší

```
next:   (ip) ⟶ w
        ip+2 ⟶ ip     ; posun na další adresu
        jump w

NEST:       ; pro JMP-based systems
            -(RSP)⟵IP
            IP⟵W + sizeof(JMP)
            JUMP NEXT

NEST:       ; pro systémy používající CALL (JSR)
            -(RSP)⟵IP
            POP IP
            JUMP NEXT
```

7.2.3. Podprogramy zřetězený kód (*Subroutine Threaded Code*)

* *$Header: /home/radek/cvs/forth-book/sec-subroutine_threaded_code.xml,v 1.4 2005/10/20 19:19:38 radek Exp $*
* *section id="subroutine-threaded-code" xreflabel="Podprogramy zřetězený kód"*

Model zřetězení slov podprogramy, také nazývaný nativně přakládaný kód (*Native Code*), je založený na tom, že definice slova v slovníku je sama o sobě výkonným kódem. Tedy volání slov je přeloženo na volání podprogramů.

Základní odlišnost od předcházejících modelu spočívá v použití instrukcí skoků místo prostých adres v definici slova.

Výhodou tohoto modelu je větší rychlost vykonávání programu. Toto však není obecně samozřejmé a je nutno pro daný konkrétní procesor porovnat jednotlivé modely. Tento model nerozlišuje mezi slovy nízkoúrovňovými a vysokoúrovňovými.

Obecně model nízkoúrovňového slova v zásobníku vypadá takto:

```
+----------++--------------------+-----+
| hlavička || Strojový kód slova | RET |
|  slova   ||                    |     |
+----------++--------------------+-----+
```

Vysokoúrovňové slovo pak vypadá takto:

```
+----------++-------+-------+-------+-----+
| hlavička || CALL  | CALL  | CALL  | RET |
|  slova   || word1 | word2 | word3 |     |
+----------++-------+-------+-------+-----+
```

Jak je na ukázkách vidět z hlediska organizace není mezi těmito slovy principiálního rozdílu. Obě obsahují přímo vykonavatelný strojový kód.

Tento model se dále vyznačuje

- neexistencí vnitřního interpretu `NEXT`
- neexistencí virtuálního IP registru, jeho funkci plně zastává čítač programu (`PC`) procesoru
- neexistencí virtuálního W registru

7.2.4. Tokeny zřetězený kód (*TTC — Token Threaded Code*)

* *$Header: /home/radek/cvs/forth-book/sec-token_threaded_code.xml,v 1.4 2005/01/30 09:48:12 radek Exp $*

* *section id="token-threaded-code" xreflabel="Tokeny zřetězený kód"*

Tento model nepoužívá v definicích přímé adresy ale tak zvané tokeny. Tokeny jsou indexy do tabulky tokenů kde je ke každému tokenu přiřazena adresa. Motivací pro tento model je snížení paměťových nároků a tím větší hustota kódu.

Pro představu, použijeme-li tokeny 8 bitů veliké, tak při realizaci 16-ti bitového forthu snížíme průměrnou velikost definice slova na polovinu oproti použití přímých adres. Nevýhodou je pak omezení maximálního počtu definovaných slov na 256 a snížení rychlosti interpretace způsobené jednou dereferencí navíc.

Omezení dané velikostí tokenu (8 bitů) lze obejít. Pokud se například slovník forthu nachází v horní polovině adresního prostoru (od adresy 0x8000), zavedeme jen 128 tokenů. Ty budou zapisovány jako hodnoty od 0 do 127. Bude-li načtena v průběhu vakonávání slova hodnota tokenu vetší než 128, tedy od 128 do 255, znamená to že se nejedná o token ale horní část adresy slova ve slovníku. Tím budeme mít k dispozici nejen 128 tokenů, ale taky prostor pro libovolné množství dalších slov dle paměťových možností našeho počítače.

Následující tabulka popisuje velikost prostoru pro přímo adresovaný slovník a maximální počet použitelných tokenů. Hodnoty jsou pro 8-mi bitové tokeny a procesor s 16-ti bitovou adresou, tedy pro bežný 8-mi bitový procesor.

tokeny [n]	slovník [KiB]	poznámka
32	56KiB	
64	48Kib	potřebujeme-li veliký prostor pro slovník
128	32KiB	
192	16KiB	vystačíme-li si z malým slovníkem
224	8KiB	extrémě malý prostor pro slovník

K tabulce je třeba dodat, že velikostí slovníku se rozumí velikost přímo adresovatelného slovníku. Do této velikosti se nepočítá velikost slovníku adresovatelného tokeny.

7.2.5. Huffman threading

Odkazy a zdroje:

- Wikipedia: Threaded code (http://en.wikipedia.org/wiki/Threaded_code)
- Wikipedia: Huffman coding (http://en.wikipedia.org/wiki/Huffman_code)

FIXME:přeložit: Huffman threaded code consists of lists of Huffman codes. A Huffman code is a variable length bit string used to identify a unique item. A Huffman-threaded interpreter locates subroutines using an index table or tree of pointers that can be navigated by the Huffman code. Huffman threaded code is one of the most compact representations known for a computer program. Basically the index and codes are organized by measuring the frequency that each subroutine occurs in the code. Frequent calls are given the shortest codes. Operations with approximately equal frequencies are given codes with nearly equal bit-lengths. Most Huffman-threaded systems have been implemented as direct-threaded Forth systems, and used to pack large amounts of slow-running code into small, cheap microcontrollers. Most published uses have been in toys, calculators or watches.

7.2.6. Rozdělený slovník

Doposud jsme uvažovali, že záznam slova ve slovníku obsahuje všechny informace: příznaky, název slova, adresy a kód slova. Tím ovšem slovo zbírá v produkčním systému více místa než je nezbytné. Jedním ze způsobů jak místem ušetřit je vyčlenit ze slovníku všechny informace, jenž nejdou pro chod programu nezbytné, a umístnit je do vlastního slovníku.

7.3. Vnitřní interpret

* *$Header: /home/radek/cvs/forth-book/sec-inner_interpreter.xml,v 1.1 2003/12/28 18:21:56 radek Exp $*

Vnitřní interpret je jádrem implementace forthu. Jeho tradiční jméno je `NEXT`. Ve své prapůvodní podstatě řídí „interpretaci“ definice vykonávaného slova.

Kapitola 8. Forth na procesoru CDP1802

* *chapter id="implementace.cdp1802" xreflabel="Forth na CDP1802"*
* *$Header: /home/radek/cvs/forth-book/ch-cdp1802.xml,v 1.5 2005/02/01 11:50:55 radek Exp $*

Analýza několika implementací a úvahy na téma jak implementovat forth na architektuře CDP1802, případně CDP1805.

Vtipný epigram

FIXME: Text kapitoly

8.1. FIG-FORTH 1802

Tato implementace stejně jako řada FIG-FORTH používá model nepřímo zřetězeného kódu. Z tohoto modelu pak vychází použitý kód, jak bude vidět dále, zejména pak na smyčce vnitřního interpretu.

Analyzuji zdrojový kód jenž pochází původně od Garyho R.Branshawa a byl modifikován Gordenem Flemmingem a Jimem McDanielem. Zdrojový text je datován 1981-03-16. Verze programu je `1802 FIG-FORTH R0.4 3/16/81`.

8.1.1. Přiřazení registrů

Než se začneme hlouběji zabývat strukturou programu, musíme si ozřejmit jaký význam je přiřazen jednotlivým registrům procesoru. Toto přiřazení ukazuje následující tabulka.

Tabulka 8-1. Význam registrů v implementaci FIG-FORTH 1802

registr	význam
R2	RSP (*Return Stack Pointer*), roste směrem k nižším adresám
R3	PC for I/O and primitives
R7, R8	Temporary Accumulator
R9	PSP (*Parameter Stack Pointer*) roste směrem k vyšším adresám
RA	IP, FORTH „I“ register
RB	FORTH „W“ register
RC	PC for inner interpreter
RD	User Pointer
RF	Disc I/O

Protože procesor CDP1802 neupřednostňuje žádný registr a všechny jsou si rovny. Není žádný preferovaný čítač instrukcí, žádný zásobník, žádný indexový registr. Proto není k tabulce co dodat. Přiřazení je možno bez jakýchkoliv následků libovolně změnit.

Ostatní registry, neuvedené v tabulce, FIG-FORTH 1802 nepoužívá a jsou k dispozici programátorovi.

8.1.2. Vnitřní interpret NEXT

* *section id="ff1802.next" xreflabel="next", xref.ff1802.next*

Jako první se podíváme na smyčku vnitřního interpretu a rozebereme si ji. Jak jsem již zmínil, tato implemetace FORTHu používá technologii nepřímo zřetězeného kódu a proto vychází z kódu:

```
next:   (ip+)⟶w
        jump (w+)
```

Nyní tedy vlastní kód smyčky. První část, instrukci `(ip+)⟶w`.

```
000235 0091                     ; NEXT INNER INTERPRETER
000236 0091 D3                  SEP R3          ; LEAVE RC AT NEXT
                                ; (ip+)⟶w     # store instruction address
                                ;               # to w
000237 0092 4A          NEXT:   LDA RA          ; (IP+)
000238 0093 BB                  PHI RB          ; ⟶W.hi
000239 0094 4A                  LDA RA          ; (IP+)
000240 0095 AB                  PLO RB          ; ⟶W.lo
```

První instrukce není součástí konstrukce získáni adresy dalšího slova, je zde jako součást mechanizmu volání vnitřního interpretu `NEXT` a uzavírá smyčku tohoto interpretu. Význam dalších instrukcí je zřejmý z komentářů, přečteme dva bajty tvořící slovo na adrese ukazované IP (RA) a uložíme do registru W (RB). Následující, pokračování dokončuje smyčku vnitřního interpretu.

```
                                ; (w+)⟶pc
000241 0096 4B          WBR:    LDA RB          ; (W+)
000242 0097 B3                  PHI R3          ; ⟶ PC.hi
000243 0098 4B                  LDA RB          ; (W+)
000244 0099 A3                  PLO R3          ; ⟶ PC.lo
000245 009A 3091                BR NEXT - 1     ; Continue on PC
```

Čtyři instrukce zajistí operaci `(w+)⟶pc` tedy načtení adresy v poli *`CF`* a její uložení do čítače instrukcí `PC`. Po vykonání tohoto kódu registr `W` ukazuje na *`PF`* vykonávaného slova. Poslední instrukce realizuje současně uzavření smyčky vnitřního interpretu a převedení řízení (skok) na získanou adresu jenž je uložena `PC`.

8.1.3. Interpret slov NEST

* *section id="ff1802.nest" xreflabel="nest", xref.ff1802.nest, link.ff1802.NEST*

Přesněji interpret slov definovaných na vyšší úrovni slovem **:**.

Implementace `nest` podle modelu nepřímo zřetězeného kódu tento je:

```
enter:  ip⟶(-rsp)     ; push ip to rs
        w⟶ip          ; jump to w
        next          ; předání řízení do smyčky vnitřního interpretu
```

V této implementaci se interpret slov nenazývá `ENTER`, ale `NEST`. Na prvních řádcích je implementována operace `push IP to RS`, tedy uložení ukazatele instrukcí na vrchol zásobníku návratových adres. To proto, abychom se mohli vrátit ve vykonávání programu zpět a pokračovat další instrukcí. Zde objevujeme další rozdíl, zásobník návratových adres roste v této implementaci směrem k vyšším adresám a nikoliv nižším jako v referenčním modelu.

```
                                ; ip⟶(rsp+)   # push ip to rs
001296 05C2 9A          NEST:   GHI RA          ; IP.hi
```

```
001297 05C3 52                      STR R2          ; ⟶(RSP)
001298 05C4 22                      DEC R2          ; RSP+=1
001299 05C5 8A                      GLO RA          ; IP.lo
001300 05C6 52                      STR R2          ; ⟶(RSP)
001301 05C7 22                      DEC R2          ; RSP+=1
```

V druhé části implementace `NEST` se adresa z pracovního registru `W` uloží do ukazatele instrukcí `IP`, a řízení se předá do smyčky vnitřního interpretu `NEXT`. Tím se provede efektivně volání slova/podprogramu jehož adresa byla v registru `W`.

```
                                    ; w⟶ip        # jump to w
001302 05C8 9B                      GHI RB          ; W.hi
001303 05C9 BA                      PHI RA          ; ⟶IP.hi
001304 05CA 8B                      GLO RB          ; W.lo
001305 05CB AA                      PLO RA          ; ⟶IP.lo
001306 05CC DC                      SEP RC          ; next
```

Slovo které se vykonává interpretem NEST je ukončeno adresou SEMIS slova **;S**

8.1.4. Návrat z interpretu slov ;S

* *section id="ff1802.unest" xreflabel=";S", xref.ff1802.unest, link.ff1802.SEMIS*

Tato funkce ukončuje interpretaci slova v interpretu vyšších slov nest. Jedná se o implementaci funkce `exit` v modelu nepřímo zřetězeného kódu. Modelový kód je:

```
exit:  (rsp+)⟶ip       # pop ip from rs
        next
```

V implementaci FIG-FORTH 1802 je toto slovo označeno `SEMIS` (**;S**)

Nejdříve tedy hlavička slova

```
000823 0374 823BD3                  .DB H'82,H'3B,H'D3; ;S (UNEST)
000824 0377 035F                    .DW RP1-6       ; link to prev. word RP1
000825 0379 037B            SEMIS:  .DW *+2; CFA⟶PF containing machine code
```

Kód slova je pak jednoduchý.

```
                                    ; (rsp+)⟶ip   # pop ip from rs
000826 037B 12                      INC R2          ; RSP+=1
000827 037C 42                      LDA R2          ; (RSP+)
000828 037D AA                      PLO RA          ; ⟶IP.lo
000829 037E 02                      LDN R2          ; (RSP)
000830 037F BA                      PHI RA          ; ⟶IP.hi
000831 0380 DC                      SEP RC          ; next
```

8.1.5. Definice slova : — *colon*

* *section id="ff1802.colon" xreflabel=":", link.ff1802.colon, xref.ff1802.colon*

Toto slovo otevírá definici nového slova. Jedná se o standardní konstrukci definice slova.

: *název* ... definice... ;

Toto slovo je samo definováno jako slovo vyšší úrovně a můžeme si tedy popsat jeho definici ve FORTHu.

```
: :  ?EXEC !CSP CURRENT @ CONTEXT ! CREATE ] -2 DP +! COMPILE nest ;
```

Nejdříve tedy hlavička.

```
002345 0DC0 C1BA                    .DW H’C1BA          ; :   (IMMEDIATE)
002346 0DC2 0D5B                    .DW CRTE - 9
002347 0DC4 05C2            COLON:  .DW NEST

002348 0DC6 08F0                    .DW EXC
002349 0DC8 08AB                    .DW DCSP
002350 0DCA 070E                    .DW CRNT
002351 0DCC 04FA                    .DW AT
002352 0DCE 0700                    .DW CNTX
002353 0DD0 051E                    .DW EX
002354 0DD2 0D64                    .DW CRTE
002355 0DD4 0975                    .DW RBK
002356 0DD6 0086                    .DW LIT
002357 0DD8 FFFE                    .DW H’FFFE          ; -2
002358 0DDA 06AF                    .DW DP
002359 0DDC 04C6                    .DW PLUSS
002360 0DDE 0951                    .DW CMPL
002361 0DE0 05C2                    .DW NEST
002362 0DE2 0379                    .DW SEMIS
```

8.1.6. LIT

* *section id="ff1802.lit" xreflabel="LIT", xref.ff1802.lit, link.ff1802.LIT*

Toto slovo se nepoužívá přímo ale je součástí použití čísla v definici. Kdykoliv použijeme číslo, toto se uloží na zásobník parametrů. Ale použijeme-li číslo v definici slova, vloží se do této definice posloupnost

DW LIT
DW *číslo*

A teď k samotné definici slova `LIT`

```
000221 0080 834C49D4                .DB H’83,"LI",H’D4 ; LIT
000222 0084 0000                    .DW H’0000
000223 0086 0088            LIT:    .DW * + 2
                                    ; (ip+)⟶(+psp)
000224 0088 19                      INC R9          ; PSP+=1
000225 0089 19                      INC R9          ; PSP+=1
000226 008A 4A                      LDA RA          ; (IP+)
000227 008B 59                      STR R9          ; ⟶PSP
000228 008C 19                      INC R9          ; PSP+=1
000229 008D 4A                      LDA RA          ; (IP+)
000230 008E 59                      STR R9          ; ⟶PSP
000231 008F 29                      DEC R9          ; PSP-=1
000232 0090 DC                      SEP RC          ; next
```

8.1.7. Vykonání slova (EXECUTE)

* *section id="ff1802.execute" xreflabel="EXECUTE", xref.ff1802.execute, link.ff1802.EXE*

Protože interpret slov je již definován jako slovo FORTHu, má hlavičku jenž pedchází tělu. Tato hlavička definuje název slova `EXECUTE`, ukazatel na předchozí slovo ve slovníku, což je v našem případě slovo `LIT` a ukazatel na strojový kód jenž ukazuje do vlastního těla `EXECUTE` jenž v poli *PF* obsahuje strojový kód.

```
000246 009C                          ; EXECUTE
000247 009C 87455845435554C5         .DB H'87,"EXECUT",H'C5; EXECUTE
000248 00A4 0080                     .DW LIT-6    ; link to previous word "LIT"
000249 00A6 00A8             EXE:    .DW *+2; CFA⟶PF containg machine code
```

Nyní již samotný kód slova. První část vyzvedne ze zásobníku parametrů (`PSP`) adresu slova a uloží ji do pomocného registru `W`.

```
                                     ; LOAD W FROM STACK
                                     ; (psp+)⟶w
000250 00A8 49                       LDA R9
000251 00A9 BB                       PHI RB
000252 00AA 09                       LDN R9          ; LOAD W FROM STACK
000253 00AB AB                       PLO RB
```

Druhá část kódu opraví stav zásobníku. V důsledku použití jedné instrukce **LDA** a druhé **LDN** se `PSP` zvětší o jedničku. O tuto jedničku jej opravíme první instrukcí **DEC** a druhé dvě instrukce se posunou na předchozí buňku. Tímto efektivně odstraníme adresu ze zásobníku.

```
000254 00AC 29                       DEC R9
000255 00AD 29                       DEC R9
000256 00AE 29                       DEC R9
```

Následuje spuštění vnitřního interpretu. Tento se ovšem spouští od adresy `WBR`. Abychom toho dosáhli, musíme opravit ukazatel instrukcí v `RB` tak že jej z `NEXT` posuneme o čtyřy bajty na `WBR`.

```
                                     ; advance RC to point to WBR
000257 00AF 1C                       INC RC
000258 00B0 1C                       INC RC
000259 00B1 1C                       INC RC          ; POINT TO WBR
000260 00B2 1C                       INC RC
000261 00B3 DC                       SEP RC          ; next
```

8.1.8. BRANCH

* *section id="ff1802.branch" xreflabel="BRANCH", xref.ff1802.branch, link.ff1802.BRCH, link.ff1802.BRANCH*

```
000264 00B4 864252414E43C8           .DB H'86,"BRANC",H'C8 ; BRANCH
000265 00BB 009C                     .DW EXE-10
000266 00BD 00BF             BRCH:   .DW * + 2       ; CFA⟶PF
000267 00BF 4A               BRANCH: LDA RA          ;
000268 00C0 52                       STR R2
000269 00C1 4A                       LDA RA
000270 00C2 AA                       PLO RA
000271 00C3 02                       LDN R2
000272 00C4 BA                       PHI RA
000273 00C5 DC                       SEP RC          ; next
```

8.1.9. 0BRANCH

* *section id="ff1802.0branch" xreflabel="0BRANCH", xref.ff1802.0branch, link.ff1802.ZBRCH*

```
000276 00C6 87304252414E43C8          .DB H'87,"0BRANC",H'C8 ; 0BRANCH
000277 00CE 00B4                      .DW BRCH-9
000278 00D0 00D2            ZBRCH:    .DW * + 2
000279 00D2 49                        LDA R9
000280 00D3 3ADD                      BNZ NO
000281 00D5 09                        LDN R9
000282 00D6 3ADD                      BNZ NO
000283 00D8 29                        DEC R9
000284 00D9 29                        DEC R9
000285 00DA 29                        DEC R9
000286 00DB 30BF                      BR BRANCH
000287 00DD 1A              NO:       INC RA
000288 00DE 1A                        INC RA
000289 00DF 29                        DEC R9
000290 00E0 29                        DEC R9
000291 00E1 29                        DEC R9
000292 00E2 DC                        SEP RC          ; next
```

8.1.10. (LOOP)

* *section id="ff1802.paren-loop-paren" xreflabel="(LOOP)", xref.ff1802.paren-loop-paren, link.ff1802.LUPE, link.ff1802.COMP*

```
000294 00E3 86284C4F4F50A9            .DB H'86,"(LOOP",H'A9 ; (LOOP)
000295 00EA 00C6                      .DW ZBRCH-10
000296 00EC 00EE            LUPE:     .DW * + 2
000297 00EE 12                        INC R2
000298 00EF 92                        GHI R2
000299 00F0 B8                        PHI R8
000300 00F1 B7                        PHI R7
000301 00F2 82                        GLO R2
000302 00F3 A8                        PLO R8
000303 00F4 A7                        PLO R7
000304 00F5 22                        DEC R2
000305 00F6 08                        LDN R8
000306 00F7 FC01                      ADI H'01
000307 00F9 58                        STR R8
000308 00FA 18                        INC R8
000309 00FB 08                        LDN R8
000310 00FC 7C00                      ADCI H'00
000311 00FE C4                        NOP                 ; TO NEW PAGE
000312 00FF C4                        NOP
000313 0100 58              COMP:     STR R8
000314 0101 18                        INC R8
000315 0102 E7                        SEX R7
000316 0103 48                        LDA R8
000317 0104 F5                        SD
000318 0105 17                        INC R7
000319 0106 08                        LDN R8
000320 0107 75                        SDB
000321 0108 FA80                      ANI H'80
000322 010A 3213                      BZ CEND
```

```
000323 010C 4A                      LDA RA
000324 010D 52                      STR R2
000325 010E 0A                      LDN RA
000326 010F AA                      PLO RA
000327 0110 02                      LDN R2
000328 0111 BA                      PHI RA
000329 0112 DC                      SEP RC          ; next
000330 0113 1A              CEND:   INC RA
000331 0114 1A                      INC RA
000332 0115 12                      INC R2
000333 0116 12                      INC R2
000334 0117 12                      INC R2
000335 0118 12                      INC R2
000336 0119 DC                      SEP RC          ; next
```

8.1.11. (+LOOP)

* *section id="ff1802.paren-plus-loop-paren" xreflabel="(+LOOP)", xref.ff1802.paren-plus-loop-paren, link.ff1802.PLUPE*

```
000338 011A 87282B4C4F4F50A9        .DB H'87,"(+LOOP",H'A9 ; (+LOOP)
000339 0122 00E3                    .DW LUPE-9
000340 0124 0126            PLUPE:  .DW * + 2
000341 0126 12                      INC R2
000342 0127 92                      GHI R2
000343 0128 B8                      PHI R8
000344 0129 B7                      PHI R7
000345 012A 82                      GLO R2
000346 012B A8                      PLO R8
000347 012C A7                      PLO R7
000348 012D 22                      DEC R2
000349 012E E9                      SEX R9
000350 012F 19                      INC R9
000351 0130 08                      LDN R8
000352 0131 F4                      ADD
000353 0132 58                      STR R8
000354 0133 18                      INC R8
000355 0134 29                      DEC R9
000356 0135 08                      LDN R8
000357 0136 74                      ADC
000358 0137 58                      STR R8
000359 0138 09                      LDN R9
000360 0139 FE                      SHL
000361 013A 29                      DEC R9
000362 013B 29                      DEC R9
000363 013C 3340                    BDF LUPE1
000364 013E 3001                    BR COMP+1
000365 0140 18              LUPE1:  INC R8
000366 0141 E7                      SEX R7
000367 0142 48                      LDA R8
000368 0143 F7                      SM
000369 0144 17                      INC R7
000370 0145 08                      LDN R8
000371 0146 77                      SMB
000372 0147 3008                    BR COMP+8
```

8.1.12. (DO)

* *section id="ff1802.paren-do-paren" xreflabel="(DO)", xref.ff1802.paren-do-paren, link.ff1802.PDO*

```
000375 0149 8428444FA9              .DB H’84,"(DO",H’A9 ; (DO)
000376 014E 011A                    .DW PLUPE-10
000377 0150 0152            PDO:    .DW * + 2
000378 0152 29                      DEC R9
000379 0153 29                      DEC R9
000380 0154 E2                      SEX R2
000381 0155 49                      LDA R9
000382 0156 73                      STXD
000383 0157 49                      LDA R9
000384 0158 73                      STXD
000385 0159 49                      LDA R9
000386 015A 73                      STXD
000387 015B 09                      LDN R9
000388 015C 73                      STXD
000389 015D 29                      DEC R9
000390 015E 29                      DEC R9
000391 015F 29                      DEC R9
000392 0160 29                      DEC R9
000393 0161 29                      DEC R9
000394 0162 DC                      SEP RC          ; next
```

8.1.13. DIGIT

* *section id="ff1802.digit" xreflabel="DIGIT", xref.ff1802.digit, link.ff1802.DGT*

```
000397 0163 8544494749D4            .DB H’85,"DIGI",H’D4 ; DIGIT
000398 0169 0149                    .DW PDO-7
000399 016B 016D            DGT:    .DW * + 2
000400 016D E9                      SEX R9
000401 016E 29                      DEC R9
000402 016F 09                      LDN R9
000403 0170 FF30                    SMI H’30
000404 0172 3B88                    BNF BAD
000405 0174 FF11                    SMI H’11
000406 0176 337C                    BDF DOK
000407 0178 FFF9                    SMI H’F9
000408 017A 3388                    BDF BAD
000409 017C FC0A            DOK:    ADI H’0A
000410 017E 59                      STR R9
000411 017F 19                      INC R9
000412 0180 19                      INC R9
000413 0181 F7                      SM
000414 0182 3386                    BDF BAD2
000415 0184 29                      DEC R9
000416 0185 DC                      SEP RC
000417 0186 29              BAD2:   DEC R9
000418 0187 29                      DEC R9
000419 0188 F800            BAD:    LDI H’00
000420 018A 73                      STXD
000421 018B 59                      STR R9
000422 018C DC                      SEP RC          ; next
```

8.1.14. (FIND)

* *section id="ff1802.paren-find-paren" xreflabel="(FIND)", xref.ff1802.paren-find-paren, link.ff1802.FIND*

```
000424 018D 862846494E44A9          .DB H'86,"(FIND",H'A9 ; (FIND)
000425 0194 0163                    .DW DGT-8
000426 0196 0198            FIND:   .DW * + 2
000427 0198 29                      DEC R9
000428 0199 29                      DEC R9
000429 019A 49                      LDA R9
000430 019B B8                      PHI R8
000431 019C 49                      LDA R9
000432 019D A8                      PLO R8
000433 019E 49                      LDA R9
000434 019F B7                      PHI R7
000435 01A0 09                      LDN R9
000436 01A1 A7                      PLO R7
000437 01A2 29                      DEC R9
000438 01A3 29                      DEC R9
000439 01A4 E7              LOOP1:  SEX R7              ; SAVE LENGTH BYTE
000440 01A5 07                      LDN R7
000441 01A6 52                      STR R2
000442 01A7 48                      LDA R8              ; COMPARE LENGTH BYTES
000443 01A8 F3                      XOR
000444 01A9 FA3F                    ANI H'3F
000445 01AB 3AD3                    BNZ BADLEN
000446 01AD 17              NEXCHR: INC R7
000447 01AE 48                      LDA R8              ; COMPARE NEXT CHARACTER
000448 01AF F3                      XOR
000449 01B0 FE                      SHL
000450 01B1 3AD4                    BNZ BADCHR          ; NO MATCH ON 7 BITS
000451 01B3 7E                      SHLC
000452 01B4 32AD                    BZ NEXCHR           ; IF NOT LAST CHARACTER
000453 01B6 47              LOOP2:  LDA R7              ;   ELSE END OF STRING
000454 01B7 FA80                    ANI H'80
000455 01B9 32B6                    BZ LOOP2
000456 01BB E9                      SEX R9              ; END OF DICT NAME
000457 01BC 87                      GLO R7
000458 01BD FC04                    ADI H'04
000459 01BF 73                      STXD
000460 01C0 97                      GHI R7
000461 01C1 7C00                    ADCI H'00
000462 01C3 59                      STR R9              ; LEAVE PFA
000463 01C4 19                      INC R9
000464 01C5 19                      INC R9
000465 01C6 F800                    LDI H'00
000466 01C8 59                      STR R9
000467 01C9 19                      INC R9
000468 01CA 02                      LDN R2              ; GET LENGTH BYTE
000469 01CB 59                      STR R9
000470 01CC 19                      INC R9
000471 01CD F8FF                    LDI H'FF
000472 01CF 59                      STR R9
000473 01D0 19                      INC R9
000474 01D1 73                      STXD                ; AND TRUE FLAG
000475 01D2 DC                      SEP RC          ; next
000476 01D3 17              BADLEN: INC R7
000477 01D4 47              BADCHR: LDA R7
```

```
000478 01D5 FA80                    ANI H'80
000479 01D7 32D4                    BZ BADCHR
000480 01D9 07                      LDN R7
000481 01DA 3AE7                    BNZ BOK
000482 01DC 17                      INC R7
000483 01DD 07                      LDN R7
000484 01DE 27                      DEC R7
000485 01DF 3AE7                    BNZ BOK
000486 01E1 F800                    LDI H'00            ; LINK=0 RETURN FALSE
000487 01E3 59                      STR R9
000488 01E4 29                      DEC R9
000489 01E5 59                      STR R9
000490 01E6 DC                      SEP RC          ; next
000491 01E7 47              BOK:    LDA R7
000492 01E8 52                      STR R2
000493 01E9 07                      LDN R7
000494 01EA A7                      PLO R7
000495 01EB 02                      LDN R2
000496 01EC B7                      PHI R7
000497 01ED 09                      LDN R9
000498 01EE A8                      PLO R8
000499 01EF 29                      DEC R9
000500 01F0 49                      LDA R9
000501 01F1 B8                      PHI R8
000502 01F2 30A4                    BR LOOP1
```

8.1.15. ENCLOSURE

* *section id="ff1802.enclosure" xreflabel="ENCLOSURE", xref.ff1802.enclosure, link.ff1802.ENCL*

```
000505 01F4 87454E434C4F53C5        .DB H'87,"ENCLOS",H'C5 ; ENCLOSURE
000506 01FC 018D                    .DW FIND-9
000507 01FE 0200            ENCL:   .DW * + 2
000508 0200 29                      DEC R9
000509 0201 29                      DEC R9
000510 0202 49                      LDA R9
000511 0203 B8                      PHI R8
000512 0204 49                      LDA R9
000513 0205 A8                      PLO R8
000514 0206 19                      INC R9
000515 0207 F800                    LDI H'00            ; R7.0 IS OFFSET
000516 0209 A7                      PLO R7
000517 020A 09                      LDN R9              ; SAVE DELIM
000518 020B 52                      STR R2
000519 020C E2                      SEX R2
000520 020D 08              LOP1:   LDN R8
000521 020E F7                      SM
000522 020F 3A15                    BNZ FRST            ; FIND FIRST NON-
000523 0211 18                      INC R8              ;   DELIM CHAR
000524 0212 17                      INC R7
000525 0213 300D                    BR LOP1
000526 0215 87              FRST:   GLO R7              ; SAVE OFFSET TO
000527 0216 59                      STR R9
000528 0217 B7                      PHI R7
000529 0218 19                      INC R9              ;   FIRST CHARACTER
```

```
000530 0219 F800                    LDI H’00
000531 021B 59                      STR R9
000532 021C 19                      INC R9
000533 021D 19                      INC R9
000534 021E 59                      STR R9
000535 021F 19                      INC R9
000536 0220 08              LOP2:   LDN R8
000537 0221 322A                    BZ NULL             ; EQUAL NULL ?
000538 0223 F7                      SM                  ; SUBTRACT DELIMIN
000539 0224 3238                    BZ DELIM
000540 0226 18                      INC R8
000541 0227 17                      INC R7
000542 0228 3020                    BR LOP2
000543 022A 87              NULL:   GLO R7              ; LEAVE OFFSET
000544 022B E9                      SEX R9
000545 022C 59                      STR R9
000546 022D 97                      GHI R7              ; TO NEXT CHARACTER
000547 022E F7                      SM
000548 022F 3A32                    BNZ SKIP
000549 0231 17                      INC R7
000550 0232 87              SKIP:   GLO R7              ; LAST CHARACTER IN
000551 0233 29                      DEC R9
000552 0234 29                      DEC R9
000553 0235 59                      STR R9              ; WORD
000554 0236 19                      INC R9
000555 0237 DC                      SEP RC          ; next
000556 0238 17              DELIM:  INC R7
000557 0239 87                      GLO R7
000558 023A 59                      STR R9
000559 023B 27                      DEC R7
000560 023C 3032                    BR SKIP
```

8.1.16. CMOVE

* *section id="ff1802.cmove" xreflabel="CMOVE", xref.ff1802.cmove, link.ff1802.CMOVE*

```
000562 023E 85434D4F56C5            .DB H’85,"CMOV",H’C5 ; CMOVE
000563 0244 01F4                    .DW ENCL-10
000564 0246 0248            CMOVE:  .DW * + 2
000565 0248 49                      LDA R9
000566 0249 FC01                    ADI H’01
000567 024B B7                      PHI R7
000568 024C 09                      LDN R9
000569 024D A7                      PLO R7
000570 024E 27                      DEC R7
000571 024F 29                      DEC R9
000572 0250 29                      DEC R9
000573 0251 29                      DEC R9
000574 0252 9A                      GHI RA              ; PUSH RA
000575 0253 52                      STR R2
000576 0254 22                      DEC R2
000577 0255 8A                      GLO RA
000578 0256 52                      STR R2
000579 0257 49                      LDA R9              ; RA IS "TO"
000580 0258 BA                      PHI RA
```

```
000581 0259 09                      LDN R9
000582 025A AA                      PLO RA
000583 025B 29                      DEC R9
000584 025C 29                      DEC R9
000585 025D 29                      DEC R9
000586 025E 49                      LDA R9              ; R8 IS "FROM"
000587 025F B8                      PHI R8
000588 0260 09                      LDN R9
000589 0261 A8                      PLO R8
000590 0262 29                      DEC R9
000591 0263 29                      DEC R9
000592 0264 29                      DEC R9
000593 0265 97              LUUP:   GHI R7
000594 0266 326E                    BZ END2
000595 0268 48                      LDA R8
000596 0269 5A                      STR RA
000597 026A 1A                      INC RA
000598 026B 27                      DEC R7
000599 026C 3065                    BR LUUP
000600 026E 42              END2:   LDA R2              ; POP RA
000601 026F AA                      PLO RA
000602 0270 02                      LDN R2
000603 0271 BA                      PHI RA
000604 0272 DC                      SEP RC          ; next
```

8.1.17. U*

* *section id="ff1802.ustar" xreflabel="U*", xref.ff1802.ustar, link.ff1802.USTAR*

```
000606 0273 8255AA                  .DB H'82,H'55,H'AA  ; U*
000607 0276                                             ; UNSIGNED 16 X 16 BIT MULTIPLY
000608 0276 023E                    .DW CMOVE-8
000609 0278 027A            USTAR:  .DW * + 2
000610 027A E9                      SEX R9
000611 027B F800                    LDI R0
000612 027D A7                      PLO R7              ; R7 IS LOW 2 BYTES
000613 027E B7                      PHI R7
000614 027F F810                    LDI H'10            ; OF PRODUCT
000615 0281 52              LP7B:   STR R2              ; MEM(2) IS LOOP COUNT
000616 0282 87                      GLO R7
000617 0283 FE                      SHL
000618 0284 A7                      PLO R7
000619 0285 97                      GHI R7
000620 0286 7E                      SHLC
000621 0287 B7                      PHI R7
000622 0288 19                      INC R9              ; DOUBLE THE PRODUCT AND
000623 0289 09                      LDN R9              ;   TEST HIGH BIT
000624 028A 7E                      SHLC
000625 028B 73                      STXD
000626 028C 09                      LDN R9              ;   OF OP2
000627 028D 7E                      SHLC
000628 028E 59                      STR R9
000629 028F 3BA0                    BNF SKP9A
000630 0291 29                      DEC R9
000631 0292 87                      GLO R7
```

```
000632 0293 F4                      ADD
000633 0294 A7                      PLO R7
000634 0295 29                      DEC R9              ;   ADD OP1
000635 0296 97                      GHI R7
000636 0297 74                      ADC
000637 0298 B7                      PHI R7
000638 0299 19                      INC R9              ; TO 24 BIT PRODUCT
000639 029A 19                      INC R9
000640 029B 19                      INC R9
000641 029C F800                    LDI H'00
000642 029E 74                      ADC
000643 029F 73                      STXD
000644 02A0 02              SKP9A:  LDN R2
000645 02A1 FF01                    SMI H'01
000646 02A3 3A81                    BNZ LP7B
000647 02A5 29              UOUT:   DEC R9              ; MOVE REST OF
000648 02A6 87                      GLO R7
000649 02A7 73                      STXD
000650 02A8 97                      GHI R7              ; PRODUCT TO STACK
000651 02A9 59                      STR R9
000652 02AA 19                      INC R9
000653 02AB 19                      INC R9
000654 02AC DC                      SEP RC          ; next
```

8.1.18. U/

* *section id="ff1802.uslash" xreflabel="U/", xref.ff1802.uslash, link.ff1802.USLSH*

```
000656 02AD 8255AF                  .DB H'82,H'55,H'AF  ; U/ UNSIGNED DIVIDE
000657 02B0 0273                    .DW USTAR-5
000658 02B2 02B4            USLSH:  .DW * + 2
000659 02B4 E9                      SEX R9
000660 02B5 29                      DEC R9
000661 02B6 29                      DEC R9
000662 02B7 49                      LDA R9
000663 02B8 B7                      PHI R7
000664 02B9 09                      LDN R9
000665 02BA A7                      PLO R7
000666 02BB 29                      DEC R9
000667 02BC 29                      DEC R9
000668 02BD 49                      LDA R9
000669 02BE FE                      SHL
000670 02BF 19                      INC R9
000671 02C0 73                      STXD
000672 02C1 29                      DEC R9
000673 02C2 29                      DEC R9
000674 02C3 49                      LDA R9
000675 02C4 7E                      SHLC
000676 02C5 19                      INC R9
000677 02C6 59                      STR R9
000678 02C7 19                      INC R9
000679 02C8 F810                    LDI H'10
000680 02CA A8                      PLO R8
000681 02CB 87              LPC5:   GLO R7
000682 02CC 7E                      SHLC
```

```
000683 02CD A7                      PLO R7
000684 02CE 97                      GHI R7
000685 02CF 7E                      SHLC
000686 02D0 B7                      PHI R7
000687 02D1 19                      INC R9
000688 02D2 19                      INC R9
000689 02D3 87                      GLO R7
000690 02D4 F7                      SM
000691 02D5 B8                      PHI R8
000692 02D6 29                      DEC R9
000693 02D7 97                      GHI R7
000694 02D8 77                      SMB
000695 02D9 3BDE                    BNF SKPD8
000696 02DB B7                      PHI R7
000697 02DC 98                      GHI R8
000698 02DD A7                      PLO R7
000699 02DE 29              SKPD8:  DEC R9
000700 02DF 09                      LDN R9
000701 02E0 7E                      SHLC
000702 02E1 73                      STXD
000703 02E2 09                      LDN R9
000704 02E3 7E                      SHLC
000705 02E4 59                      STR R9
000706 02E5 19                      INC R9
000707 02E6 28                      DEC R8
000708 02E7 88                      GLO R8
000709 02E8 3ACB                    BNZ LPC5
000710 02EA 29                      DEC R9
000711 02EB 30A5                    BR UOUT
```

8.1.19. AND

* *section id="ff1802.and" xreflabel="AND", xref.ff1802.and, link.ff1802.FAND*

```
000713 02ED 83414EC4                .DB H'83,"AN",H'C4  ; AND
000714 02F1 02AD                    .DW USLSH - 5
000715 02F3 02F5            FAND:   .DW * + 2
000716 02F5 89                      GLO R9
000717 02F6 A8                      PLO R8
000718 02F7 99                      GHI R9
000719 02F8 B8                      PHI R8
000720 02F9 28                      DEC R8
000721 02FA E8                      SEX R8
000722 02FB 19                      INC R9
000723 02FC 09                      LDN R9
000724 02FD F2                      AND
000725 02FE 73                      STXD
000726 02FF 29                      DEC R9
000727 0300 09                      LDN R9
000728 0301 F2                      AND
000729 0302 58                      STR R8
000730 0303 29                      DEC R9
000731 0304 29                      DEC R9
000732 0305 DC                      SEP RC          ; next
```

8.1.20. Neanalyzovaná část

FIXME:od ....

8.1.21. + — *plus*

* *section id="ff1802.plus" xreflabel="+", xref.ff1802.plus, link.ff1802.PLUS*

Funkce sečte dvě hodnoty na vrcholu zásobníku (TOS a NOS), obě odstraní a místo nich uloží na zásobník výsledek. Zásobníkový efekt se dá tedy popsat takto

(a b ⟶ a+b)

Nejdříve se podíváme na hlavičku. Protože slovo `+` je definováno jako nízkoúrovňové a je přímo popsáno strojovým kódem, ukazuje `CFA` na `PF`.

```
000919 03F3 81AB                    .DW H'81AB          ; +
000920 03F5 03E6                    .DW ZLESS - 5
000921 03F7 03F9            PLUS:   .DW * + 2; CFA⟶PF containg machine code
```

V první části kódu se do `PSP` zkopíruje do pomocného registru `R8`.

```
                                    ; PSP⟶R8
000922 03F9 89                      GLO R9
000923 03FA A8                      PLO R8
000924 03FB 99                      GHI R9
000925 03FC B8                      PHI R8
```

V této chvíli vypadá stav zásobníku a registrů `PSP` a `R8` jak je ukázáno na obrázku 8-1 část a). Oba dva registry `PSP` i `R8` ukazují na stejný bajt a to horní část buňky `TOS`. Následuje posunití obou ukazatelů, `R8` dolů a `PSP` nahoru tak že oba ukazují na dolní bajty `NOS` a `TOS`. Registr `R8` je také nastaven jako indexový registr.

```
                                    ; now both PSP and R8 points to TOS, as you
                                    ; can see in figure 8-1 part a).
000926 03FD 28                      DEC R8          ; R8-=1
000927 03FE E8                      SEX R8          ; R8 is index register
000928 03FF 19                      INC R9          ; PSP+=1; points to TOS.lo
```

Stav zásobníku nyní odpovídá obrázku 8-1 část b). Následuje sečtení dolních bajtů `TOS` a `NOS` a jejich uložení do dolního bajtu `NOS`.

```
                                    ; PSP points to TOS.lo and R8 to NOS.lo as
                                    ; you can see in figure 8-1 part b).
000929 0400 09                      LDN R9          ; (PSP)⟶D ; D=TOS.lo
000930 0401 F4                      ADD             ; D,DF⟵TOS.lo+NOS.lo
000931 0402 73                      STXD            ; ⟶(R8-); store to NOS.lo
000932 0403 29                      DEC R9          ; PSP points to TOS.hi
```

V tomto okamžiku ukazují `PSP` na `TOS.hi` a `R8` na `NOS.hi`. V `NOS.lo` je již uložen součet dolních bajtů `TOS` a `NOS`. Následuje sečtení horních bajtů `TOS` a `NOS` a příznaku `DF` přetečení předchozí operace sčítání. Výsledek je uložen do `NOS.hi` na který ukazuje registr `R8`.

```
                                    ; PSP points to TOS.hi and R8 to NOS.hi as
                                    ; displayed in figure 8-1 part c).
000933 0404 09                      LDN R9          ; (PSP)⟶D ; D=TOS.hi
000934 0405 74                      ADC             ; D,DF⟵DF+TOS.hi+NOS.hi
000935 0406 58                      STR R8          ; ⟶(R8)
```

Následuje poslední část kódu, korekce PS. Posuneme PSP o dvě pozice dolů, takže ukazuje na dřívější NOS, nynější nové TOS. Stav zásobníku v této chvíli ukazuje obrázek 8-1 část d). Jako poslední instrukce je přepnutí na smyčku vnitřního interpretu které ukončuje kód slova +.

```
000936 0407 29                      DEC R9          ; PSP-=2
000937 0408 29                      DEC R9
                                    ; the old TOS is droped and new contains
                                    ; the sum.  See figure 8-1 part d).
000938 0409 DC                      SEP RC          ; next
```

Následující obrázek ukazuje stav zásobníku v jednotlivých fázích.

Obrázek 8-1. Stav zásobníku při vykonávání slova +

```
            a)                b)                c)                d)
         +--------+        +--------+        +--------+        +--------+
         | TOS.lo |   PSP->| TOS.lo |        | TOS.lo |        |        |
R8,PSP->| TOS.hi |        | TOS.hi |   PSP->| TOS.hi |        |--------|
         | NOS.lo |    R8->| NOS.lo |        | NOS.lo |        | TOS.lo |
         | NOS.hi |        | NOS.hi |    R8->| NOS.hi |   PSP->| TOS.hi |
         |        |        |        |        |        |        | NOS.lo |
```

8.1.22. !CSP

* *section id="fig-forth-1802.to-csp" xreflabel="!CSP"*

```
001692 08A4 84214353D0              .DB H’84,H’21,H’43,H’53,H’D0 ; !CSP
001693 08A9 0890                    .DW PFA - 6
001694 08AB 05C2            DCSP:   .DW NEST
001695 08AD 033D                    .DW FSPAT
001696 08AF 0743                    .DW CSP
001697 08B1 051E                    .DW EX
001698 08B3 0379                    .DW SEMIS
```

8.1.23. Neanalyzováno

V této části se vyskytují definice slov: MINUS, D+, DMINUS, OVER, DROP, SWAP, DUP, +!, TOGGLE, @, C@, !, C!, EMIT, KEY, ?TERMINAL, CR, nest a VAR

8.1.24. CONST

* *section id="ff1802.const" xreflabel="CONST"*

Toto slovo se nepoužívá přímo ale je součástí definice konstanty CONSTANT. Definujeme-li tedy nějakou konstantu příkazem:

číslo CONSTANT *název*

vytvoří se ve slovníku záznam

hlavička se jménem
DW *link na předchozí slovo ve slovníku*

DW CONST
DW *číslo*

Kód slova přečte „následující“ buňku a uloží její obsah na zásobník parametrů PS.

```
CONST:  ; (W+)⟶(+PS)
        INC R9          ; PSP+=1
        INC R9          ; PSP+=1
        LDA RB          ; (W+)
        STR R9          ; ⟶(PSP)
        INC R9          ; PSP+=1
        LDA RB          ; (W+)
        STR R9          ; ⟶(PSP)
        DEC R9          ; PSP-=1
        SEP RC          ; next
```

8.1.25. Neanalyzováno

V této části se vyskytují definice slov: USER

8.1.26. Definice konstant

Definicemi konstant začíná část kódu jenž je psána jako odkazy na již existující slova. Tedy tuto část je již možno definovat ve FORTHu bez použití strojového kódu.

Některé z konstant jsou implementačně závislé. Odráží charakteristiky daného sytému a umožňují nám psát přenositelné programy. Například blok na disku. Na různých systémech může mít různou velikost ale nám je známa jeho velikost definovaná konstantou B/BUF.

Ve slovníku jsou definovány následující konstanty.

0

Slovo **0** uloží na záspobník parametrů číslo 0. Definice ve FORTHu s pomocí slova CONSTANT vypadá takto:

0 CONSTANT 0

A vytvořený záznam ve slovníku zase takto:

```
        DW 081B0h       ; 0
        DW CR-5         ; link to previous definition
ZERO:   DW CONST
        DW 00000h
```

1

Definice konstanty **1**.

Záznam ve slovníku.

```
        DW 081B1h       ; 1
        DW 0-4       ; link to previous definition in dictionary
ONE:    DW CONST
        DW 00001h
```

2

Definice konstanty **2**.

Záznam ve slovníku.

```
        DW 081B2h       ; 2
        DW ONE-4        ; link to previous record in dictionary
TWO:    DW CONST
        DW 00002h
```

BL

Konstanta **BL** je prázdný znak (mezera, space).

32 CONSTANT BL

```
        DB 082h,042h,0CCh  ; BL
        DW TWO-4         ; CONSTANT ASCII BLANK
BL:     DW CONST
        DW 00020h
```

C/L

Tato konstanta určuje počet znaků na řádku terminálu.

64 CONSTANT C/L

```
        DB 083h,043h,02Fh,0CCh ; C/L
                               ; CHARACTERS PER LINE
        DW BL-5
CL:     DW CONST
        DW 00040h          ; 64 (DECIMAL)
```

FIRST

FIXME:

16384 CONSTANT FIRST

```
        DB 085h,"FIRS",0D4h ; FIRST
        DW CL-6
FIRST:  DW CONST
        DW FIRSTB       ; 04000h adresa prvního diskového bufferu
```

LIMIT

FIXME:

27692 CONSTANT LIMIT

```
        DB 085h,"LIMI",0D4h ; LIMIT
        DW FIRST-8
LIMIT:  DW CONST
        DW LIMITB       ; 6C2Ch -- konec oblasti diskových bufferů
```

B/BUF

Velikost diskového bloku. Data na disku jsou ukládána v blocích stejné délky. Tato konstanta definuje jak veliký je tento blok v bytech.

1024 CONSTANT B/BUF

```
        DB 085h,"B/BU",0C6h ; B/BUF
                        ; BYTES PER BUFFER
        DW LIMIT-8
```

```
BBUF:   DW CONST
        DW 00400h          ; 1024 BYTES/BUFFER
```

B/SCR

Počet dakových bloků jenž je potřeba na uložení jedné obrazovky..

1 CONSTANT B/SCR

```
        DB 085h,"B/SC",0D2h ; B/SCR
                            ; BUFFERS/SCREEN
        DW BBUF-8
BSCR:   DW CONST
        DW 00001h
```

ORIGIN

FIXME:

94 CONSTANT ORIGIN

```
        DB 086h,"ORIGI",0CEh ; ORIGIN
        DW BSCR-8
ORGN:   DW CONST
        DW 0005Eh
```

8.1.27. +ORIGIN

: ORIGIN + ;

DB 087h,"+ORIGI",0CEh ; +ORIGIN

DW ORGN-9
PORGN: DW NEST
DW ORGN
DW PLUS
DW SEMIS

8.1.28. CONSTANT

```
        DB 088h,"CONSTAN",0D4h ; CONSTANT
        DW DCODE-8
CNST:   DW nest
        DW LIT
        DW CONST
        DW DCODE
        DW SEMIS
```

8.1.29. ; — *semicolon*

* *section id="ff1802.semicolon" xreflabel=";"*

Uzavírá definici slova započatou symbolem :.

```
: ; ?CSP CMPL

        DW 0C1BBh       ; ;   (IMMEDIATE)
        DW ABORT-8
SEMIC:  DW NEST
        DW QCSP
        DW CMPL
        DW SEMIS
        DW SMDG
        DW LB
        DW SEMIS
```

8.1.30. ?CSP

```
        DB 084h,03Fh,043h,053h,0D0h ; ?CSP
        DW QPR-9
QCSP:   DW NEST
        DW FSPAT
        DW CSP
        DW AT
        DW MINS
        DW LIT
        DW 00014h
        DW QERR
        DW SEMIS
```

8.1.31. Neanalyzovaná slova

8.1.31.1.]

* *section id="fig-forth-1802-right-bracket" xreflabel="]"*

```
        DW 081DDh       ; ]   RIGHT BRACKET
        DW LB-4
RBK:    DW NEST
        DW LIT
        DW 000C0h
        DW STT
        DW EX
        DW SEMIS
```

8.1.31.2. DP

* *section id="ff1802.dp" xreflabel="DP"*

```
        DB 082h,044h,0D0h  ; DP
        DW FNCE-8
DP:     DW USER
```

```
        DW 00012h
```

8.1.31.3. +!

* *section id="ff1802.plus-store" xreflabel="+!"*

```
        DB 082h,02Bh,0A1h  ; +!
        DW DUP-6
PLUSS:  DW $+2
        LDA R9
        PHI R8
        LDN R9
        PLO R8
        DEC R9
        DEC R9
        INC R8
        SEX R8
        LDN R9
        ADD
        STXD
        DEC R9
        LDN R9
        ADC
        STR R8
POP:    DEC R9
        DEC R9
        SEP RC
```

8.1.31.4. COMPILE

* *section id="ff1802.compile" xreflabel="COMPILE"*

```
        DB 087h,"COMPIL",0C5h ; COMPILE
 DW QLDG-11
CMPL:   DW NEST
        DW QCMP
        DW RG
        DW DUP
        DW PLUS2
        DW GR
        DW AT
        DW COMMA
        DW SEMIS
```

8.1.31.5. CREATE

* *section id="fig-forth-1802.create" xreflabel="CREATE"*

```
            DB 086h,"CREAT",0C5h ; CREATE
            DW ID-6
CRTE:       DW NEST
            DW FSPAT
            DW HERE
            DW LIT
            DW 000A0h
            DW PLUS
            DW LESS
            DW TWO
            DW QERR
            DW MFIND
            DW ZBRCH
            DW CRT1
            DW DROP
            DW NFA
            DW ID
            DW LIT
            DW 00004h
            DW MSG
            DW SPC
CRT1:       DW HERE
            DW DUP
            DW CAT
            DW WIDTH
            DW AT
            DW MIN
            DW PLUS1
            DW ALLOT
            DW DUP
            DW LIT
            DW 000A0h
            DW TGLE
            DW HERE
            DW ONE
            DW MINS
            DW LIT
            DW 00080h
            DW TGLE
            DW LTST
            DW COMMA
            DW CRNT
            DW AT
            DW EX
            DW HERE
            DW PLUS2
            DW COMMA
            DW SEMIS
```

8.1.31.6. ?EXEC

* *section id="fig-forth-1802.Qexec" xreflabel="?EXEC"*

```
            DB 085h,"?EXE",0C3h ; ?EXEC
            DW QCMP-8
    EXC:    DW NEST
            DW STT
            DW AT
            DW LIT
            DW 00012h
            DW QERR
            DW SEMIS
```

8.1.31.7. CONTEXT

* *section id="fig-forth-1802.context" xreflabel="CONTEXT"*

```
        DB 087h,"CONTEX",0D4h ; CONTEXT
     DW OFST-9
    CNTX:   DW USER
        DW 00020h
```

8.1.31.8. CURRENT

* *section id="fig-forth-1802.current" xreflabel="CURRENT"*

```
        DB 087h,"CURREN",0D4h ; CURRENT
     DW CNTX-10
    CRNT:   DW USER
        DW 00022h
```

8.1.31.9. @

* *section id="fig-forth-1802.at" xreflabel="@"*

```
        DW 081C0h       ; @
        DW TGLE-9
    AT:   DW $+2
        LDA R9
        PHI R8
        LDN R9
        PLO R8
        DEC R9
        LDA R8
        STR R9
        INC R9
        LDN R8
        STR R9
        DEC R9
        SEP RC
```

8.1.31.10. !

* *section id="fig-forth-1802.store" xreflabel="!"*

```
        DW 081A1h       ; !   STORE
        DW CAT-5
EX:     DW $+2
        LDA R9
        PHI R8
        LDN R9
        PLO R8
        DEC R9
        DEC R9
        DEC R9
        LDA R9
        STR R8
        INC R8
        LDN R9
        STR R8
        DEC R9
        DEC R9
        DEC R9
        SEP RC
```

8.1.32. Nepopsaná slova

Další slova jenž jsem blíže nepopsal, bez nároku na pořadí jsou: **FIND**, **ENCLOSURE**, **CMOVE**, **U***, **U/**, **AND**, **OR**, **XOR**, **SP@**, **SP!**, **RP!**, **>R**, **R>**, **R**, **0=**, **0<**, **+**, **MINUS**, **D+**, **DMINUS**, **OVER**, **DROP**, **SWAP**, **DUP**, **+!**, **TOGGLE**, **@**, **C@**, **!**, **C!**, **EMIT**, **KEY**, **?TERMINAL**, **CR**, **VAR**, **CONST**, ...

8.1.33. Zbývá analyzovat

Následující výpis obsahuje doposud neanalyzovaný kód. Jednotlivé části po analýze z výpisu vyjímám.

```
000001 0000                     ;
000002 0000                     ; FFFF  I   GGG         FFFF   OO   RRRR   TTTTT  H   H
000003 0000                     ; F     I  G            F     O  O  R  R     T    H   H
000004 0000                     ; FFF   I  G GGG   XX   FFF   O  O  RRR      T    HHHHH
000005 0000                     ; F     I  G   G        F     O  O  R  R     T    H   H
000006 0000                     ; F     I   GGG         F      OO   R   R    T    H   H
000007 0000                     ;
000008 0000                     ;                1   8888    00   2222
000009 0000                     ;                1   8  8   0  0     2
000010 0000                     ;                1   8888   0  0   22
000011 0000                     ;                1   8  8   0  0  2
000012 0000                     ;                1   8888    00   2222
000013 0000                     ;
000014 0000                     ;
000015 0000                     ;       ALL PUBLICATIONS OF THE FORTH INTREST GROUP
000016 0000                     ;       ARE PUBLIC DOMAIN.  THEY MAY BE FURTHER
000017 0000                     ;       DISTRIBUTED BY INCLUSION OF THIS CREDIT
```

```
000018 0000                    ;       NOTICE:
000019 0000                    ;
000020 0000                    ;       THIS PUBLICATION HAS BEEN MADE AVAILABLE BY THE
000021 0000                    ;                       FORTH INTREST GROUP
000022 0000                    ;                       P. O. BOX 1105
000023 0000                    ;                       SAN CARLOS, CA 94070
000024 0000                    ;
000025 0000                    ;       IMPLEMENTAION BY:
000026 0000                    ;               GARY R. BRADSHAW
000027 0000                    ;               RFD 1 BOX 80
000028 0000                    ;               GIDLEY ROAD
000029 0000                    ;               ESPERANCE, NY  12066
000030 0000                    ;
000031 0000                    ;       MODIFIED BY:
000032 0000                    ;               GORDEN FLEMMING
000033 0000                    ;               13490 SIMSHAW ST.
000034 0000                    ;               SYLMAR, CA  91342
000035 0000                    ;
000036 0000                    ;               JIM MCDANIEL
000037 0000                    ;               1109 POINCIANA DR.
000038 0000                    ;               SUNNYVALE, CA 94086
000039 0000                    ;
000040 0000                    ;   ADDRESS COMMENTS & CORRECTIONS TO  JIM MCDANIEL.
000041 0000                    ;
000042 0000                    ;       ACKNOWLEDGEMENTS:
000043 0000                    ;
000044 0000                    ;               KEN MANTEI
000045 0000                    ;
000046 0000                    ;               FIG INSTALLATION MANUAL
000047 0000                    ;
000048 0000                    ;               FIG 8080 ASSEMBLY SOURCE LISTING..;
000049 0000                    ;
000050 0000                    ;
000051 0000                    ;               THIS LISTING TYPED     PRINTED 3/18/81
000052 0000                    ;
000053 0000                    ;
000054 0000                    ;       THE I/O VECTORS FOR DISC  ARE POINTING TO
000055 0000                    ;       ROUTINES FOR THE RCA CDP18S007, CDP18S008
000056 0000                    ;       OR THE CDP 18SOO5..
000057 0000                    ;       FOR OTHER SYSTEMS YOU WILL NEED TO
000058 0000                    ;       CHANGE THE POINTERS AND WRITE YOUR
000059 0000                    ;       OWN ROUTINES.;
000060 0000                    ;
000061 0000                    ; THE USER VARIABLE  DV IS 3 BYTES LONG
000062 0000                    ; (USER AREA OFFSET 32H 33H 34H) AND IS USED
000063 0000                    ; TO PASS VARIABLES TO THE RCA ROM UTILITY
000064 0000                    ; WHEN CALLING FOR DISK I/O.  THE RESIDENT ROM
000065 0000                    ; UTILITIES PRODUCE THEIR OWN ERROR MESSAGES.
000066 0000                    ;
000067 0000                    ; THIS VERSION ASSEMBLES WITH START UP CONSTANTS THAT
000068 0000                    ; ASSUME 28K OF RAM. DISC BUFFERS ARE
000069 0000                    ; SET FOR 1K (THIS CAN BE EASILY CHANGED
000070 0000                    ; BY CHANGING FIRST, LIMIT, B/BUF AND B/SCR).
000071 0000                    ; BLOCK 0 BEGINS AT TRACK 0 SECTOR 1.
000072 0000                    ;
000073 0000                    ;
000074 0000                    ; DR1 (SET DRIVE TO 1) IS IMPLEMENTED AS:
```

```
000075 0000                ;  : DR1 B/SCR 250 * OFFSET ! ;
000076 0000                ; THERFORE B/SCR AND B/BUF CAN BE CHANGED WITHOUT
000077 0000                ; HAVING TO REWRITE
000078 0000                ; DR1.
000079 0000                ; THE TERMINAL I/O ASSUMES THE USE OF AN
000080 0000                ; RCA CDP1854 UART CONFIGURED IN ONE OF THE
000081 0000                ; ABOVE MENTIONED SYSTEMS.  THE UART IS DRIVEN
000082 0000                ; DIRECTLY WITHOUT CALLING RCA ROM UTILITY
000083 0000                ; ROUTINES.
000084 0000                ;
000085 0000                ;
000086 0000                ; THE FORTH WORD   MON   EXITS FORTH AND RETURNS
000087 0000                ; TO THE RESIDENT ROM UTILITY.
000088 0000                ;
000089 0000                ; THE FORTH WORD  BYE  IS DEFINED AS:
000090 0000                ; : BYE  FLUSH  MON ;;
000091 0000                ;
000092 0000                ;
000093 0000                ;       REGISTER ALLOCATIONS FOR THIS VERSION
000094 0000                ;
000095 0000                ;               R2   RETURNS STACK POINTER  R0
000096 0000                ;                    GROWS DOWN LEFT POINTING TO
000097 0000                ;                    FREE LOCATION
000098 0000                ;
000099 0000                ;               R3   PC FOR I/O AND PRIMITIVES
000100 0000                ;
000101 0000                ;               R7, R8  TEMPORARY ACCUMULATORS
000102 0000                ;
000103 0000                ;               R9   COMPUTATION STACK POINTER   S0
000104 0000                ;                    GROWS UPWARD
000105 0000                ;                    LEFT POINTING AT HIGH BYTE
000106 0000                ;
000107 0000                ;               RA   FORTH "I" REGISTER   IP
000108 0000                ;
000109 0000                ;               RB   FORTH "W" REGISTER   WP
000110 0000                ;
000111 0000                ;               RC   PC FOR INNER INTERPRETER
000112 0000                ;
000113 0000                ;               RD   USER POINTER         UP
000114 0000                ;
000115 0000                ;               RF   DISC I/O
000116 0000                ;
000117 0000                ;               OTHER REGISTERS ARE LEFT AVAILABLE
000118 0000                ;               EXCEPT THAT RF.0 IS ZERO AFTER COLD
000119 0000                ;               OR WARM STARTS
000120 0000                ;
000121 0000                ;
000122 0000                ;             MEMORY MAP
000123 0000                ;
000124 0000                ;       ---------------------- LIMIT
000125 0000                ;
000126 0000                ;            RAM BUFFERS
000127 0000                ;
000128 0000                ;       ---------------------- FIRST
000129 0000                ;
000130 0000                ;             USER AREA
000131 0000                ;
```

```
000132 0000                     ;       ---------------------- UP
000133 0000                     ;
000134 0000                     ;            RETURN STACK      R0
000135 0000                     ;
000136 0000                     ;       ----------------------
000137 0000                     ;
000138 0000                     ;          TERMINAL BUFFER
000139 0000                     ;
000140 0000                     ;       ---------------------- TIB
000141 0000                     ;
000142 0000                     ;         COMPUTATION STACK
000143 0000                     ;
000144 0000                     ;       ---------------------- S0
000145 0000                     ;
000146 0000                     ;             FREE SPACE
000147 0000                     ;
000148 0000                     ;       ----------------------
000149 0000                     ;
000150 0000                     ;            TEXT BUFFER
000151 0000                     ;
000152 0000                     ;       ---------------------- PAD
000153 0000                     ;
000154 0000                     ;             WORD BUFFER
000155 0000                     ;
000156 0000                     ;       ---------------------- DP
000157 0000                     ;
000158 0000                     ;             DICTIONARY
000159 0000                     ;
000160 0000                     ;       ----------------------
000161 0000                     ;
000162 0000                     ;         BOOT UP PARAMETERS
000163 0000                     ;
000164 0000                     ;       ---------------------- ORGIN 005E
000165 0000                     ;
000166 0000                     ;          ANY REQUIRED I/O
000167 0000                     ;           INITIALIZATION
000168 0000                     ;
000169 0000                     ;       ---------------------- 0000
000170 0000                     ;
000171 0000                     ;
000172 4000                             .EQU FIRSTB, H'4000 ; ADDRESS OF FIRST DISK BUFFER
000173 6C2C                             .EQU LIMITB, H'6C2C ; END OF DISK BUFFER AREA
000174 0009                             .EQU CSTACK, 9
000175 0002                             .EQU RSTACK, 2
000176 0000                      ;
000177 0000                             .ORG H'0000
000178 0000                             ;
000179 0000                             ; SET-UP ROUTINES
000180 0000                             ;
000181 0000 71                          DIS
000182 0001 00                          .DB H'00
000183 0002 61                          OUT 1               ; SET LEVEL 1 I/O
000184 0003 01                          .DB H'01
000185 0004 63                          OUT 3               ; SET UP UART
000186 0005 1D                          .DB H'1D
000187 0006 F800                        LDI H'00
000188 0008 B3                          PHI 3
```

```
000189 0009 F85E                    LDI H’5E
000190 000B A3                      PLO 3
000191 000C F82F                    LDI H’2F
000192 000E B2                      PHI 2
000193 000F F8FF                    LDI H’FF
000194 0011 A2                      PLO 2
000195 0012 D3                      SEP 3
000196 0013                  ;
000197 0013                  ;
000198 0013                         ;
000199 005E                         .ORG H’005E
000200 005E                  ;
000201 005E C4               START:  NOP
000202 005F C01900                   LBR COLD            ; COLD START
000203 0062 C4                       NOP
000204 0063 C01915                   LBR WARM            ; WARM START
000205 0066 070A                     .DW H’070A          ; CPU NUMBER
000206 0068 0001                     .DW H’0001          ; REVISION NUMBER
000207 006A 18BF                     .DW TASK - 7        ; TOPMOST PRGM IN FORTH VOCABULARY
000208 006C 0008                     .DW H’0008          ; BACKSPACE
000209 006E 2000                     .DW H’2000          ; INITIAL USER AREA         UP
000210 0070 1F00                     .DW H’1F00          ; INITAL STACK              S0
000211 0072 1FFF                     .DW H’1FFF          ; INITAL RETURN STACK       R0
000212 0074 1F80                     .DW H’1F80          ; TERMINAL BUFFER           TIB
000213 0076 001F                     .DW H’001F          ; NAME FIELD WIDTH          WIDTH
000214 0078                                              ;   (31 DECIMAL)
000215 0078 0000                     .DW H’0000          ; WARNING                   WARNING
000216 007A 193D                     .DW LEND            ; FENCE                     FENCE
000217 007C 193D                     .DW LEND            ; INIT DICTIONARY POINTER   DP
000218 007E 0F6A                     .DW FRTH + 16       ; INIT VOCAB                VOC-LINK
&vellip;
000734 0306 824FD2                   .DB H’82,H’4F,H’D2  ; OR
000735 0309 02ED                     .DW FAND - 6
000736 030B 030D             FFOR:   .DW * + 2
000737 030D 89                       GLO R9
000738 030E A8                       PLO R8
000739 030F 99                       GHI R9
000740 0310 B8                       PHI R8
000741 0311 28                       DEC R8
000742 0312 E8                       SEX R8
000743 0313 19                       INC R9
000744 0314 09                       LDN R9
000745 0315 F1                       OR
000746 0316 73                       STXD
000747 0317 29                       DEC R9
000748 0318 09                       LDN R9
000749 0319 F1                       OR
000750 031A 58                       STR R8
000751 031B 29                       DEC R9
000752 031C 29                       DEC R9
000753 031D DC                       SEP RC
000754 031E                  ;
000755 031E 83584FD2                 .DB H’83,"XO",H’D2   ; XOR
000756 0322 0306                     .DW FFOR - 5
000757 0324 0326             FXOR:   .DW * + 2
000758 0326 89                       GLO R9
000759 0327 A8                       PLO R8
```

```
000760 0328 99                      GHI R9
000761 0329 B8                      PHI R8
000762 032A 28                      DEC R8
000763 032B E8                      SEX R8
000764 032C 19                      INC R9
000765 032D 09                      LDN R9
000766 032E F3                      XOR
000767 032F 73                      STXD
000768 0330 29                      DEC R9
000769 0331 09                      LDN R9
000770 0332 F3                      XOR
000771 0333 58                      STR R8
000772 0334 29                      DEC R9
000773 0335 29                      DEC R9
000774 0336 DC                      SEP RC
000775 0337                  ;
000776 0337 835350C0                .DB H'83,"SP",H'C0  ; SP@
000777 033B 031E                    .DW FXOR - 6
000778 033D 033F            FSPAT:  .DW * + 2
000779 033F 99                      GHI R9
000780 0340 52                      STR R2
000781 0341 89                      GLO R9
000782 0342 19                      INC R9
000783 0343 19                      INC R9
000784 0344 19                      INC R9
000785 0345 59                      STR R9
000786 0346 29                      DEC R9
000787 0347 02                      LDN R2
000788 0348 59                      STR R9
000789 0349 DC                      SEP RC
000790 034A                  ;
000791 034A 835350A1                .DB H'83,"SP",H'A1  ; SP!
000792 034E                                             ;   stack pointer store
000793 034E 0337                    .DW FSPAT - 6
000794 0350 0352            SP1:    .DW * + 2
000795 0352 8D                      GLO RD
000796 0353 FC06                    ADI H'06
000797 0355 A8                      PLO R8
000798 0356 9D                      GHI RD
000799 0357 7C00                    ADCI H'00
000800 0359 B8                      PHI R8
000801 035A 48                      LDA R8
000802 035B B9                      PHI R9
000803 035C 08                      LDN R8
000804 035D A9                      PLO R9
000805 035E DC                      SEP RC
000806 035F                  ;
000807 035F 835250A1                .DB H'83,"RP",H'A1  ; RP!
000808 0363                                             ; RETURN STACK POINTER STORE
000809 0363 034A                    .DW SP1 - 6
000810 0365 0367            RP1:    .DW * + 2
000811 0367 8D                      GLO RD
000812 0368 FC08                    ADI R8
000813 036A A8                      PLO R8
000814 036B 9D                      GHI RD
000815 036C 7C00                    ADCI H'00
000816 036E B8                      PHI R8
```

```
000817 036F 48                      LDA R8
000818 0370 B2                      PHI R2
000819 0371 08                      LDN R8
000820 0372 A2                      PLO R2
000821 0373 DC                      SEP RC
&vellip;
000833 0381 854C454156C5            .DB H'85,"LEAV",H'C5 ; LEAVE
000834 0387 0374                    .DW SEMIS - 5
000835 0389 038B            LVE:    .DW * + 2
000836 038B 92                      GHI R2
000837 038C B8                      PHI R8
000838 038D 82                      GLO R2
000839 038E A8                      PLO R8
000840 038F 18                      INC R8
000841 0390 48                      LDA R8
000842 0391 18                      INC R8
000843 0392 58                      STR R8
000844 0393 28                      DEC R8
000845 0394 48                      LDA R8
000846 0395 18                      INC R8
000847 0396 58                      STR R8
000848 0397 DC                      SEP RC
000849 0398                  ;
000850 0398 823ED2                  .DB H'82,H'3E,H'D2  ; >R   TO R
000851 039B 0381                    .DW LVE - 8
000852 039D 039F            GR:     .DW * + 2
000853 039F E2                      SEX R2
000854 03A0 49                      LDA R9
000855 03A1 73                      STXD
000856 03A2 09                      LDN R9
000857 03A3 73                      STXD
000858 03A4 29                      DEC R9
000859 03A5 29                      DEC R9
000860 03A6 29                      DEC R9
000861 03A7 DC                      SEP RC
000862 03A8                  ;
000863 03A8 8252BE                  .DB H'82,H'52,H'BE  ; R>   FROM R
000864 03AB 0398                    .DW GR - 5
000865 03AD 03AF            RG:     .DW * + 2
000866 03AF 19                      INC R9
000867 03B0 19                      INC R9
000868 03B1 19                      INC R9
000869 03B2 12                      INC R2
000870 03B3 42                      LDA R2
000871 03B4 59                      STR R9
000872 03B5 29                      DEC R9
000873 03B6 02                      LDN R2
000874 03B7 59                      STR R9
000875 03B8 DC                      SEP RC
000876 03B9                  ;
000877 03B9 81D2                    .DW H'81D2          ; R   COPY TOP OF RETN
000878 03BB 03A8                    .DW RG - 5          ; STACK TO TOP OF
000879 03BD 03BF            R:      .DW * + 2           ; COMPUTATION STACK
000880 03BF 82                      GLO R2
000881 03C0 A8                      PLO R8
000882 03C1 92                      GHI R2
000883 03C2 B8                      PHI R8
```

```
000884 03C3 18                      INC R8
000885 03C4 19                      INC R9
000886 03C5 19                      INC R9
000887 03C6 19                      INC R9
000888 03C7 48                      LDA R8
000889 03C8 59                      STR R9
000890 03C9 29                      DEC R9
000891 03CA 08                      LDN R8
000892 03CB 59                      STR R9
000893 03CC DC                      SEP RC
000894 03CD                  ;
000895 03CD 8230BD                  .DB H'82,H'30,H'BD  ; 0=
000896 03D0 03B9                    .DW R - 4
000897 03D2 03D4            ZEQAL:  .DW * + 2
000898 03D4 49                      LDA R9
000899 03D5 3ADE                    BNZ NONE
000900 03D7 09                      LDN R9
000901 03D8 3ADE                    BNZ NONE
000902 03DA F801            ZONE:   LDI H'01
000903 03DC 30E0                    BR STOR
000904 03DE F800            NONE:   LDI H'00
000905 03E0 59              STOR:   STR R9
000906 03E1 29                      DEC R9
000907 03E2 F800                    LDI H'00
000908 03E4 59                      STR R9
000909 03E5 DC                      SEP RC
000910 03E6                  ;
000911 03E6 8230BC                  .DB H'82,H'30,H'BC  ; 0<
000912 03E9 03CD                    .DW ZEQAL - 5
000913 03EB 03ED            ZLESS:  .DW * + 2
000914 03ED 49                      LDA R9
000915 03EE FE                      SHL
000916 03EF 33DA                    BDF ZONE
000917 03F1 30DE                    BR NONE
&vellip;
000940 040A 854D494E55D3            .DB H'85,"MINU",H'D3 ; MINUS
000941 0410 03F3                    .DW PLUS - 4
000942 0412 0414            MINUS:  .DW * + 2
000943 0414 FF00                    SMI H'00            ; SET CARRY
000944 0416 19              MINOS:  INC R9
000945 0417 09                      LDN R9
000946 0418 FBFF                    XRI H'FF
000947 041A 7C00                    ADCI H'00
000948 041C 59                      STR R9
000949 041D 29                      DEC R9
000950 041E 09                      LDN R9
000951 041F FBFF                    XRI H'FF
000952 0421 7C00                    ADCI H'00
000953 0423 59                      STR R9
000954 0424 DC                      SEP RC
000955 0425                  ;
000956 0425 8244AB                  .DB H'82,H'44,H'AB  ; D+ DBL PRCN INTEGERS
000957 0428 040A                    .DW MINUS - 8       ; ARE STORED HIGH 16 BITS TOP
000958 042A 042C            DPLUS:  .DW * + 2           ; LOW 16 BITS BENEATH
000959 042C 89                      GLO R9
000960 042D FF05                    SMI H'05
000961 042F A8                      PLO R8
```

```
000962 0430 99                      GHI R9
000963 0431 7F00                    SMBI H'00
000964 0433 B8                      PHI R8
000965 0434 29                      DEC R9
000966 0435 E8                      SEX R8
000967 0436 09                      LDN R9
000968 0437 F4                      ADD
000969 0438 73                      STXD
000970 0439 29                      DEC R9
000971 043A 09                      LDN R9
000972 043B 74                      ADC
000973 043C 58                      STR R8
000974 043D 18                      INC R8
000975 043E 18                      INC R8
000976 043F 18                      INC R8
000977 0440 19                      INC R9
000978 0441 19                      INC R9
000979 0442 19                      INC R9
000980 0443 09                      LDN R9
000981 0444 74                      ADC
000982 0445 73                      STXD
000983 0446 29                      DEC R9
000984 0447 09                      LDN R9
000985 0448 74                      ADC
000986 0449 58                      STR R8
000987 044A 29                      DEC R9
000988 044B 29                      DEC R9
000989 044C 29                      DEC R9
000990 044D 29                      DEC R9
000991 044E DC                      SEP RC
000992 044F                  ;
000993 044F 86444D494E55D3          .DB H'86,"DMINU",H'D3 ; DMINUS
000994 0456 0425                    .DW DPLUS - 5
000995 0458 045A             DMIN:  .DW * + 2
000996 045A E9                      SEX R9
000997 045B 29                      DEC R9
000998 045C 09                      LDN R9
000999 045D FBFF                    XRI H'FF
001000 045F FC01                    ADI H'01
001001 0461 73                      STXD
001002 0462 09                      LDN R9
001003 0463 FBFF                    XRI H'FF
001004 0465 7C00                    ADCI H'00
001005 0467 59                      STR R9
001006 0468 19                      INC R9
001007 0469 19                      INC R9
001008 046A 3016                    BR MINOS
001009 046C                  ;
001010 046C 844F5645D2              .DB H'84,"OVE",H'D2 ; OVER
001011 0471 044F                    .DW DMIN - 9
001012 0473 0475             OVER:  .DW * + 2
001013 0475 29                      DEC R9
001014 0476 29                      DEC R9
001015 0477 49                      LDA R9
001016 0478 19                      INC R9
001017 0479 19                      INC R9
001018 047A 19                      INC R9
```

```
001019 047B 59                      STR R9
001020 047C 29                      DEC R9
001021 047D 29                      DEC R9
001022 047E 29                      DEC R9
001023 047F 49                      LDA R9
001024 0480 19                      INC R9
001025 0481 19                      INC R9
001026 0482 19                      INC R9
001027 0483 59                      STR R9
001028 0484 29                      DEC R9
001029 0485 DC                      SEP RC
001030 0486                  ;
001031 0486 8444524FD0              .DB H'84,"DRO",H'D0 ; DROP
001032 048B 046C                    .DW OVER - 7
001033 048D 048F            DROP:   .DW * + 2
001034 048F 29                      DEC R9
001035 0490 29                      DEC R9
001036 0491 DC                      SEP RC
001037 0492                  ;
001038 0492 84535741D0              .DB H'84,"SWA",H'D0 ; SWAP
001039 0497 0486                    .DW DROP - 7
001040 0499 049B            SWAP:   .DW * + 2
001041 049B 89                      GLO R9
001042 049C A8                      PLO R8
001043 049D 99                      GHI R9
001044 049E B8                      PHI R8
001045 049F 28                      DEC R8
001046 04A0 08                      LDN R8
001047 04A1 52                      STR R2
001048 04A2 19                      INC R9
001049 04A3 09                      LDN R9
001050 04A4 58                      STR R8
001051 04A5 02                      LDN R2
001052 04A6 59                      STR R9
001053 04A7 29                      DEC R9
001054 04A8 28                      DEC R8
001055 04A9 08                      LDN R8
001056 04AA 52                      STR R2
001057 04AB 09                      LDN R9
001058 04AC 58                      STR R8
001059 04AD 02                      LDN R2
001060 04AE 59                      STR R9
001061 04AF DC                      SEP RC
001062 04B0                  ;
001063 04B0 834455D0                .DB H'83,"DU",H'D0  ; DUP
001064 04B4 0492                    .DW SWAP - 7
001065 04B6 04B8            DUP:    .DW * + 2
001066 04B8 49                      LDA R9
001067 04B9 19                      INC R9
001068 04BA 59                      STR R9
001069 04BB 29                      DEC R9
001070 04BC 49                      LDA R9
001071 04BD 19                      INC R9
001072 04BE 59                      STR R9
001073 04BF 29                      DEC R9
001074 04C0 DC                      SEP RC
001075 04C1                  ;
```

```
001076 04C1 822BA1                  .DB H'82,H'2B,H'A1  ; +!
001077 04C4 04B0                    .DW DUP - 6
001078 04C6 04C8            PLUSS:  .DW * + 2
001079 04C8 49                      LDA R9
001080 04C9 B8                      PHI R8
001081 04CA 09                      LDN R9
001082 04CB A8                      PLO R8
001083 04CC 29                      DEC R9
001084 04CD 29                      DEC R9
001085 04CE 18                      INC R8
001086 04CF E8                      SEX R8
001087 04D0 09                      LDN R9
001088 04D1 F4                      ADD
001089 04D2 73                      STXD
001090 04D3 29                      DEC R9
001091 04D4 09                      LDN R9
001092 04D5 74                      ADC
001093 04D6 58                      STR R8
001094 04D7 29              POP:    DEC R9
001095 04D8 29                      DEC R9
001096 04D9 DC                      SEP RC
001097 04DA                  ;
001098 04DA 86544F47474CC5          .DB H'86,"TOGGL",H'C5 ; TOGGLE
001099 04E1 04C1                    .DW PLUSS - 5
001100 04E3 04E5            TGLE:   .DW * + 2
001101 04E5 19                      INC R9
001102 04E6 09                      LDN R9
001103 04E7 A7                      PLO R7
001104 04E8 29                      DEC R9
001105 04E9 29                      DEC R9
001106 04EA 29                      DEC R9
001107 04EB 49                      LDA R9
001108 04EC B8                      PHI R8
001109 04ED 09                      LDN R9
001110 04EE A8                      PLO R8
001111 04EF E8                      SEX R8
001112 04F0 87                      GLO R7
001113 04F1 F3                      XOR
001114 04F2 58                      STR R8
001115 04F3 29                      DEC R9
001116 04F4 30D7                    BR POP
001117 04F6                         ;
001118 04F6 81C0                    .DW H'81C0          ; @
001119 04F8 04DA                    .DW TGLE - 9
001120 04FA 04FC            AT:     .DW * + 2
001121 04FC 49                      LDA R9
001122 04FD B8                      PHI R8
001123 04FE 09                      LDN R9
001124 04FF A8                      PLO R8
001125 0500 29                      DEC R9
001126 0501 48                      LDA R8
001127 0502 59                      STR R9
001128 0503 19                      INC R9
001129 0504 08                      LDN R8
001130 0505 59                      STR R9
001131 0506 29                      DEC R9
001132 0507 DC                      SEP RC
```

```
001133 0508                  ;
001134 0508 8243C0                   .DB H'82,H'43,H'C0  ; C@
001135 050B 04F6                     .DW AT - 4
001136 050D 050F             CAT:    .DW * + 2
001137 050F 49                       LDA R9
001138 0510 B8                       PHI R8
001139 0511 09                       LDN R9
001140 0512 A8                       PLO R8
001141 0513 08                       LDN R8
001142 0514 59                       STR R9
001143 0515 29                       DEC R9
001144 0516 F800                     LDI H'00
001145 0518 59                       STR R9
001146 0519 DC                       SEP RC
001147 051A                  ;
001148 051A 81A1                     .DW H'81A1          ; !   STORE
001149 051C 0508                     .DW CAT - 5
001150 051E 0520             EX:     .DW * + 2
001151 0520 49                       LDA R9
001152 0521 B8                       PHI R8
001153 0522 09                       LDN R9
001154 0523 A8                       PLO R8
001155 0524 29                       DEC R9
001156 0525 29                       DEC R9
001157 0526 29                       DEC R9
001158 0527 49                       LDA R9
001159 0528 58                       STR R8
001160 0529 18                       INC R8
001161 052A 09                       LDN R9
001162 052B 58                       STR R8
001163 052C 29                       DEC R9
001164 052D 29                       DEC R9
001165 052E 29                       DEC R9
001166 052F DC                       SEP RC
001167 0530                  ;
001168 0530 8243A1                   .DB H'82,H'43,H'A1  ; C!   C STORE
001169 0533 051A                     .DW EX - 4
001170 0535 0537             CEX:    .DW * + 2
001171 0537 49                       LDA R9
001172 0538 B8                       PHI R8
001173 0539 09                       LDN R9
001174 053A A8                       PLO R8
001175 053B 29                       DEC R9
001176 053C 29                       DEC R9
001177 053D 09                       LDN R9
001178 053E 58                       STR R8
001179 053F 29                       DEC R9
001180 0540 29                       DEC R9
001181 0541 29                       DEC R9
001182 0542 DC                       SEP RC
001183 0543                  ;
001184 0543                  ;
001185 0543                  ; THE FOLLOWING FOUR ROUTINES ARE USER DEFINED
001186 0543                  ; FOR THE CONSULE INPUT AND OUTPUT
001187 0543                          ;
001188 0543                          ; SEND ASCII TO TERMINAL
001189 0543                          ;
```

```
001190 0543 84454D49D4              .DB H’84,"EMI",H’D4 ; EMIT
001191 0548 0530                    .DW CEX - 5
001192 054A 05C2            EMIT:   .DW NEST            ; OUTPUT CHAR TO
001193 054C                                             ; TERMINAL AND INCREMENT
001194 054C                                             ; "OUT"
001195 054C 06DB                    .DW FOUT            ; GET ADDRESS OF OUT
001196 054E 0552                    .DW CSEND
001197 0550 0379                    .DW SEMIS
001198 0552 0554            CSEND:  .DW * + 2
001199 0554                 ; INCREMENT USER VARIABLE "OUT"
001200 0554                 ; WHOSE ADDRESS IN ON THE COMPUTATION
001201 0554                 ; STACK
001202 0554 49                      LDA R9
001203 0555 B8                      PHI R8
001204 0556 09                      LDN R9
001205 0557 A8                      PLO R8
001206 0558 29                      DEC R9
001207 0559 29                      DEC R9
001208 055A 29                      DEC R9
001209 055B 18                      INC R8
001210 055C 08                      LDN R8
001211 055D FC01                    ADI H’01
001212 055F 58                      STR R8
001213 0560 28                      DEC R8
001214 0561 08                      LDN R8
001215 0562 7C00                    ADCI H’00
001216 0564 58                      STR R8
001217 0565                 ; WAIT UNTIL UART FREE AND
001218 0565                 ; SEND OUT CHARACTER WHICH IS ON
001219 0565                 ; THE COMP.STACK
001220 0565 E2              CSEND1: SEX R2
001221 0566 6B                      INP R3              ; GET UART STATUS
001222 0567 FE                      SHL                 ; GET THRE FLAG
001223 0568 3B65                    BNF CSEND1
001224 056A E9                      SEX R9
001225 056B 19                      INC R9
001226 056C 62                      OUT 2
001227 056D 29                      DEC R9
001228 056E 29                      DEC R9
001229 056F 29                      DEC R9
001230 0570 29                      DEC R9
001231 0571 DC                      SEP RC
001232 0572                  ;
001233 0572                         ;
001234 0572 834B45D9                .DB H’83,"KE",H’D9  ; KEY   READ KEYBOARD
001235 0576 0543                    .DW EMIT - 7
001236 0578 057A            KEY:    .DW * + 2
001237 057A E9                      SEX CSTACK
001238 057B 60                      IRX
001239 057C 60                      IRX
001240 057D 60                      IRX
001241 057E 6B              KEY1:   INP R3              ; GET DA BIT FROM
001242 057F F6                      SHR     ; UART STATUS BYTE
001243 0580 3B7E                    BNF KEY1            ; WAIT ’TILL DATA AVAIL.
001244 0582 6A                      INP 2               ; GET DATA
001245 0583 FA7F                    ANI H’7F            ; STRIP PARITY BIT
001246 0585 59                      STR CSTACK
```

```
001247 0586 29                      DEC CSTACK
001248 0587 F800                    LDI H'00
001249 0589 59                      STR CSTACK
001250 058A DC                      SEP RC
001251 058B                         ;
001252 058B                         ; TEST FOR BREAK
001253 058B                         ;
001254 058B 893F5445524D494E        .DB H'89,"?TERMINA",H'CC ; ?TERMINAL
       0593 41CC
001255 0595 0572             .DW KEY - 6
001256 0597                 ; ASSUME THAT A FRAMING ERROR INDICATES
001257 0597                 ; THAT THE BREAK KEY IS OR WAS PRESSED.
001258 0597 0599            QTERM:  .DW * +2
001259 0599 E9                      SEX CSTACK
001260 059A 60                      IRX
001261 059B 60                      IRX
001262 059C 60                      IRX
001263 059D 6B                      INP 3               ; GET NEW STATUS BYTE
001264 059E FA08                    ANI H'08            ; FET FRAMING ERROR BIT
001265 05A0 73                      STXD
001266 05A1 6A                      INP 2               ; DO A DUMMY READ TO
001267 05A2                 ; CLEAR THE DATA AVAILABLE SIGNAL
001268 05A2 F800                    LDI H'00
001269 05A4 59                      STR CSTACK
001270 05A5 DC                      SEP RC
001271 05A6                         ;
001272 05A6                         ; SEND CR/LF TO TERMINAL
001273 05A6                         ;
001274 05A6 8243D2                  .DB H'82,H'43,H'D2  ; CR
001275 05A9 058B                    .DW QTERM - 12
001276 05AB 05AD            CR:     .DW * + 2
001277 05AD E2                      SEX RSTACK
001278 05AE 6B              CR1:    INP 3               ; GET THRE BIT
001279 05AF FE                      SHL
001280 05B0 3BAE                    BNF CR1
001281 05B2 F80D                    LDI H'0D
001282 05B4 52                      STR RSTACK
001283 05B5 62                      OUT 2
001284 05B6 22                      DEC RSTACK
001285 05B7 6B              CR2:    INP 3
001286 05B8 FE                      SHL
001287 05B9 3BB7                    BNF CR2
001288 05BB F80A                    LDI H'0A
001289 05BD 52                      STR RSTACK
001290 05BE 62                      OUT 2
001291 05BF 22                      DEC RSTACK
001292 05C0 E9                      SEX R9
001293 05C1 DC                      SEP RC
&vellip;
001308 05CD 19              VAR:    INC R9
001309 05CE 19                      INC R9
001310 05CF 9B                      GHI RB
001311 05D0 59                      STR R9
001312 05D1 19                      INC R9
001313 05D2 8B                      GLO RB
001314 05D3 59                      STR R9
001315 05D4 29                      DEC R9
```

```
001316 05D5 DC                  SEP RC
001317 05D6                     ;
001318 05D6 19          CONST:  INC R9
001319 05D7 19                  INC R9
001320 05D8 4B                  LDA RB
001321 05D9 59                  STR R9
001322 05DA 19                  INC R9
001323 05DB 4B                  LDA RB
001324 05DC 59                  STR R9
001325 05DD 29                  DEC R9
001326 05DE DC                  SEP RC
001327 05DF                     ;
001328 05DF 19          USER:   INC R9
001329 05E0 19                  INC R9
001330 05E1 E9                  SEX R9
001331 05E2 4B                  LDA RB
001332 05E3 59                  STR R9
001333 05E4 19                  INC R9
001334 05E5 4B                  LDA RB
001335 05E6 59                  STR R9
001336 05E7 8D                  GLO RD
001337 05E8 F4                  ADD
001338 05E9 73                  STXD
001339 05EA 9D                  GHI RD
001340 05EB 74                  ADC
001341 05EC 59                  STR R9
001342 05ED DC                  SEP RC
001343 05EE             ;
001344 05EE             ; FROM HERE ON THE SOURCE IS GENERALLY
001345 05EE             ; DEFINED WITH FORTH WORD ADDRESSES
001346 05EE             ;
001347 05EE 81B0                .DW H'81B0          ; 0
001348 05F0 05A6                .DW CR - 5
001349 05F2 05D6        ZERO:   .DW CONST
001350 05F4 0000                .DW H'0000
001351 05F6                     ;
001352 05F6 81B1                .DW H'81B1          ; 1
001353 05F8 05EE                .DW ZERO - 4
001354 05FA 05D6        ONE:    .DW CONST
001355 05FC 0001                .DW H'0001
001356 05FE                     ;
001357 05FE 81B2                .DW H'81B2          ; 2
001358 0600 05F6                .DW ONE - 4
001359 0602 05D6        TWO:    .DW CONST
001360 0604 0002                .DW H'0002
001361 0606                     ;
001362 0606 8242CC              .DB H'82,H'42,H'CC  ; BL
001363 0609 05FE                .DW TWO - 4         ; CONSTANT ASCII BLANK
001364 060B 05D6        BL:     .DW CONST
001365 060D 0020                .DW H'0020
001366 060F                     ;
001367 060F 83432FCC            .DB H'83,H'43,H'2F,H'CC ; C/L
001368 0613                                         ; CHARACTERS PER LINE
001369 0613 0606                .DW BL - 5
001370 0615 05D6        CL:     .DW CONST
001371 0617 0040                .DW H'0040          ; 64 (DECIMAL)
001372 0619                     ;
```

```
001373 0619 8546495253D4            .DB H’85,"FIRS",H’D4 ; FIRST
001374 061F 060F                    .DW CL - 6
001375 0621 05D6            FIRST:  .DW CONST
001376 0623 4000                    .DW FIRSTB
001377 0625                         ;
001378 0625 854C494D49D4            .DB H’85,"LIMI",H’D4 ; LIMIT
001379 062B 0619                    .DW FIRST - 8
001380 062D 05D6            LIMIT:  .DW CONST
001381 062F 6C2C                    .DW LIMITB
001382 0631                         ;
001383 0631 85422F4255C6            .DB H’85,"B/BU",H’C6 ; B/BUF
001384 0637                                      ; BYTES PER BUFFER
001385 0637 0625                    .DW LIMIT - 8
001386 0639 05D6            BBUF:   .DW CONST
001387 063B 0400                    .DW H’0400          ; 1024 BYTES/BUFFER
001388 063D                         ;
001389 063D 85422F5343D2            .DB H’85,"B/SC",H’D2 ; B/SCR
001390 0643                                      ; BUFFERS/SCREEN
001391 0643 0631                    .DW BBUF - 8
001392 0645 05D6            BSCR:   .DW CONST
001393 0647 0001                    .DW H’0001
001394 0649                         ;
001395 0649 864F52494749CE          .DB H’86,"ORIGI",H’CE ; ORIGIN
001396 0650 063D                    .DW BSCR - 8
001397 0652 05D6            ORGN:   .DW CONST
001398 0654 005E                    .DW H’005E
001399 0656                         ;
001400 0656 872B4F52494749CE        .DB H’87,"+ORIGI",H’CE ; +ORIGIN
001401 065E
001402 065E 0649                    .DW ORGN - 9
001403 0660 05C2            PORGN:  .DW NEST
001404 0662 0652                    .DW ORGN
001405 0664 03F7                    .DW PLUS
001406 0666 0379                    .DW SEMIS
001407 0668                         ;
001408 0668                         ; USER VARIABLES
001409 0668                         ;
001410 0668 8253B0                  .DB H’82,H’53,H’B0  ; S0
001411 066B 0656                    .DW PORGN - 10
001412 066D 05DF            SO:     .DW USER
001413 066F 0006                    .DW H’0006
001414 0671                         ;
001415 0671 8252B0                  .DB H’82,H’52,H’B0  ; R0
001416 0674 0668                    .DW SO - 5
001417 0676 05DF            RO:     .DW USER
001418 0678 0008                    .DW H’0008
001419 067A                         ;
001420 067A 835449C2                .DB H’83,"TI",H’C2  ; TIB
001421 067E 0671                    .DW RO - 5
001422 0680 05DF            TIB:    .DW USER
001423 0682 000A                    .DW H’000A
001424 0684                         ;
001425 0684 8557494454C8            .DB H’85,"WIDT",H’C8 ; WIDTH
001426 068A 067A                    .DW TIB - 6
001427 068C 05DF            WIDTH:  .DW USER
001428 068E 000C                    .DW H’000C
001429 0690                         ;
```

```
001430 0690 875741524E494EC7         .DB H'87,"WARNIN",H'C7 ; WARNING
001431 0698 0684              .DW WIDTH - 8
001432 069A 05DF             WRNG:   .DW USER
001433 069C 000E                     .DW H'000E
001434 069E                          ;
001435 069E 8546454E43C5             .DB H'85,"FENC",H'C5 ; FENCE   FORGET BOUNDRY
001436 06A4 0690                     .DW WRNG - 10
001437 06A6 05DF             FNCE:   .DW USER
001438 06A8 0010                     .DW H'0010
001439 06AA                          ;
001440 06AA 8244D0                   .DB H'82,H'44,H'D0  ; DP
001441 06AD 069E                     .DW FNCE - 8
001442 06AF 05DF             DP:     .DW USER
001443 06B1 0012                     .DW H'0012
001444 06B3                          ;
001445 06B3 88564F432D4C494E         .DB H'88,"VOC-LIN",H'CB ; VOC-LINK
       06BB CB
001446 06BC 06AA              .DW DP - 5
001447 06BE 05DF             VL:     .DW USER
001448 06C0 0014                     .DW H'0014
001449 06C2                          ;
001450 06C2 83424CCB                 .DB H'83,H'42,H'4C,H'CB ; BLK
001451 06C6 06B3                     .DW VL - 11
001452 06C8 05DF             BLK:    .DW USER
001453 06CA 0016                     .DW H'0016
001454 06CC                          ;
001455 06CC 8249CE                   .DB H'82,H'49,H'CE  ; IN
001456 06CF 06C2                     .DW BLK - 6
001457 06D1 05DF             FIN:    .DW USER
001458 06D3 0018                     .DW H'0018
001459 06D5                          ;
001460 06D5 834F55D4                 .DB H'83,"OU",H'D4  ; OUT
001461 06D9 06CC                     .DW FIN - 5
001462 06DB 05DF             FOUT:   .DW USER
001463 06DD 001A                     .DW H'001A
001464 06DF                          ;
001465 06DF 835343D2                 .DB H'83,"SC",H'D2  ; SCR
001466 06E3 06D5                     .DW FOUT - 6
001467 06E5 05DF             FSCR:   .DW USER
001468 06E7 001C                     .DW H'001C
001469 06E9                          ;
001470 06E9 864F46465345D4           .DB H'86,"OFFSE",H'D4 ; OFFSET
001471 06F0 06DF                     .DW FSCR - 6
001472 06F2 05DF             OFST:   .DW USER
001473 06F4 001E                     .DW H'001E
001474 06F6                          ;
001475 06F6 87434F4E544558D4         .DB H'87,"CONTEX",H'D4 ; CONTEXT
001476 06FE 06E9              .DW OFST - 9
001477 0700 05DF             CNTX:   .DW USER
001478 0702 0020                     .DW H'0020
001479 0704                          ;
001480 0704 8743555252454ED4         .DB H'87,"CURREN",H'D4 ; CURRENT
001481 070C 06F6              .DW CNTX - 10
001482 070E 05DF             CRNT:   .DW USER
001483 0710 0022                     .DW H'0022
001484 0712                          ;
001485 0712 8553544154C5             .DB H'85,"STAT",H'C5 ; STATE
```

```
001486 0718 0704                    .DW CRNT - 10
001487 071A 05DF            STT:    .DW USER
001488 071C 0024                    .DW H'0024
001489 071E                         ;
001490 071E 84424153C5              .DB H'84,"BAS",H'C5 ; BASE
001491 0723 0712                    .DW STT - 8
001492 0725 05DF            BASE:   .DW USER
001493 0727 0026                    .DW H'0026
001494 0729                         ;
001495 0729 834450CC                .DB H'83,H'44,H'50,H'CC ; DPL
001496 072D 071E                    .DW BASE - 7
001497 072F 05DF            DPL:    .DW USER
001498 0731 0028                    .DW H'0028
001499 0733                         ;
001500 0733 83464CC4                .DB H'83,H'46,H'4C,H'C4 ; FLD
001501 0737 0729                    .DW DPL - 6
001502 0739 05DF            FLD:    .DW USER
001503 073B 002A                    .DW H'002A
001504 073D                         ;
001505 073D 834353D0                .DB H'83,H'43,H'53,H'D0 ; CSP
001506 0741 0733                    .DW FLD - 6
001507 0743 05DF            CSP:    .DW USER
001508 0745 002C                    .DW H'002C
001509 0747                         ;
001510 0747 8252A3                  .DB H'82,H'52,H'A3  ; R#
001511 074A 073D                    .DW CSP - 6
001512 074C 05DF            RNU:    .DW USER
001513 074E 002E                    .DW H'002E
001514 0750                         ;
001515 0750 83484CC4                .DB H'83,H'48,H'4C,H'C4 ; HLD
001516 0754 0747                    .DW RNU - 5
001517 0756 05DF            HLD:    .DW USER
001518 0758 0030                    .DW H'0030
001519 075A 8244D6                  .DB H'82,H'44,H'D6  ; DV
001520 075D 0750                    .DW HLD - 6         ; 3 BYTE VECTOR AREA
001521 075F                 ; USED DURING DISK OPERATIONS
001522 075F 05DF            DV:     .DW USER
001523 0761 0032                    .DW H'0032
001524 0763                         ;
001525 0763                         ; END OF USER VARIABLES
001526 0763                         ;
001527 0763 8231AB                  .DB H'82,H'31,H'AB  ; 1+
001528 0766 075A                    .DW DV - 5
001529 0768 05C2            PLUS1:  .DW NEST
001530 076A 05FA                    .DW ONE
001531 076C 03F7                    .DW PLUS
001532 076E 0379                    .DW SEMIS
001533 0770                         ;
001534 0770 8232AB                  .DB H'82,H'32,H'AB  ; 2+
001535 0773 0763                    .DW PLUS1 - 5
001536 0775 05C2            PLUS2:  .DW NEST
001537 0777 0602                    .DW TWO
001538 0779 03F7                    .DW PLUS
001539 077B 0379                    .DW SEMIS
001540 077D                         ;
001541 077D 84484552C5              .DB H'84,"HER",H'C5 ; HERE
001542 0782 0770                    .DW PLUS2 - 5
```

```
001543 0784 05C2            HERE:   .DW NEST
001544 0786 06AF                    .DW DP
001545 0788 04FA                    .DW AT
001546 078A 0379                    .DW SEMIS
001547 078C                         ;
001548 078C 85414C4C4FD4            .DB H'85,"ALLO",H'D4 ; ALLOT
001549 0792 077D                    .DW HERE - 7
001550 0794 05C2            ALLOT:  .DW NEST
001551 0796 06AF                    .DW DP
001552 0798 04C6                    .DW PLUSS
001553 079A 0379                    .DW SEMIS
001554 079C                         ;
001555 079C 81AC                    .DW H'81AC          ; , (COMMA)
001556 079E 078C                    .DW ALLOT - 8
001557 07A0 05C2            COMMA:  .DW NEST
001558 07A2 0784                    .DW HERE
001559 07A4 051E                    .DW EX
001560 07A6 0602                    .DW TWO
001561 07A8 0794                    .DW ALLOT
001562 07AA 0379                    .DW SEMIS
001563 07AC                         ;
001564 07AC 8243AC                  .DB H'82,H'43,H'AC  ; C,
001565 07AF 079C                    .DW COMMA - 4
001566 07B1 05C2            CCMA:   .DW NEST
001567 07B3 0784                    .DW HERE
001568 07B5 0535                    .DW CEX
001569 07B7 05FA                    .DW ONE
001570 07B9 0794                    .DW ALLOT
001571 07BB 0379                    .DW SEMIS
001572 07BD                         ;
001573 07BD 81AD                    .DW H'81AD          ; - (MINUS SIGN)
001574 07BF 07AC                    .DW CCMA - 5
001575 07C1 05C2            MINS:   .DW NEST
001576 07C3 0412                    .DW MINUS
001577 07C5 03F7                    .DW PLUS
001578 07C7 0379                    .DW SEMIS
001579 07C9                         ;
001580 07C9 81BD                    .DW H'81BD          ; = (EQUAL SIGN)
001581 07CB 07BD                    .DW MINS - 4
001582 07CD 05C2            EQL:    .DW NEST
001583 07CF 07C1                    .DW MINS
001584 07D1 03D2                    .DW ZEQAL
001585 07D3 0379                    .DW SEMIS
001586 07D5                         ;
001587 07D5 81BC                    .DW H'81BC          ; < (LESS THAN SIGN)
001588 07D7 07C9                    .DW EQL - 4
001589 07D9 05C2            LESS:   .DW NEST
001590 07DB 07C1                    .DW MINS
001591 07DD 03EB                    .DW ZLESS
001592 07DF 0379                    .DW SEMIS
001593 07E1                         ;
001594 07E1 81BE                    .DW H'81BE          ; > (GTR THAN SIGN)
001595 07E3 07D5                    .DW LESS - 4
001596 07E5 05C2            GTR:    .DW NEST
001597 07E7 0499                    .DW SWAP
001598 07E9 07D9                    .DW LESS
001599 07EB 0379                    .DW SEMIS
```

```
001600 07ED                          ;
001601 07ED 83524FD4                 .DB H'83,H'52,H'4F,H'D4 ; ROT
001602 07F1 07E1                     .DW GTR - 4
001603 07F3 05C2             ROT:    .DW NEST
001604 07F5 039D                     .DW GR
001605 07F7 0499                     .DW SWAP
001606 07F9 03AD                     .DW RG
001607 07FB 0499                     .DW SWAP
001608 07FD 0379                     .DW SEMIS
001609 07FF                          ;
001610 07FF 8553504143C5             .DB H'85,"SPAC",H'C5 ; SPACE
001611 0805 07ED                     .DW ROT - 6
001612 0807 05C2             SPC:    .DW NEST
001613 0809 060B                     .DW BL
001614 080B 054A                     .DW EMIT
001615 080D 0379                     .DW SEMIS
001616 080F                          ;
001617 080F 842D4455D0               .DB H'84,"-DU",H'D0 ; -DUP
001618 0814 07FF                     .DW SPC - 8
001619 0816 05C2             MDUP:   .DW NEST
001620 0818 04B6                     .DW DUP
001621 081A 00D0                     .DW ZBRCH
001622 081C 0820                     .DW * + 4
001623 081E 04B6                     .DW DUP
001624 0820 0379                     .DW SEMIS
001625 0822                          ;
001626 0822 8854524156455253         .DB H'88,"TRAVERS",H'C5 ; TRAVERSE
       082A C5
001627 082B 080F              .DW MDUP - 7
001628 082D 05C2             TRVS:   .DW NEST
001629 082F 0499                     .DW SWAP
001630 0831 0473             TR1:    .DW OVER
001631 0833 03F7                     .DW PLUS
001632 0835 0086                     .DW LIT
001633 0837 007F                     .DW H'007F
001634 0839 0473                     .DW OVER
001635 083B 050D                     .DW CAT
001636 083D 07D9                     .DW LESS
001637 083F 00D0                     .DW ZBRCH
001638 0841 0831                     .DW TR1
001639 0843 0499                     .DW SWAP
001640 0845 048D                     .DW DROP
001641 0847 0379                     .DW SEMIS
001642 0849                          ;
001643 0849                          ;
001644 0849 864C41544553D4           .DB H'86,"LATES",H'D4 ; LATEST
001645 0850 0822                     .DW TRVS - 11
001646 0852 05C2             LTST:   .DW NEST
001647 0854 070E                     .DW CRNT
001648 0856 04FA                     .DW AT
001649 0858 04FA                     .DW AT
001650 085A 0379                     .DW SEMIS
001651 085C                          ;
001652 085C                          ;
001653 085C 834C46C1                 .DB H'83,H'4C,H'46,H'C1 ; LFA
001654 0860 0849                     .DW LTST - 9        ; LINK FIELD ADDRESS
001655 0862 05C2             LFA:    .DW NEST
```

```
001656 0864 0086                    .DW LIT
001657 0866 0004                    .DW H’0004
001658 0868 07C1                    .DW MINS
001659 086A 0379                    .DW SEMIS
001660 086C                         ;
001661 086C 834346C1                .DB H’83,H’43,H’46,H’C1 ; CFA
001662 0870 085C                    .DW LFA - 6         ; CODE FIELD ADDRESS
001663 0872 05C2            CFA:    .DW NEST
001664 0874 0602                    .DW TWO
001665 0876 07C1                    .DW MINS
001666 0878 0379                    .DW SEMIS
001667 087A                         ;
001668 087A                         ;
001669 087A 834E46C1                .DB H’83,H’4E,H’46,H’C1 ; NFA
001670 087E 086C                    .DW CFA - 6         ; NAME FIELD ADDRESS
001671 0880 05C2            NFA:    .DW NEST
001672 0882 0086                    .DW LIT
001673 0884 0005                    .DW H’0005
001674 0886 07C1                    .DW MINS
001675 0888 0086                    .DW LIT
001676 088A FFFF                    .DW H’FFFF
001677 088C 082D                    .DW TRVS
001678 088E 0379                    .DW SEMIS
001679 0890                         ;
001680 0890                         ;
001681 0890 835046C1                .DB H’83,H’50,H’46,H’C1 ; PFA
001682 0894 087A                    .DW NFA - 6         ; PARAMETER FIELD ADDRESS
001683 0896 05C2            PFA:    .DW NEST
001684 0898 05FA                    .DW ONE
001685 089A 082D                    .DW TRVS
001686 089C 0086                    .DW LIT
001687 089E 0005                    .DW H’0005
001688 08A0 03F7                    .DW PLUS
001689 08A2 0379                    .DW SEMIS
&vellip;
001701 08B5 863F4552524FD2          .DB H’86,"?ERRO",H’D2 ; ?ERROR
001702 08BC 08A4                    .DW DCSP - 7
001703 08BE 05C2            QERR:   .DW NEST
001704 08C0 0499                    .DW SWAP
001705 08C2 00D0                    .DW ZBRCH
001706 08C4 08CC                    .DW * + 8
001707 08C6 0CE5                    .DW ERROR
001708 08C8 00BD                    .DW BRCH
001709 08CA 08CE                    .DW * + 4
001710 08CC 048D                    .DW DROP
001711 08CE 0379                    .DW SEMIS
001712 08D0                         ;
001713 08D0                         ;
001714 08D0 853F434F4DD0            .DB H’85,"?COM",H’D0 ; ?COMP
001715 08D6 08B5                    .DW QERR - 9
001716 08D8 05C2            QCMP:   .DW NEST
001717 08DA 071A                    .DW STT
001718 08DC 04FA                    .DW AT
001719 08DE 03D2                    .DW ZEQAL
001720 08E0 0086                    .DW LIT
001721 08E2 0011                    .DW H’0011
001722 08E4 08BE                    .DW QERR
```

```
001723 08E6 0379                    .DW SEMIS
001724 08E8                         ;
001725 08E8                         ;
001726 08E8 853F455845C3            .DB H'85,"?EXE",H'C3 ; ?EXEC
001727 08EE 08D0                    .DW QCMP - 8
001728 08F0 05C2            EXC:    .DW NEST
001729 08F2 071A                    .DW STT
001730 08F4 04FA                    .DW AT
001731 08F6 0086                    .DW LIT
001732 08F8 0012                    .DW H'0012
001733 08FA 08BE                    .DW QERR
001734 08FC 0379                    .DW SEMIS
001735 08FE                         ;
001736 08FE                         ;
001737 08FE 863F50414952D3          .DB H'86,"?PAIR",H'D3 ; ?PAIRS
001738 0905 08E8                    .DW EXC - 8
001739 0907 05C2            QPR:    .DW NEST
001740 0909 07C1                    .DW MINS
001741 090B 0086                    .DW LIT
001742 090D 0013                    .DW H'0013
001743 090F 08BE                    .DW QERR
001744 0911 0379                    .DW SEMIS
001745 0913                         ;
001746 0913                         ;
001747 0913 843F4353D0              .DB H'84,H'3F,H'43,H'53,H'D0 ; ?CSP
001748 0918 08FE                    .DW QPR - 9
001749 091A 05C2            QCSP:   .DW NEST
001750 091C 033D                    .DW FSPAT
001751 091E 0743                    .DW CSP
001752 0920 04FA                    .DW AT
001753 0922 07C1                    .DW MINS
001754 0924 0086                    .DW LIT
001755 0926 0014                    .DW H'0014
001756 0928 08BE                    .DW QERR
001757 092A 0379                    .DW SEMIS
001758 092C                         ;
001759 092C                         ;
001760 092C 883F4C4F4144494E        .DB H'88,"?LOADIN",H'C7 ; ?LOADING
       0934 C7
001761 0935
001762 0935 0913                    .DW QCSP - 7
001763 0937 05C2            QLDG:   .DW NEST
001764 0939 06C8                    .DW BLK
001765 093B 04FA                    .DW AT
001766 093D 03D2                    .DW ZEQAL
001767 093F 0086                    .DW LIT
001768 0941 0016                    .DW H'0016
001769 0943 08BE                    .DW QERR
001770 0945 0379                    .DW SEMIS
001771 0947                         ;
001772 0947                         ;
001773 0947 87434F4D50494CC5        .DB H'87,"COMPIL",H'C5 ; COMPILE
001774 094F 092C             .DW QLDG - 11
001775 0951 05C2            CMPL:   .DW NEST
001776 0953 08D8                    .DW QCMP
001777 0955 03AD                    .DW RG
001778 0957 04B6                    .DW DUP
```

```
001779 0959 0775                    .DW PLUS2
001780 095B 039D                    .DW GR
001781 095D 04FA                    .DW AT
001782 095F 07A0                    .DW COMMA
001783 0961 0379                    .DW SEMIS
001784 0963                         ;
001785 0963                  ;
001786 0963 C1DB                    .DW H'C1DB          ; [   LEFT BRACKET
001787 0965 0947                    .DW CMPL - 10
001788 0967 05C2             LB:    .DW NEST
001789 0969 05F2                    .DW ZERO
001790 096B 071A                    .DW STT
001791 096D 051E                    .DW EX
001792 096F 0379                    .DW SEMIS
001793 0971                         ;
001794 0971                         ;
001795 0971 81DD                    .DW H'81DD          ; ]   RIGHT BRACKET
001796 0973 0963                    .DW LB - 4
001797 0975 05C2             RBK:   .DW NEST
001798 0977 0086                    .DW LIT
001799 0979 00C0                    .DW H'00C0
001800 097B 071A                    .DW STT
001801 097D 051E                    .DW EX
001802 097F 0379                    .DW SEMIS
001803 0981                         ;
001804 0981                         ;
001805 0981 86534D554447C5          .DB H'86,"SMUDG",H'C5 ; SMUDGE
001806 0988 0971                    .DW RBK - 4
001807 098A 05C2             SMDG:  .DW NEST
001808 098C 0852                    .DW LTST
001809 098E 0086                    .DW LIT
001810 0990 0020                    .DW H'0020
001811 0992 04E3                    .DW TGLE
001812 0994 0379                    .DW SEMIS
001813 0996                         ;
001814 0996                         ;
001815 0996 834845D8                .DB H'83,H'48,H'45,H'D8 ; HEX
001816 099A 0981                    .DW SMDG - 9
001817 099C 05C2             MHEX:  .DW NEST
001818 099E 0086                    .DW LIT
001819 09A0 0010                    .DW H'0010
001820 09A2 0725                    .DW BASE
001821 09A4 051E                    .DW EX
001822 09A6 0379                    .DW SEMIS
001823 09A8                         ;
001824 09A8                         ;
001825 09A8 87444543494D41CC        .DB H'87,"DECIMA",H'CC ; DECIMAL
001826 09B0 0996                    .DW MHEX - 6
001827 09B2 05C2             MDCML: .DW NEST
001828 09B4 0086                    .DW LIT
001829 09B6 000A                    .DW H'000A
001830 09B8 0725                    .DW BASE
001831 09BA 051E                    .DW EX
001832 09BC 0379                    .DW SEMIS
001833 09BE                         ;
001834 09BE                         ;
001835 09BE 87283B434F4445A9        .DB H'87,"(;CODE",H'A9 ; (;CODE)
```

```
001836 09C6 09A8                    .DW MDCML - 10
001837 09C8 05C2          PCODE:    .DW NEST
001838 09CA 03AD                    .DW RG
001839 09CC 0852                    .DW LTST
001840 09CE 0896                    .DW PFA
001841 09D0 0872                    .DW CFA
001842 09D2 051E                    .DW EX
001843 09D4 0379                    .DW SEMIS
001844 09D6                         ;
001845 09D6                         ;
001846 09D6 C53B434F44C5            .DB H'C5,";COD",H'C5 ; ;CODE (IMMEDIATE)
001847 09DC 09BE                    .DW PCODE - 10
001848 09DE 05C2          CODE:     .DW NEST
001849 09E0 091A                    .DW QCSP
001850 09E2 0951                    .DW CMPL
001851 09E4 09C8                    .DW PCODE
001852 09E6 0967                    .DW LB
001853 09E8 098A                    .DW SMDG
001854 09EA 0379                    .DW SEMIS
001855 09EC                         ;
001856 09EC                         ;
001857 09EC 85434F554ED4            .DB H'85,"COUN",H'D4 ; COUNT
001858 09F2 09D6                    .DW CODE - 8
001859 09F4 05C2          CNT:      .DW NEST
001860 09F6 04B6                    .DW DUP
001861 09F8 0768                    .DW PLUS1
001862 09FA 0499                    .DW SWAP
001863 09FC 050D                    .DW CAT
001864 09FE 0379                    .DW SEMIS
001865 0A00                         ;
001866 0A00                         ;
001867 0A00 84545950C5              .DB H'84,"TYP",H'C5 ; TYPE
001868 0A05 09EC                    .DW CNT - 8
001869 0A07 05C2          TYPE:     .DW NEST
001870 0A09 0816                    .DW MDUP
001871 0A0B 00D0                    .DW ZBRCH
001872 0A0D 0A25                    .DW * + 24
001873 0A0F 0473                    .DW OVER
001874 0A11 03F7                    .DW PLUS
001875 0A13 0499                    .DW SWAP
001876 0A15 0150                    .DW PDO
001877 0A17 03BD          TYP1:     .DW R
001878 0A19 050D                    .DW CAT
001879 0A1B 054A                    .DW EMIT
001880 0A1D 00EC                    .DW LUPE
001881 0A1F 0A17                    .DW TYP1
001882 0A21 00BD                    .DW BRCH
001883 0A23 0A27                    .DW * + 4
001884 0A25 048D                    .DW DROP
001885 0A27 0379                    .DW SEMIS
001886 0A29                         ;
001887 0A29                         ;
001888 0A29 892D545241494C49        .DB H'89,"-TRAILIN",H'C7 ; -TRAILING
       0A31 4EC7
001889 0A33 0A00                    .DW TYPE - 7
001890 0A35 05C2          TRLG:     .DW NEST
001891 0A37 04B6                    .DW DUP
```

```
001892 0A39 05F2                    .DW ZERO
001893 0A3B 0150                    .DW PDO
001894 0A3D 0473            TRL1:   .DW OVER
001895 0A3F 0473                    .DW OVER
001896 0A41 03F7                    .DW PLUS
001897 0A43 05FA                    .DW ONE
001898 0A45 07C1                    .DW MINS
001899 0A47 050D                    .DW CAT
001900 0A49 060B                    .DW BL
001901 0A4B 07C1                    .DW MINS
001902 0A4D 00D0                    .DW ZBRCH
001903 0A4F 0A57                    .DW * + 8
001904 0A51 0389                    .DW LVE
001905 0A53 00BD                    .DW BRCH
001906 0A55 0A5B                    .DW * + 6
001907 0A57 05FA                    .DW ONE
001908 0A59 07C1                    .DW MINS
001909 0A5B 00EC                    .DW LUPE
001910 0A5D 0A3D                    .DW TRL1
001911 0A5F 0379                    .DW SEMIS
001912 0A61                         ;
001913 0A61                         ;
001914 0A61 84282E22A9              .DB H'84,H'28,H'2E,H'22,H'A9 ; (.")
001915 0A66 0A29                    .DW TRLG - 12
001916 0A68 05C2            PDQ:    .DW NEST
001917 0A6A 03BD                    .DW R
001918 0A6C 09F4                    .DW CNT
001919 0A6E 04B6                    .DW DUP
001920 0A70 0768                    .DW PLUS1
001921 0A72 03AD                    .DW RG
001922 0A74 03F7                    .DW PLUS
001923 0A76 039D                    .DW GR
001924 0A78 0A07                    .DW TYPE
001925 0A7A 0379                    .DW SEMIS
001926 0A7C                         ;
001927 0A7C                         ;
001928 0A7C 864558504543D4          .DB H'86,"EXPEC",H'D4 ; EXPECT
001929 0A83 0A61                    .DW PDQ - 7
001930 0A85 05C2            EXPT:   .DW NEST
001931 0A87 0473                    .DW OVER
001932 0A89 03F7                    .DW PLUS
001933 0A8B 0473                    .DW OVER
001934 0A8D 0150                    .DW PDO
001935 0A8F 0578            EXPT4:  .DW KEY
001936 0A91 04B6                    .DW DUP
001937 0A93 0086                    .DW LIT
001938 0A95 000E                    .DW H'000E
001939 0A97 0660                    .DW PORGN
001940 0A99 04FA                    .DW AT
001941 0A9B 07CD                    .DW EQL
001942 0A9D 00D0                    .DW ZBRCH
001943 0A9F 0ABF                    .DW EXPT1
001944 0AA1 048D                    .DW DROP
001945 0AA3 0086                    .DW LIT
001946 0AA5 0008                    .DW H'0008
001947 0AA7 0473                    .DW OVER
001948 0AA9 03BD                    .DW R
```

```
001949 0AAB 07CD                    .DW EQL
001950 0AAD 04B6                    .DW DUP
001951 0AAF 03AD                    .DW RG
001952 0AB1 0602                    .DW TWO
001953 0AB3 07C1                    .DW MINS
001954 0AB5 03F7                    .DW PLUS
001955 0AB7 039D                    .DW GR
001956 0AB9 07C1                    .DW MINS
001957 0ABB 00BD                    .DW BRCH
001958 0ABD 0AE5                    .DW EXPT2
001959 0ABF 04B6            EXPT1:  .DW DUP
001960 0AC1 0086                    .DW LIT
001961 0AC3 000D                    .DW H'000D
001962 0AC5 07CD                    .DW EQL
001963 0AC7 00D0                    .DW ZBRCH
001964 0AC9 0AD7                    .DW EXPT3
001965 0ACB 0389                    .DW LVE
001966 0ACD 048D                    .DW DROP
001967 0ACF 060B                    .DW BL
001968 0AD1 05F2                    .DW ZERO
001969 0AD3 00BD                    .DW BRCH
001970 0AD5 0AD9                    .DW EXPT5
001971 0AD7 04B6            EXPT3:  .DW DUP
001972 0AD9 03BD            EXPT5:  .DW R
001973 0ADB 0535                    .DW CEX
001974 0ADD 05F2                    .DW ZERO
001975 0ADF 03BD                    .DW R
001976 0AE1 0768                    .DW PLUS1
001977 0AE3 051E                    .DW EX
001978 0AE5 054A            EXPT2:  .DW EMIT
001979 0AE7 00EC                    .DW LUPE
001980 0AE9 0A8F                    .DW EXPT4
001981 0AEB 048D                    .DW DROP
001982 0AED 0379                    .DW SEMIS
001983 0AEF                         ;
001984 0AEF                         ;
001985 0AEF 8551554552D9            .DB H'85,"QUER",H'D9 ; QUERY
001986 0AF5 0A7C                    .DW EXPT - 9         ; INPUT LINE OF TEXT
001987 0AF7 05C2            QUER:   .DW NEST
001988 0AF9 0680                    .DW TIB
001989 0AFB 04FA                    .DW AT
001990 0AFD 0086                    .DW LIT
001991 0AFF 0050                    .DW H'0050
001992 0B01 0A85                    .DW EXPT
001993 0B03 05F2                    .DW ZERO
001994 0B05 06D1                    .DW FIN
001995 0B07 051E                    .DW EX
001996 0B09 0379                    .DW SEMIS
001997 0B0B                         ;
001998 0B0B                         ;
001999 0B0B C180                    .DW H'C180           ; X   (IMMEDIATE)
002000 0B0D 0AEF                    .DW QUER - 8
002001 0B0F 05C2            X:      .DW NEST
002002 0B11 06C8                    .DW BLK
002003 0B13 04FA                    .DW AT
002004 0B15 00D0                    .DW ZBRCH
002005 0B17 0B3F                    .DW X2
```

```
002006 0B19 05FA                    .DW ONE
002007 0B1B 06C8                    .DW BLK
002008 0B1D 04C6                    .DW PLUSS
002009 0B1F 05F2                    .DW ZERO
002010 0B21 06D1                    .DW FIN
002011 0B23 051E                    .DW EX
002012 0B25 06C8                    .DW BLK
002013 0B27 04FA                    .DW AT
002014 0B29 0086                    .DW LIT
002015 0B2B 0007                    .DW H'0007
002016 0B2D 02F3                    .DW FAND
002017 0B2F 03D2                    .DW ZEQAL
002018 0B31 00D0                    .DW ZBRCH
002019 0B33 0B3B                    .DW X1
002020 0B35 08F0                    .DW EXC
002021 0B37 03AD                    .DW RG
002022 0B39 048D                    .DW DROP
002023 0B3B 00BD            X1:     .DW BRCH
002024 0B3D 0B43                    .DW XEND
002025 0B3F 03AD            X2:     .DW RG
002026 0B41 048D                    .DW DROP
002027 0B43 0379            XEND:   .DW SEMIS
002028 0B45                         ;
002029 0B45                         ;
002030 0B45 8446494CCC              .DB H'84,"FIL",H'CC ; FILL   FILL MEMORY
002031 0B4A 0B0B                    .DW X - 4
002032 0B4C 05C2            FILL:   .DW NEST
002033 0B4E 0499                    .DW SWAP
002034 0B50 039D                    .DW GR
002035 0B52 0473                    .DW OVER
002036 0B54 0535                    .DW CEX
002037 0B56 04B6                    .DW DUP
002038 0B58 0768                    .DW PLUS1
002039 0B5A 03AD                    .DW RG
002040 0B5C 05FA                    .DW ONE
002041 0B5E 07C1                    .DW MINS
002042 0B60 0246                    .DW CMOVE
002043 0B62 0379                    .DW SEMIS
002044 0B64                         ;
002045 0B64                         ;
002046 0B64 8545524153C5            .DB H'85,"ERAS",H'C5 ; ERASE   ZERO MEMORY
002047 0B6A 0B45                    .DW FILL - 7
002048 0B6C 05C2            ERS:    .DW NEST
002049 0B6E 05F2                    .DW ZERO
002050 0B70 0B4C                    .DW FILL
002051 0B72 0379                    .DW SEMIS
002052 0B74                         ;
002053 0B74                         ;
002054 0B74 86424C414E4BD3          .DB H'86,"BLANK",H'D3 ; BLANKS
002055 0B7B 0B64                    .DW ERS - 8           ; FILL MEMORY WITH
002056 0B7D 05C2            BLNK:   .DW NEST              ; ASCII BLANKS
002057 0B7F 060B                    .DW BL
002058 0B81 0B4C                    .DW FILL
002059 0B83 0379                    .DW SEMIS
002060 0B85                         ;
002061 0B85                         ;
002062 0B85 84484F4CC4              .DB H'84,"HOL",H'C4 ; HOLD
```

```
002063 0B8A 0B74                    .DW BLNK - 9
002064 0B8C 05C2            HOLD:   .DW NEST
002065 0B8E 0086                    .DW LIT
002066 0B90 FFFF                    .DW H'FFFF          ; -1
002067 0B92 0756                    .DW HLD
002068 0B94 04C6                    .DW PLUSS
002069 0B96 0756                    .DW HLD
002070 0B98 04FA                    .DW AT
002071 0B9A 0535                    .DW CEX
002072 0B9C 0379                    .DW SEMIS
002073 0B9E                         ;
002074 0B9E                         ;
002075 0B9E 835041C4                .DB H'83,"PA",H'C4  ; PAD
002076 0BA2 0B85                    .DW HOLD - 7
002077 0BA4 05C2            PAD:    .DW NEST
002078 0BA6 0784                    .DW HERE
002079 0BA8 0086                    .DW LIT
002080 0BAA 0044                    .DW H'0044
002081 0BAC 03F7                    .DW PLUS
002082 0BAE 0379                    .DW SEMIS
002083 0BB0                         ;
002084 0BB0                         ;
002085 0BB0 84574F52C4              .DB H'84,"WOR",H'C4 ; WORD
002086 0BB5 0B9E                    .DW PAD - 6
002087 0BB7 05C2            WORD:   .DW NEST
002088 0BB9 06C8                    .DW BLK
002089 0BBB 04FA                    .DW AT
002090 0BBD 00D0                    .DW ZBRCH
002091 0BBF 0BCB                    .DW WD1
002092 0BC1 06C8                    .DW BLK
002093 0BC3 04FA                    .DW AT
002094 0BC5 160A                    .DW BLOCK
002095 0BC7 00BD                    .DW BRCH
002096 0BC9 0BCF                    .DW WD2
002097 0BCB 0680            WD1:    .DW TIB
002098 0BCD 04FA                    .DW AT
002099 0BCF 06D1            WD2:    .DW FIN
002100 0BD1 04FA                    .DW AT
002101 0BD3 03F7                    .DW PLUS
002102 0BD5 0499                    .DW SWAP
002103 0BD7 01FE                    .DW ENCL
002104 0BD9 0784                    .DW HERE
002105 0BDB 0086                    .DW LIT
002106 0BDD 0022                    .DW H'0022
002107 0BDF 0B7D                    .DW BLNK
002108 0BE1 06D1                    .DW FIN
002109 0BE3 04C6                    .DW PLUSS
002110 0BE5 0473                    .DW OVER
002111 0BE7 07C1                    .DW MINS
002112 0BE9 039D                    .DW GR
002113 0BEB 03BD                    .DW R
002114 0BED 0784                    .DW HERE
002115 0BEF 0535                    .DW CEX
002116 0BF1 03F7                    .DW PLUS
002117 0BF3 0784                    .DW HERE
002118 0BF5 0768                    .DW PLUS1
002119 0BF7 03AD                    .DW RG
```

```
002120 0BF9 0246                    .DW CMOVE
002121 0BFB 0379                    .DW SEMIS
002122 0BFD                         ;
002123 0BFD                         ;
002124 0BFD 88284E554D424552        .DB H'88,"(NUMBER",H'A9 ; (NUMBER)
       0C05 A9
002125 0C06 0BB0                    .DW WORD - 7
002126 0C08 05C2            PNMBR:  .DW NEST
002127 0C0A 0768                    .DW PLUS1
002128 0C0C 04B6                    .DW DUP
002129 0C0E 039D                    .DW GR
002130 0C10 050D                    .DW CAT
002131 0C12 0725                    .DW BASE
002132 0C14 04FA                    .DW AT
002133 0C16 016B                    .DW DGT
002134 0C18 00D0                    .DW ZBRCH
002135 0C1A 0C46                    .DW PNM2
002136 0C1C 0499                    .DW SWAP
002137 0C1E 0725                    .DW BASE
002138 0C20 04FA                    .DW AT
002139 0C22 0278                    .DW USTAR
002140 0C24 048D                    .DW DROP
002141 0C26 07F3                    .DW ROT
002142 0C28 0725                    .DW BASE
002143 0C2A 04FA                    .DW AT
002144 0C2C 0278                    .DW USTAR
002145 0C2E 042A                    .DW DPLUS
002146 0C30 072F                    .DW DPL
002147 0C32 04FA                    .DW AT
002148 0C34 0768                    .DW PLUS1
002149 0C36 00D0                    .DW ZBRCH
002150 0C38 0C40                    .DW PNM1
002151 0C3A 05FA                    .DW ONE
002152 0C3C 072F                    .DW DPL
002153 0C3E 04C6                    .DW PLUSS
002154 0C40 03AD            PNM1:   .DW RG
002155 0C42 00BD                    .DW BRCH
002156 0C44 0C0A                    .DW PNMBR + 2
002157 0C46 03AD            PNM2:   .DW RG
002158 0C48 0379                    .DW SEMIS
002159 0C4A                         ;
002160 0C4A                         ;
002161 0C4A                         ;
002162 0C4A 864E554D4245D2          .DB H'86,"NUMBE",H'D2 ; NUMBER
002163 0C51 0BFD                    .DW PNMBR - 11
002164 0C53 05C2            NMBR:   .DW NEST
002165 0C55 05F2                    .DW ZERO
002166 0C57 05F2                    .DW ZERO
002167 0C59 07F3                    .DW ROT
002168 0C5B 04B6                    .DW DUP
002169 0C5D 0768                    .DW PLUS1
002170 0C5F 050D                    .DW CAT
002171 0C61 0086                    .DW LIT
002172 0C63 002D                    .DW H'002D
002173 0C65 07CD                    .DW EQL
002174 0C67 04B6                    .DW DUP
002175 0C69 039D                    .DW GR
```

```
002176 0C6B 03F7                    .DW PLUS
002177 0C6D 0086                    .DW LIT
002178 0C6F FFFF                    .DW H'FFFF          ; -1
002179 0C71 072F            NMB1:   .DW DPL
002180 0C73 051E                    .DW EX
002181 0C75 0C08                    .DW PNMBR
002182 0C77 04B6                    .DW DUP
002183 0C79 050D                    .DW CAT
002184 0C7B 060B                    .DW BL
002185 0C7D 07C1                    .DW MINS
002186 0C7F 00D0                    .DW ZBRCH
002187 0C81 0C97                    .DW NMB2
002188 0C83 04B6                    .DW DUP
002189 0C85 050D                    .DW CAT
002190 0C87 0086                    .DW LIT
002191 0C89 002E                    .DW H'002E
002192 0C8B 07C1                    .DW MINS
002193 0C8D 05F2                    .DW ZERO
002194 0C8F 08BE                    .DW QERR
002195 0C91 05F2                    .DW ZERO
002196 0C93 00BD                    .DW BRCH
002197 0C95 0C71                    .DW NMB1
002198 0C97 048D            NMB2:   .DW DROP
002199 0C99 03AD                    .DW RG
002200 0C9B 00D0                    .DW ZBRCH
002201 0C9D 0CA1                    .DW NMB3
002202 0C9F 0458                    .DW DMIN
002203 0CA1 0379            NMB3:   .DW SEMIS
002204 0CA3                         ;
002205 0CA3                         ;
002206 0CA3 852D46494EC4            .DB H'85,"-FIN",H'C4 ; -FIND
002207 0CA9 0C4A                    .DW NMBR - 9
002208 0CAB 05C2            MFIND:  .DW NEST
002209 0CAD 060B                    .DW BL
002210 0CAF 0BB7                    .DW WORD
002211 0CB1 0784                    .DW HERE
002212 0CB3 0700                    .DW CNTX
002213 0CB5 04FA                    .DW AT
002214 0CB7 04FA                    .DW AT
002215 0CB9 0196                    .DW FIND
002216 0CBB 04B6                    .DW DUP
002217 0CBD 03D2                    .DW ZEQAL
002218 0CBF 00D0                    .DW ZBRCH
002219 0CC1 0CCB                    .DW MF1
002220 0CC3 048D                    .DW DROP
002221 0CC5 0784                    .DW HERE
002222 0CC7 0852                    .DW LTST
002223 0CC9 0196                    .DW FIND
002224 0CCB 0379            MF1:    .DW SEMIS
002225 0CCD                         ;
002226 0CCD                         ;
002227 0CCD 872841424F5254A9        .DB H'87,"(ABORT",H'A9 ; (ABORT)
002228 0CD5 0CA3                    .DW MFIND - 8
002229 0CD7 05C2            PABRT:  .DW NEST
002230 0CD9 0FBE                    .DW ABORT
002231 0CDB 0379                    .DW SEMIS
002232 0CDD                         ;
```

```
002233 0CDD 854552524FD2            .DB H'85,"ERRO",H'D2 ; ERROR
002234 0CE3 0CCD                    .DW PABRT - 10
002235 0CE5 05C2            ERROR:  .DW NEST
002236 0CE7 069A                    .DW WRNG
002237 0CE9 04FA                    .DW AT
002238 0CEB 03EB                    .DW ZLESS
002239 0CED 00D0                    .DW ZBRCH
002240 0CEF 0CF3                    .DW ERR1
002241 0CF1 0CD7                    .DW PABRT
002242 0CF3 0784            ERR1:   .DW HERE
002243 0CF5 09F4                    .DW CNT
002244 0CF7 0A07                    .DW TYPE
002245 0CF9 0A68                    .DW PDQ
002246 0CFB 0320203F                .DB H'03,"  ?"
002247 0CFF 148C                    .DW MSG
002248 0D01 0350                    .DW SP1
002249 0D03 06D1                    .DW FIN
002250 0D05 04FA                    .DW AT
002251 0D07 06C8                    .DW BLK
002252 0D09 04FA                    .DW AT
002253 0D0B 0F8D                    .DW QUIT
002254 0D0D 0379                    .DW SEMIS
002255 0D0F                         ;
002256 0D0F 834D49CE                .DB H'83,"MI",H'CE  ; MIN
002257 0D13 0CDD                    .DW ERROR - 8
002258 0D15 05C2            MIN:    .DW NEST
002259 0D17 0473                    .DW OVER
002260 0D19 0473                    .DW OVER
002261 0D1B 07E5                    .DW GTR
002262 0D1D 00D0                    .DW ZBRCH
002263 0D1F 0D23                    .DW MN1
002264 0D21 0499                    .DW SWAP
002265 0D23 048D            MN1:    .DW DROP
002266 0D25 0379                    .DW SEMIS
002267 0D27                         ;
002268 0D27 834944AE                .DB H'83,"ID",H'AE  ; ID.
002269 0D2B 0D0F                    .DW MIN - 6
002270 0D2D 05C2            ID:     .DW NEST
002271 0D2F 0BA4                    .DW PAD
002272 0D31 0086                    .DW LIT
002273 0D33 0020                    .DW H'0020
002274 0D35 0086                    .DW LIT
002275 0D37 005F                    .DW H'005F
002276 0D39 0B4C                    .DW FILL
002277 0D3B 04B6                    .DW DUP
002278 0D3D 0896                    .DW PFA
002279 0D3F 0862                    .DW LFA
002280 0D41 0473                    .DW OVER
002281 0D43 07C1                    .DW MINS
002282 0D45 0BA4                    .DW PAD
002283 0D47 0499                    .DW SWAP
002284 0D49 0246                    .DW CMOVE
002285 0D4B 0BA4                    .DW PAD
002286 0D4D 09F4                    .DW CNT
002287 0D4F 0086                    .DW LIT
002288 0D51 001F                    .DW H'001F
002289 0D53 02F3                    .DW FAND
```

```
002290 0D55 0A07                    .DW TYPE
002291 0D57 0807                    .DW SPC
002292 0D59 0379                    .DW SEMIS
002293 0D5B                         ;
002294 0D5B                         ;
002295 0D5B 864352454154C5          .DB H’86,"CREAT",H’C5 ; CREATE
002296 0D62 0D27                    .DW ID - 6
002297 0D64 05C2            CRTE:   .DW NEST
002298 0D66 033D                    .DW FSPAT
002299 0D68 0784                    .DW HERE
002300 0D6A 0086                    .DW LIT
002301 0D6C 00A0                    .DW H’00A0
002302 0D6E 03F7                    .DW PLUS
002303 0D70 07D9                    .DW LESS
002304 0D72 0602                    .DW TWO
002305 0D74 08BE                    .DW QERR
002306 0D76 0CAB                    .DW MFIND
002307 0D78 00D0                    .DW ZBRCH
002308 0D7A 0D8A                    .DW CRT1
002309 0D7C 048D                    .DW DROP
002310 0D7E 0880                    .DW NFA
002311 0D80 0D2D                    .DW ID
002312 0D82 0086                    .DW LIT
002313 0D84 0004                    .DW H’0004
002314 0D86 148C                    .DW MSG
002315 0D88 0807                    .DW SPC
002316 0D8A 0784            CRT1:   .DW HERE
002317 0D8C 04B6                    .DW DUP
002318 0D8E 050D                    .DW CAT
002319 0D90 068C                    .DW WIDTH
002320 0D92 04FA                    .DW AT
002321 0D94 0D15                    .DW MIN
002322 0D96 0768                    .DW PLUS1
002323 0D98 0794                    .DW ALLOT
002324 0D9A 04B6                    .DW DUP
002325 0D9C 0086                    .DW LIT
002326 0D9E 00A0                    .DW H’00A0
002327 0DA0 04E3                    .DW TGLE
002328 0DA2 0784                    .DW HERE
002329 0DA4 05FA                    .DW ONE
002330 0DA6 07C1                    .DW MINS
002331 0DA8 0086                    .DW LIT
002332 0DAA 0080                    .DW H’0080
002333 0DAC 04E3                    .DW TGLE
002334 0DAE 0852                    .DW LTST
002335 0DB0 07A0                    .DW COMMA
002336 0DB2 070E                    .DW CRNT
002337 0DB4 04FA                    .DW AT
002338 0DB6 051E                    .DW EX
002339 0DB8 0784                    .DW HERE
002340 0DBA 0775                    .DW PLUS2
002341 0DBC 07A0                    .DW COMMA
002342 0DBE 0379                    .DW SEMIS
&vellip;
002365 0DE4 8521434F44C5            .DB H’85,"!COD",H’C5 ; !CODE
002366 0DEA 0DC0                    .DW COLON - 4
002367 0DEC 05C2            DCODE:  .DW NEST
```

```
002368 0DEE 0D64                    .DW CRTE
002369 0DF0 098A                    .DW SMDG
002370 0DF2 0852                    .DW LTST
002371 0DF4 0896                    .DW PFA
002372 0DF6 0872                    .DW CFA
002373 0DF8 051E                    .DW EX
002374 0DFA 07A0                    .DW COMMA
002375 0DFC 0379                    .DW SEMIS
002376 0DFE                         ;
002377 0DFE                         ;
002378 0DFE 88434F4E5354414E        .DB H'88,"CONSTAN",H'D4 ; CONSTANT
       0E06 D4
002379 0E07 0DE4                    .DW DCODE - 8
002380 0E09 05C2            CNST:   .DW NEST
002381 0E0B 0086                    .DW LIT
002382 0E0D 05D6                    .DW CONST
002383 0E0F 0DEC                    .DW DCODE
002384 0E11 0379                    .DW SEMIS
002385 0E13                         ;
002386 0E13                         ;
002387 0E13 885641524941424C        .DB H'88,"VARIABL",H'C5 ; VARIABLE
       0E1B C5
002388 0E1C 0DFE                    .DW CNST - 11
002389 0E1E 05C2            VARB:   .DW NEST
002390 0E20 0086                    .DW LIT
002391 0E22 05CD                    .DW VAR
002392 0E24 0DEC                    .DW DCODE
002393 0E26 0379                    .DW SEMIS
002394 0E28                         ;
002395 0E28                         ;
002396 0E28 84555345D2              .DB H'84,"USE",H'D2 ; USER
002397 0E2D 0E13                    .DW VARB - 11
002398 0E2F 05C2            USR:    .DW NEST
002399 0E31 0086                    .DW LIT
002400 0E33 05DF                    .DW USER
002401 0E35 0DEC                    .DW DCODE
002402 0E37 0379                    .DW SEMIS
002403 0E39                         ;
002404 0E39                         ;
002405 0E39 873C4255494C44D3        .DB H'87,"<BUILD",H'D3 ; <BUILDS
002406 0E41 0E28                    .DW USR - 7
002407 0E43 05C2            LBLD:   .DW NEST
002408 0E45 05F2                    .DW ZERO
002409 0E47 0E09                    .DW CNST
002410 0E49 0379                    .DW SEMIS
002411 0E4B                         ;
002412 0E4B                         ;
002413 0E4B 85444F4553BE            .DB H'85,"DOES",H'BE ; DOES>
002414 0E51 0E39                    .DW LBLD - 10
002415 0E53 05C2            DOSEG:  .DW NEST
002416 0E55 03AD                    .DW RG
002417 0E57 0852                    .DW LTST
002418 0E59 0896                    .DW PFA
002419 0E5B 051E                    .DW EX
002420 0E5D 09C8                    .DW PCODE
002421 0E5F E2              DUZ1:   SEX R2
002422 0E60 9A                      GHI RA
```

```
002423 0E61 73                      STXD
002424 0E62 8A                      GLO RA
002425 0E63 73                      STXD
002426 0E64 4B                      LDA RB
002427 0E65 BA                      PHI RA
002428 0E66 4B                      LDA RB
002429 0E67 AA                      PLO RA
002430 0E68 19                      INC R9
002431 0E69 19                      INC R9
002432 0E6A 9B                      GHI RB
002433 0E6B 59                      STR R9
002434 0E6C 19                      INC R9
002435 0E6D 8B                      GLO RB
002436 0E6E 59                      STR R9
002437 0E6F 29                      DEC R9
002438 0E70 DC                      SEP RC
002439 0E71                         ;
002440 0E71                         ;
002441 0E71 C74C4954455241CC        .DB H'C7,"LITERA",H'CC ; LITERAL (IMMEDIATE)
002442 0E79 0E4B                    .DW DOSEG - 8
002443 0E7B 05C2            LTL:    .DW NEST
002444 0E7D 071A                    .DW STT
002445 0E7F 04FA                    .DW AT
002446 0E81 00D0                    .DW ZBRCH
002447 0E83 0E8B                    .DW LT1
002448 0E85 0951                    .DW CMPL
002449 0E87 0086                    .DW LIT
002450 0E89 07A0                    .DW COMMA
002451 0E8B 0379            LT1:    .DW SEMIS
002452 0E8D                         ;
002453 0E8D                         ;
002454 0E8D C8444C4954455241        .DB H'C8,"DLITERA",H'CC ; DLITERAL (IMMEDIATE)
       0E95 CC
002455 0E96 0E71                    .DW LTL - 10
002456 0E98 05C2            DLTL:   .DW NEST
002457 0E9A 071A                    .DW STT
002458 0E9C 04FA                    .DW AT
002459 0E9E 00D0                    .DW ZBRCH
002460 0EA0 0EA8                    .DW DLTL1
002461 0EA2 0499                    .DW SWAP
002462 0EA4 0E7B                    .DW LTL
002463 0EA6 0E7B                    .DW LTL
002464 0EA8 0379            DLTL1:  .DW SEMIS
002465 0EAA                         ;
002466 0EAA                         ;
002467 0EAA 863F53544143CB          .DB H'86,"?STAC",H'CB ; ?STACK
002468 0EB1 0E8D                    .DW DLTL - 11
002469 0EB3 05C2            QSTK:   .DW NEST
002470 0EB5 066D                    .DW SO
002471 0EB7 04FA                    .DW AT
002472 0EB9 04B6                    .DW DUP
002473 0EBB 033D                    .DW FSPAT
002474 0EBD 07E5                    .DW GTR
002475 0EBF 05FA                    .DW ONE
002476 0EC1 08BE                    .DW QERR
002477 0EC3 0086                    .DW LIT
002478 0EC5 0100                    .DW H'0100
```

```
002479 0EC7 03F7                    .DW PLUS
002480 0EC9 033D                    .DW FSPAT
002481 0ECB 07D9                    .DW LESS
002482 0ECD 0086                    .DW LIT
002483 0ECF 0007                    .DW H'0007
002484 0ED1 08BE                    .DW QERR
002485 0ED3 0379                    .DW SEMIS
002486 0ED5                         ;
002487 0ED5                         ;
002488 0ED5 89494E5445525052        .DB H'89,"INTERPRE",H'D4 ; INTERPRET
       0EDD 45D4
002489 0EDF 0EAA                    .DW QSTK - 9
002490 0EE1 05C2            INPT:   .DW NEST
002491 0EE3 0CAB                    .DW MFIND
002492 0EE5 00D0                    .DW ZBRCH
002493 0EE7 0F05                    .DW PT1
002494 0EE9 071A                    .DW STT
002495 0EEB 04FA                    .DW AT
002496 0EED 07D9                    .DW LESS
002497 0EEF 00D0                    .DW ZBRCH
002498 0EF1 0EFB                    .DW PT2
002499 0EF3 0872                    .DW CFA
002500 0EF5 07A0                    .DW COMMA
002501 0EF7 00BD                    .DW BRCH
002502 0EF9 0EFF                    .DW PT3
002503 0EFB 0872            PT2:    .DW CFA
002504 0EFD 00A6                    .DW EXE
002505 0EFF 0EB3            PT3:    .DW QSTK
002506 0F01 00BD                    .DW BRCH
002507 0F03 0F1F                    .DW PT4
002508 0F05 0784            PT1:    .DW HERE
002509 0F07 0C53                    .DW NMBR
002510 0F09 072F                    .DW DPL
002511 0F0B 04FA                    .DW AT
002512 0F0D 0768                    .DW PLUS1
002513 0F0F 00D0                    .DW ZBRCH
002514 0F11 0F19                    .DW PT5
002515 0F13 0E98                    .DW DLTL
002516 0F15 00BD                    .DW BRCH
002517 0F17 0F1D                    .DW PT6
002518 0F19 048D            PT5:    .DW DROP
002519 0F1B 0E7B                    .DW LTL
002520 0F1D 0EB3            PT6:    .DW QSTK
002521 0F1F 00BD            PT4:    .DW BRCH
002522 0F21 0EE3                    .DW INPT + 2
002523 0F23 0379                    .DW SEMIS
002524 0F25                         ;
002525 0F25                         ;
002526 0F25 8A564F434142554C        .DB H'8A,"VOCABULAR",H'D9 ; VOCABULARY
       0F2D 4152D9
002527 0F30 0ED5                    .DW INPT - 12
002528 0F32 05C2            VBLY:   .DW NEST
002529 0F34 0E43                    .DW LBLD
002530 0F36 0086                    .DW LIT
002531 0F38 81A0                    .DW H'81A0
002532 0F3A 07A0                    .DW COMMA
002533 0F3C 070E                    .DW CRNT
```

```
002534 0F3E 04FA                    .DW AT
002535 0F40 0872                    .DW CFA
002536 0F42 07A0                    .DW COMMA
002537 0F44 0784                    .DW HERE
002538 0F46 06BE                    .DW VL
002539 0F48 04FA                    .DW AT
002540 0F4A 07A0                    .DW COMMA
002541 0F4C 06BE                    .DW VL
002542 0F4E 051E                    .DW EX
002543 0F50 0E53                    .DW DOSEG
002544 0F52 0775            VB1:    .DW PLUS2
002545 0F54 0700                    .DW CNTX
002546 0F56 051E                    .DW EX
002547 0F58 0379                    .DW SEMIS
002548 0F5A                         ;
002549 0F5A                         ;
002550 0F5A C5464F5254C8    FRTH:   .DB H'C5,"FORT",H'C8 ; FORTH (IMMEDIATE)
002551 0F60 0F25                    .DW VBLY - 13
002552 0F62 0E5F                    .DW DUZ1
002553 0F64 0F52                    .DW VB1
002554 0F66 81A0                    .DW H'81A0
002555 0F68 18BF                    .DW TASK - 7        ; DICTION LINK
002556 0F6A 0000                    .DW H'0000          ; NOTE THIS MUST BE CHANGED
002557 0F6C                                             ; TO REFLECT THE LINK TO THE
002558 0F6C                                             ; LAST DICT WORD IN THE
002559 0F6C                                             ; FORTH VOCAB
002560 0F6C                         ;
002561 0F6C                         ;
002562 0F6C 8B444546494E4954        .DB H'8B,"DEFINITION",H'D3 ; DEFINITIONS
       0F74 494F4ED3
002563 0F78 0F5A                    .DW FRTH
002564 0F7A 05C2            DFN:    .DW NEST
002565 0F7C 0700                    .DW CNTX
002566 0F7E 04FA                    .DW AT
002567 0F80 070E                    .DW CRNT
002568 0F82 051E                    .DW EX
002569 0F84 0379                    .DW SEMIS
002570 0F86                         ;
002571 0F86 84515549D4              .DB H'84,"QUI",H'D4 ; QUIT
002572 0F8B 0F6C                    .DW DFN - 14
002573 0F8D 05C2            QUIT:   .DW NEST
002574 0F8F 05F2                    .DW ZERO
002575 0F91 06C8                    .DW BLK
002576 0F93 051E                    .DW EX
002577 0F95 0967                    .DW LB
002578 0F97 0365            Q2:     .DW RP1
002579 0F99 05AB                    .DW CR
002580 0F9B 0AF7                    .DW QUER
002581 0F9D 0EE1                    .DW INPT
002582 0F9F 071A                    .DW STT
002583 0FA1 04FA                    .DW AT
002584 0FA3 03D2                    .DW ZEQAL
002585 0FA5 00D0                    .DW ZBRCH
002586 0FA7 0FB0                    .DW Q1
002587 0FA9 0A68                    .DW PDQ
002588 0FAB 0420204F4B              .DB H'04,"  OK"
002589 0FB0 00BD            Q1:     .DW BRCH
```

```
002590 0FB2 0F97                    .DW Q2
002591 0FB4 0379                    .DW SEMIS
002592 0FB6                         ;
002593 0FB6                         ;
002594 0FB6 8541424F52D4            .DB H'85,"ABOR",H'D4 ; ABORT
002595 0FBC 0F86                    .DW QUIT - 7
002596 0FBE 05C2            ABORT:  .DW NEST
002597 0FC0 0350                    .DW SP1
002598 0FC2 09B2                    .DW MDCML
002599 0FC4 05AB                    .DW CR
002600 0FC6 0A68                    .DW PDQ
002601 0FC8 1C31383032204649        .DB H'1C,"1802 FIG-FORTH R0.4  3/16/81"
       0FD0 472D464F52544820
       0FD8 52302E342020332F
       0FE0 31362F3831
002602 0FE5 17B5             .DW DRZER
002603 0FE7 15AB                    .DW MTBUF
002604 0FE9 0621                    .DW FIRST
002605 0FEB 04B6                    .DW DUP
002606 0FED 154D                    .DW PREV
002607 0FEF 051E                    .DW EX
002608 0FF1 1542                    .DW USE
002609 0FF3 051E                    .DW EX
002610 0FF5 0F62                    .DW FRTH + 8
002611 0FF7 0F7A                    .DW DFN
002612 0FF9 0F8D                    .DW QUIT
002613 0FFB 0379                    .DW SEMIS
002614 0FFD                         ;
002615 0FFD                         ;
002616 0FFD C1BB                    .DW H'C1BB          ; ;   (IMMEDIATE)
002617 0FFF 0FB6                    .DW ABORT - 8
002618 1001 05C2            SEMIC:  .DW NEST
002619 1003 091A                    .DW QCSP
002620 1005 0951                    .DW CMPL
002621 1007 0379                    .DW SEMIS
002622 1009 098A                    .DW SMDG
002623 100B 0967                    .DW LB
002624 100D 0379                    .DW SEMIS
002625 100F                         ;
002626 100F                         ;
002627 100F                         ;
002628 100F C22EA2                  .DB H'C2,H'2E,H'A2  ; ."   (IMMEDIATE)
002629 1012 0FFD                    .DW SEMIC - 4
002630 1014 05C2            DOTQ:   .DW NEST
002631 1016 0086                    .DW LIT
002632 1018 0022                    .DW H'0022
002633 101A 071A                    .DW STT
002634 101C 04FA                    .DW AT
002635 101E 00D0                    .DW ZBRCH
002636 1020 1034                    .DW DOTQ1
002637 1022 0951                    .DW CMPL
002638 1024 0A68                    .DW PDQ
002639 1026 0BB7                    .DW WORD
002640 1028 0784                    .DW HERE
002641 102A 050D                    .DW CAT
002642 102C 0768                    .DW PLUS1
002643 102E 0794                    .DW ALLOT
```

```
002644 1030 00BD                    .DW BRCH
002645 1032 103C                    .DW DOTQ2
002646 1034 0BB7            DOTQ1:  .DW WORD
002647 1036 0784                    .DW HERE
002648 1038 09F4                    .DW CNT
002649 103A 0A07                    .DW TYPE
002650 103C 0379            DOTQ2:  .DW SEMIS
002651 103E                         ;
002652 103E                         ;
002653 103E C95B434F4D50494C        .DB H'C9,"[COMPILE",H'DD ; [COMPILE]
       1046 45DD
002654 1048 100F                    .DW DOTQ - 5            ; (IMMEDIATE)
002655 104A 05C2            BCOMP:  .DW NEST
002656 104C 0CAB                    .DW MFIND
002657 104E 03D2                    .DW ZEQAL
002658 1050 05F2                    .DW ZERO
002659 1052 08BE                    .DW QERR
002660 1054 048D                    .DW DROP
002661 1056 0872                    .DW CFA
002662 1058 07A0                    .DW COMMA
002663 105A 0379                    .DW SEMIS
002664 105C                         ;
002665 105C                         ;
002666 105C 89494D4D45444941        .DB H'89,"IMMEDIAT",H'C5 ; IMMEDIATE
       1064 54C5
002667 1066 103E                    .DW BCOMP - 12
002668 1068 05C2            IMMED:  .DW NEST
002669 106A 0852                    .DW LTST
002670 106C 0086                    .DW LIT
002671 106E 0040                    .DW H'0040
002672 1070 04E3                    .DW TGLE
002673 1072 0379                    .DW SEMIS
002674 1074                         ;
002675 1074                         ;
002676 1074 C1A8                    .DW H'C1A8          ; (   (IMMEDIATE)
002677 1076 105C                    .DW IMMED - 12
002678 1078 05C2            PAREN:  .DW NEST
002679 107A 0086                    .DW LIT
002680 107C 0029                    .DW H'0029
002681 107E 0BB7                    .DW WORD
002682 1080 0379                    .DW SEMIS
002683 1082                         ;
002684 1082                         ;
002685 1082 81B3                    .DW H'81B3          ; 3
002686 1084 1074                    .DW PAREN - 4
002687 1086 05D6            THREE:  .DW CONST
002688 1088 0003                    .DW H'0003
002689 108A                         ;
002690 108A                         ;
002691 108A C1A7                    .DW H'C1A7          ; '   (TICK)   (IMMEDIATE)
002692 108C 1082                    .DW THREE - 4
002693 108E 05C2            TICK:   .DW NEST
002694 1090 0CAB                    .DW MFIND
002695 1092 03D2                    .DW ZEQAL
002696 1094 05F2                    .DW ZERO
002697 1096 08BE                    .DW QERR
002698 1098 048D                    .DW DROP
```

```
002699 109A 0E7B                    .DW LTL
002700 109C 0379                    .DW SEMIS
002701 109E                         ;
002702 109E                         ;
002703 109E 86464F524745D4          .DB H'86,"FORGE",H'D4 ; FORGET
002704 10A5 108A                    .DW TICK - 4
002705 10A7 05C2            FORG:   .DW NEST
002706 10A9 070E                    .DW CRNT
002707 10AB 04FA                    .DW AT
002708 10AD 0700                    .DW CNTX
002709 10AF 04FA                    .DW AT
002710 10B1 07C1                    .DW MINS
002711 10B3 0086                    .DW LIT
002712 10B5 0018                    .DW H'0018
002713 10B7 08BE                    .DW QERR
002714 10B9 108E                    .DW TICK
002715 10BB 04B6                    .DW DUP
002716 10BD 06A6                    .DW FNCE
002717 10BF 04FA                    .DW AT
002718 10C1 07D9                    .DW LESS
002719 10C3 0086                    .DW LIT
002720 10C5 0015                    .DW H'0015
002721 10C7 08BE                    .DW QERR
002722 10C9 04B6                    .DW DUP
002723 10CB 0880                    .DW NFA
002724 10CD 06AF                    .DW DP
002725 10CF 051E                    .DW EX
002726 10D1 0862                    .DW LFA
002727 10D3 04FA                    .DW AT
002728 10D5 0700                    .DW CNTX
002729 10D7 04FA                    .DW AT
002730 10D9 051E                    .DW EX
002731 10DB 0379                    .DW SEMIS
002732 10DD                         ;
002733 10DD                         ;
002734 10DD 822BAD                  .DB H'82,H'2B,H'AD  ; +-
002735 10E0 109E                    .DW FORG - 9
002736 10E2 05C2            PM:     .DW NEST
002737 10E4 03EB                    .DW ZLESS
002738 10E6 00D0                    .DW ZBRCH
002739 10E8 10EC                    .DW PM1
002740 10EA 0412                    .DW MINUS
002741 10EC 0379            PM1:    .DW SEMIS
002742 10EE                         ;
002743 10EE 83442BAD                .DB H'83,H'44,H'2B,H'AD ; D+-
002744 10F2 10DD                    .DW PM - 5
002745 10F4 05C2            DPM:    .DW NEST
002746 10F6 03EB                    .DW ZLESS
002747 10F8 00D0                    .DW ZBRCH
002748 10FA 10FE                    .DW DPM1
002749 10FC 0458                    .DW DMIN
002750 10FE 0379            DPM1:   .DW SEMIS
002751 1100                         ;
002752 1100 834142D3                .DB H'83,H'41,H'42,H'D3 ; ABS
002753 1104 10EE                    .DW DPM - 6
002754 1106 05C2            ABS:    .DW NEST
002755 1108 04B6                    .DW DUP
```

```
002756 110A 10E2                    .DW PM
002757 110C 0379                    .DW SEMIS
002758 110E                         ;
002759 110E 84444142D3              .DB H'84,"DAB",H'D3 ; DABS
002760 1113 1100                    .DW ABS - 6
002761 1115 05C2            DABS:   .DW NEST
002762 1117 04B6                    .DW DUP
002763 1119 10F4                    .DW DPM
002764 111B 0379                    .DW SEMIS
002765 111D                         ;
002766 111D 834D41D8                .DB H'83,H'4D,H'41,H'D8 ; MAX
002767 1121 110E                    .DW DABS - 7
002768 1123 05C2            MAX:    .DW NEST
002769 1125 0473                    .DW OVER
002770 1127 0473                    .DW OVER
002771 1129 07D9                    .DW LESS
002772 112B 00D0                    .DW ZBRCH
002773 112D 1131                    .DW MAX1
002774 112F 0499                    .DW SWAP
002775 1131 048D            MAX1:   .DW DROP
002776 1133 0379                    .DW SEMIS
002777 1135                         ;
002778 1135 824DAA                  .DB H'82,H'4D,H'AA  ; M*
002779 1138 111D                    .DW MAX - 6
002780 113A 05C2            MSTAR:  .DW NEST
002781 113C 0473                    .DW OVER
002782 113E 0473                    .DW OVER
002783 1140 0324                    .DW FXOR
002784 1142 039D                    .DW GR
002785 1144 1106                    .DW ABS
002786 1146 0499                    .DW SWAP
002787 1148 1106                    .DW ABS
002788 114A 0278                    .DW USTAR
002789 114C 03AD                    .DW RG
002790 114E 10F4                    .DW DPM
002791 1150 0379                    .DW SEMIS
002792 1152                         ;
002793 1152 824DAF                  .DB H'82,H'4D,H'AF  ; M/
002794 1155 1135                    .DW MSTAR - 5
002795 1157 05C2            MSLAS:  .DW NEST
002796 1159 0473                    .DW OVER
002797 115B 039D                    .DW GR
002798 115D 039D                    .DW GR
002799 115F 1115                    .DW DABS
002800 1161 03BD                    .DW R
002801 1163 1106                    .DW ABS
002802 1165 02B2                    .DW USLSH
002803 1167 03AD                    .DW RG
002804 1169 03BD                    .DW R
002805 116B 0324                    .DW FXOR
002806 116D 10E2                    .DW PM
002807 116F 0499                    .DW SWAP
002808 1171 03AD                    .DW RG
002809 1173 10E2                    .DW PM
002810 1175 0499                    .DW SWAP
002811 1177 0379                    .DW SEMIS
002812 1179                         ;
```

```
002813 1179 81AA                    .DW H'81AA          ; *
002814 117B 1152                    .DW MSLAS - 5
002815 117D 05C2            STAR:   .DW NEST
002816 117F 113A                    .DW MSTAR
002817 1181 048D                    .DW DROP
002818 1183 0379                    .DW SEMIS
002819 1185                         ;
002820 1185 842F4D4FC4              .DB H'84,H'2F,H'4D,H'4F,H'C4 ; /MOD
002821 118A 1179                    .DW STAR - 4
002822 118C 05C2            SLMOD:  .DW NEST
002823 118E 039D                    .DW GR
002824 1190 14F1                    .DW STOD
002825 1192 03AD                    .DW RG
002826 1194 1157                    .DW MSLAS
002827 1196 0379                    .DW SEMIS
002828 1198                         ;
002829 1198 81AF                    .DW H'81AF          ; /
002830 119A 1185                    .DW SLMOD - 7
002831 119C 05C2            SLASH:  .DW NEST
002832 119E 118C                    .DW SLMOD
002833 11A0 0499                    .DW SWAP
002834 11A2 048D                    .DW DROP
002835 11A4 0379                    .DW SEMIS
002836 11A6                         ;
002837 11A6 834D4FC4                .DB H'83,H'4D,H'4F,H'C4 ; MOD
002838 11AA 1198                    .DW SLASH - 4
002839 11AC 05C2            MODD:   .DW NEST
002840 11AE 118C                    .DW SLMOD
002841 11B0 048D                    .DW DROP
002842 11B2 0379                    .DW SEMIS
002843 11B4                         ;
002844 11B4 852A2F4D4FC4            .DB H'85,"*/MO",H'C4 ; */MOD
002845 11BA 11A6                    .DW MODD - 6
002846 11BC 05C2            SSMOD:  .DW NEST
002847 11BE 039D                    .DW GR
002848 11C0 113A                    .DW MSTAR
002849 11C2 03AD                    .DW RG
002850 11C4 1157                    .DW MSLAS
002851 11C6 0379                    .DW SEMIS
002852 11C8                         ;
002853 11C8 822AAF                  .DB H'82,H'2A,H'AF ; */
002854 11CB 11B4                    .DW SSMOD - 8
002855 11CD 05C2            SSLA:   .DW NEST
002856 11CF 11BC                    .DW SSMOD
002857 11D1 0499                    .DW SWAP
002858 11D3 048D                    .DW DROP
002859 11D5 0379                    .DW SEMIS
002860 11D7                         ;
002861 11D7 854D2F4D4FC4            .DB H'85,"M/MO",H'C4 ; M/MOD
002862 11DD 11C8                    .DW SSLA - 5
002863 11DF 05C2            MSMOD:  .DW NEST
002864 11E1 039D                    .DW GR
002865 11E3 05F2                    .DW ZERO
002866 11E5 03BD                    .DW R
002867 11E7 02B2                    .DW USLSH
002868 11E9 03AD                    .DW RG
002869 11EB 0499                    .DW SWAP
```

```
002870 11ED 039D                    .DW GR
002871 11EF 02B2                    .DW USLSH
002872 11F1 03AD                    .DW RG
002873 11F3 0379                    .DW SEMIS
002874 11F5                         ;
002875 11F5                         ;
002876 11F5 834D4FCE                .DB H'83,"MO",H'CE  ; MON
002877 11F9 11D7                    .DW MSMOD - 8       ; RETURN TO MONITOR
002878 11FB 11FD            MON:    .DW * + 2           ; AT 8000 HEX
002879 11FD F880                    LDI H'80
002880 11FF B0                      PHI R0
002881 1200 F800                    LDI H'00
002882 1202 A0                      PLO R0
002883 1203 E0                      SEX R0
002884 1204 D0                      SEP R0
002885 1205                         ;
002886 1205                         ;
002887 1205 834259C5                .DB H'83,"BY",H'C5  ; BYE
002888 1209 11F5                    .DW MON - 6
002889 120B 05C2            BYE:    .DW NEST
002890 120D 189B                    .DW FLUSH
002891 120F 11FB                    .DW MON
002892 1211                         ;
002893 1211                         ;
002894 1211 84424143CB              .DB H'84,"BAC",H'CB ; BACK
002895 1216 1205                    .DW BYE - 6
002896 1218 05C2            BACK:   .DW NEST
002897 121A 07A0                    .DW COMMA
002898 121C 0379                    .DW SEMIS
002899 121E                         ;
002900 121E C542454749CE            .DB H'C5,"BEGI",H'CE ; BEGIN
002901 1224 1211                    .DW BACK - 7
002902 1226 05C2            BEGIN:  .DW NEST
002903 1228 08D8                    .DW QCMP
002904 122A 0784                    .DW HERE
002905 122C 05FA                    .DW ONE
002906 122E 0379                    .DW SEMIS
002907 1230                         ;
002908 1230 C5454E4449C6            .DB H'C5,"ENDI",H'C6 ; ENDIF
002909 1236 121E                    .DW BEGIN - 8
002910 1238 05C2            ENDIFF: .DW NEST
002911 123A 08D8                    .DW QCMP
002912 123C 0602                    .DW TWO
002913 123E 0907                    .DW QPR
002914 1240 0784                    .DW HERE
002915 1242 0499                    .DW SWAP
002916 1244 051E                    .DW EX
002917 1246 0379                    .DW SEMIS
002918 1248                         ;
002919 1248 C4544845CE              .DB H'C4,"THE",H'CE ; THEN
002920 124D 1230                    .DW ENDIFF - 8
002921 124F 05C2            THEN:   .DW NEST
002922 1251 1238                    .DW ENDIFF
002923 1253 0379                    .DW SEMIS
002924 1255                         ;
002925 1255 C244CF                  .DB H'C2,H'44,H'CF  ; DO
002926 1258 1248                    .DW THEN - 7
```

```
002927 125A 05C2            DO:     .DW NEST
002928 125C 0951                    .DW CMPL
002929 125E 0150                    .DW PDO
002930 1260 0784                    .DW HERE
002931 1262 1086                    .DW THREE
002932 1264 0379                    .DW SEMIS
002933 1266                         ;
002934 1266 C44C4F4FD0              .DB H'C4,H'4C,H'4F,H'4F,H'D0 ; LOOP
002935 126B 1255                    .DW DO - 5
002936 126D 05C2            LOOP:   .DW NEST
002937 126F 1086                    .DW THREE
002938 1271 0907                    .DW QPR
002939 1273 0951                    .DW CMPL
002940 1275 00EC                    .DW LUPE
002941 1277 1218                    .DW BACK
002942 1279 0379                    .DW SEMIS
002943 127B                         ;
002944 127B C52B4C4F4FD0            .DB H'C5,"+LOO",H'D0 ; +LOOP
002945 1281 1266                    .DW LOOP - 7
002946 1283 05C2            PLOOP:  .DW NEST
002947 1285 1086                    .DW THREE
002948 1287 0907                    .DW QPR
002949 1289 0951                    .DW CMPL
002950 128B 0124                    .DW PLUPE
002951 128D 1218                    .DW BACK
002952 128F 0379                    .DW SEMIS
002953 1291                         ;
002954 1291 C5554E5449CC            .DB H'C5,"UNTI",H'CC ; UNTIL
002955 1297 127B                    .DW PLOOP - 8
002956 1299 05C2            UNTIL:  .DW NEST
002957 129B 05FA                    .DW ONE
002958 129D 0907                    .DW QPR
002959 129F 0951                    .DW CMPL
002960 12A1 00D0                    .DW ZBRCH
002961 12A3 1218                    .DW BACK
002962 12A5 0379                    .DW SEMIS
002963 12A7                         ;
002964 12A7 C3454EC4                .DB H'C3,H'45,H'4E,H'C4 ; END
002965 12AB 1291                    .DW UNTIL - 8
002966 12AD 05C2            ENDD:   .DW NEST
002967 12AF 1299                    .DW UNTIL
002968 12B1 0379                    .DW SEMIS
002969 12B3                         ;
002970 12B3 C541474149CE            .DB H'C5,"AGAI",H'CE ; AGAIN
002971 12B9 12A7                    .DW ENDD - 6
002972 12BB 05C2            AGAIN:  .DW NEST
002973 12BD 05FA                    .DW ONE
002974 12BF 0907                    .DW QPR
002975 12C1 0951                    .DW CMPL
002976 12C3 00BD                    .DW BRCH
002977 12C5 1218                    .DW BACK
002978 12C7 0379                    .DW SEMIS
002979 12C9                         ;
002980 12C9 C65245504541D4          .DB H'C6,"REPEA",H'D4 ; REPEAT
002981 12D0 12B3                    .DW AGAIN - 8
002982 12D2 05C2            REPEA:  .DW NEST
002983 12D4 039D                    .DW GR
```

```
002984 12D6 039D                    .DW GR
002985 12D8 12BB                    .DW AGAIN
002986 12DA 03AD                    .DW RG
002987 12DC 03AD                    .DW RG
002988 12DE 0602                    .DW TWO
002989 12E0 07C1                    .DW MINS
002990 12E2 1238                    .DW ENDIFF
002991 12E4 0379                    .DW SEMIS
002992 12E6                         ;
002993 12E6 C249C6                  .DB H'C2,H'49,H'C6 ; IF
002994 12E9 12C9                    .DW REPEA - 9
002995 12EB 05C2            IFF:    .DW NEST
002996 12ED 0951                    .DW CMPL
002997 12EF 00D0                    .DW ZBRCH
002998 12F1 0784                    .DW HERE
002999 12F3 05F2                    .DW ZERO
003000 12F5 07A0                    .DW COMMA
003001 12F7 0602                    .DW TWO
003002 12F9 0379                    .DW SEMIS
003003 12FB                         ;
003004 12FB C4454C53C5              .DB H'C4,"ELS",H'C5 ; ELSE
003005 1300 12E6                    .DW IFF - 5
003006 1302 05C2            ELSEE:  .DW NEST
003007 1304 0602                    .DW TWO
003008 1306 0907                    .DW QPR
003009 1308 0951                    .DW CMPL
003010 130A 00BD                    .DW BRCH
003011 130C 0784                    .DW HERE
003012 130E 05F2                    .DW ZERO
003013 1310 07A0                    .DW COMMA
003014 1312 0499                    .DW SWAP
003015 1314 0602                    .DW TWO
003016 1316 1238                    .DW ENDIFF
003017 1318 0602                    .DW TWO
003018 131A 0379                    .DW SEMIS
003019 131C                         ;
003020 131C C55748494CC5            .DB H'C5,"WHIL",H'C5 ; WHILE
003021 1322 12FB                    .DW ELSEE - 7
003022 1324 05C2            WHILE:  .DW NEST
003023 1326 12EB                    .DW IFF
003024 1328 0775                    .DW PLUS2
003025 132A 0379                    .DW SEMIS
003026 132C                         ;
003027 132C 865350414345D3          .DB H'86,"SPACE",H'D3 ; SPACES
003028 1333 131C                    .DW WHILE - 8
003029 1335 05C2            SPACS:  .DW NEST
003030 1337 05F2                    .DW ZERO
003031 1339 1123                    .DW MAX
003032 133B 0816                    .DW MDUP
003033 133D 00D0                    .DW ZBRCH
003034 133F 134B                    .DW SPAX1
003035 1341 05F2                    .DW ZERO
003036 1343 0150                    .DW PDO
003037 1345 0807            SPAX2:  .DW SPC
003038 1347 00EC                    .DW LUPE
003039 1349 1345                    .DW SPAX2
003040 134B 0379            SPAX1:  .DW SEMIS
```

```
003041 134D                        ;
003042 134D 823CA3                 .DB H'82,H'3C,H'A3  ; <#
003043 1350 132C                   .DW SPACS - 9
003044 1352 05C2           BDIGS:  .DW NEST
003045 1354 0BA4                   .DW PAD
003046 1356 0756                   .DW HLD
003047 1358 051E                   .DW EX
003048 135A 0379                   .DW SEMIS
003049 135C                        ;
003050 135C 8223BE                 .DB H'82,H'23,H'BE  ; #>
003051 135F 134D                   .DW BDIGS - 5
003052 1361 05C2           EDIGS:  .DW NEST
003053 1363 048D                   .DW DROP
003054 1365 048D                   .DW DROP
003055 1367 0756                   .DW HLD
003056 1369 04FA                   .DW AT
003057 136B 0BA4                   .DW PAD
003058 136D 0473                   .DW OVER
003059 136F 07C1                   .DW MINS
003060 1371 0379                   .DW SEMIS
003061 1373                        ;
003062 1373 84534947CE             .DB H'84,"SIG",H'CE ; SIGN
003063 1378 135C                   .DW EDIGS - 5
003064 137A 05C2           SIGN:   .DW NEST
003065 137C 07F3                   .DW ROT
003066 137E 03EB                   .DW ZLESS
003067 1380 00D0                   .DW ZBRCH
003068 1382 138A                   .DW SIGN1
003069 1384 0086                   .DW LIT
003070 1386 002D                   .DW H'002D
003071 1388 0B8C                   .DW HOLD
003072 138A 0379           SIGN1:  .DW SEMIS
003073 138C                        ;
003074 138C 81A3                   .DW H'81A3          ; #
003075 138E 1373                   .DW SIGN - 7
003076 1390 05C2           DIG:    .DW NEST
003077 1392 0725                   .DW BASE
003078 1394 04FA                   .DW AT
003079 1396 11DF                   .DW MSMOD
003080 1398 07F3                   .DW ROT
003081 139A 0086                   .DW LIT
003082 139C 0009                   .DW H'0009
003083 139E 0473                   .DW OVER
003084 13A0 07D9                   .DW LESS
003085 13A2 00D0                   .DW ZBRCH
003086 13A4 13AC                   .DW DIG1
003087 13A6 0086                   .DW LIT
003088 13A8 0007                   .DW H'0007
003089 13AA 03F7                   .DW PLUS
003090 13AC 0086           DIG1:   .DW LIT
003091 13AE 0030                   .DW H'0030
003092 13B0 03F7                   .DW PLUS
003093 13B2 0B8C                   .DW HOLD
003094 13B4 0379                   .DW SEMIS
003095 13B6                        ;
003096 13B6 8223D3                 .DB H'82,H'23,H'D3  ; #S
003097 13B9 138C                   .DW DIG - 4
```

```
003098 13BB 05C2            DIGS:   .DW NEST
003099 13BD 1390            DIGS1:  .DW DIG
003100 13BF 0473                    .DW OVER
003101 13C1 0473                    .DW OVER
003102 13C3 030B                    .DW FFOR
003103 13C5 03D2                    .DW ZEQAL
003104 13C7 00D0                    .DW ZBRCH
003105 13C9 13BD                    .DW DIGS1
003106 13CB 0379                    .DW SEMIS
003107 13CD                         ;
003108 13CD 83442ED2                .DB H'83,H'44,H'2E,H'D2 ; D.R
003109 13D1 13B6                    .DW DIGS - 5
003110 13D3 05C2            DDOTR:  .DW NEST
003111 13D5 039D                    .DW GR
003112 13D7 0499                    .DW SWAP
003113 13D9 0473                    .DW OVER
003114 13DB 1115                    .DW DABS
003115 13DD 1352                    .DW BDIGS
003116 13DF 13BB                    .DW DIGS
003117 13E1 137A                    .DW SIGN
003118 13E3 1361                    .DW EDIGS
003119 13E5 03AD                    .DW RG
003120 13E7 0473                    .DW OVER
003121 13E9 07C1                    .DW MINS
003122 13EB 1335                    .DW SPACS
003123 13ED 0A07                    .DW TYPE
003124 13EF 0379                    .DW SEMIS
003125 13F1                         ;
003126 13F1 822ED2                  .DB H'82,H'2E,H'D2  ; .R
003127 13F4 13CD                    .DW DDOTR - 6
003128 13F6 05C2            DOTR:   .DW NEST
003129 13F8 039D                    .DW GR
003130 13FA 14F1                    .DW STOD
003131 13FC 03AD                    .DW RG
003132 13FE 13D3                    .DW DDOTR
003133 1400 0379                    .DW SEMIS
003134 1402                         ;
003135 1402 8244AE                  .DB H'82,H'44,H'AE  ; D.
003136 1405 13F1                    .DW DOTR - 5
003137 1407 05C2            DDOT:   .DW NEST
003138 1409 05F2                    .DW ZERO
003139 140B 13D3                    .DW DDOTR
003140 140D 0807                    .DW SPC
003141 140F 0379                    .DW SEMIS
003142 1411                         ;
003143 1411 81AE                    .DW H'81AE          ; .   (DOT)
003144 1413 1402                    .DW DDOT - 5
003145 1415 05C2            DOT:    .DW NEST
003146 1417 14F1                    .DW STOD
003147 1419 1407                    .DW DDOT
003148 141B 0379                    .DW SEMIS
003149 141D                         ;
003150 141D 81BF                    .DW H'81BF          ; ?
003151 141F 1411                    .DW DOT - 4
003152 1421 05C2            QUES:   .DW NEST
003153 1423 04FA                    .DW AT
003154 1425 1415                    .DW DOT
```

```
003155 1427 0379                    .DW SEMIS
003156 1429                         ;
003157 1429 8255AE                  .DB H’82,H’55,H’AE  ; U.
003158 142C 141D                    .DW QUES - 4
003159 142E 05C2            UDOT:   .DW NEST
003160 1430 05F2                    .DW ZERO
003161 1432 1407                    .DW DDOT
003162 1434 0379                    .DW SEMIS
003163 1436                         ;
003164 1436 85564C4953D4            .DB H’85,"VLIS",H’D4 ; VLIST
003165 143C 1429                    .DW UDOT - 5
003166 143E 05C2            VLIST:  .DW NEST
003167 1440 05AB                    .DW CR
003168 1442 0086                    .DW LIT
003169 1444 0080                    .DW H’0080
003170 1446 06DB                    .DW FOUT
003171 1448 051E                    .DW EX
003172 144A 0700                    .DW CNTX
003173 144C 04FA                    .DW AT
003174 144E 04FA                    .DW AT
003175 1450 06DB            VLIS1:  .DW FOUT
003176 1452 04FA                    .DW AT
003177 1454 0615                    .DW CL
003178 1456 07E5                    .DW GTR
003179 1458 00D0                    .DW ZBRCH
003180 145A 1464                    .DW VLIS2
003181 145C 05AB                    .DW CR
003182 145E 05F2                    .DW ZERO
003183 1460 06DB                    .DW FOUT
003184 1462 051E                    .DW EX
003185 1464 04B6            VLIS2:  .DW DUP
003186 1466 0D2D                    .DW ID
003187 1468 0807                    .DW SPC
003188 146A 0807                    .DW SPC
003189 146C 0896                    .DW PFA
003190 146E 0862                    .DW LFA
003191 1470 04FA                    .DW AT
003192 1472 04B6                    .DW DUP
003193 1474 03D2                    .DW ZEQAL
003194 1476 0597                    .DW QTERM
003195 1478 030B                    .DW FFOR
003196 147A 00D0                    .DW ZBRCH
003197 147C 1450                    .DW VLIS1
003198 147E 048D                    .DW DROP
003199 1480 0379                    .DW SEMIS
003200 1482                         ;
003201 1482                         ;
003202 1482 874D4553534147C5        .DB H’87,"MESSAG",H’C5 ; MESSAGE
003203 148A 1436                    .DW VLIST - 8
003204 148C 05C2            MSG:    .DW NEST
003205 148E 069A                    .DW WRNG
003206 1490 04FA                    .DW AT
003207 1492 00D0                    .DW ZBRCH
003208 1494 14B2                    .DW MESS1
003209 1496 0816                    .DW MDUP
003210 1498 00D0                    .DW ZBRCH
003211 149A 14AE                    .DW MESS2
```

```
003212 149C 0086                    .DW LIT
003213 149E 0004                    .DW H'0004
003214 14A0 06F2                    .DW OFST
003215 14A2 04FA                    .DW AT
003216 14A4 0645                    .DW BSCR
003217 14A6 119C                    .DW SLASH
003218 14A8 07C1                    .DW MINS
003219 14AA 1532                    .DW DLINE
003220 14AC 0807                    .DW SPC
003221 14AE 00BD            MESS2:  .DW BRCH
003222 14B0 14BE                    .DW MESS3
003223 14B2 0A68            MESS1:  .DW PDQ
003224 14B4 07204D5347202320        .DB H'07," MSG # "
003225 14BC 1415                    .DW DOT
003226 14BE 0379            MESS3:  .DW SEMIS
003227 14C0                         ;
003228 14C0                         ;
003229 14C0 81C9                    .DW H'81C9          ; I
003230 14C2 1482                    .DW MSG - 10
003231 14C4 14C6            I:      .DW * + 2
003232 14C6 12                      INC R2
003233 14C7 19                      INC R9
003234 14C8 19                      INC R9
003235 14C9 19                      INC R9
003236 14CA 42                      LDA R2
003237 14CB 59                      STR R9
003238 14CC 29                      DEC R9
003239 14CD 02                      LDN R2
003240 14CE 59                      STR R9
003241 14CF 22                      DEC R2
003242 14D0 22                      DEC R2
003243 14D1 DC                      SEP RC
003244 14D2                         ;
003245 14D2 84574152CD              .DB H'84,"WAR",H'CD ; WARM
003246 14D7 14C0                    .DW I - 4
003247 14D9 14DB            WRM:    .DW * + 2
003248 14DB C01915                  LBR WARM
003249 14DE                         ;
003250 14DE                         ;
003251 14DE 84434F4CC4              .DB H'84,"COL",H'C4 ; COLD
003252 14E3 14D2                    .DW WRM - 7
003253 14E5 14E7            CLD:    .DW * + 2
003254 14E7 C01900                  LBR COLD
003255 14EA                         ;
003256 14EA                         ;
003257 14EA 84532D3EC4              .DB H'84,"S->",H'C4 ; S->D
003258 14EF 14DE                    .DW CLD - 7
003259 14F1 14F3            STOD:   .DW * + 2
003260 14F3 49                      LDA R9
003261 14F4 FE                      SHL
003262 14F5 33FB                    BDF SNEG
003263 14F7 F800                    LDI H'00
003264 14F9 30FD                    BR SSKP
003265 14FB F8FF            SNEG:   LDI H'FF
003266 14FD 19              SSKP:   INC R9
003267 14FE 59                      STR R9
003268 14FF 19                      INC R9
```

```
003269 1500 59                      STR R9
003270 1501 29                      DEC R9
003271 1502 DC                      SEP RC
003272 1503                         ;
003273 1503 86284C494E45A9          .DB H'86,"(LINE",H'A9 ; (LINE)
003274 150A 14EA                    .DW STOD - 7
003275 150C 05C2            PLINE:  .DW NEST
003276 150E 039D                    .DW GR
003277 1510 0086                    .DW LIT
003278 1512 0040                    .DW H'0040
003279 1514 0639                    .DW BBUF
003280 1516 11BC                    .DW SSMOD
003281 1518 03AD                    .DW RG
003282 151A 0645                    .DW BSCR
003283 151C 117D                    .DW STAR
003284 151E 03F7                    .DW PLUS
003285 1520 160A                    .DW BLOCK
003286 1522 03F7                    .DW PLUS
003287 1524 0086                    .DW LIT
003288 1526 0040                    .DW H'0040
003289 1528 0379                    .DW SEMIS
003290 152A                         ;
003291 152A 852E4C494EC5            .DB H'85,".LIN",H'C5 ; .LINE
003292 1530 1503                    .DW PLINE - 9
003293 1532 05C2            DLINE:  .DW NEST
003294 1534 150C                    .DW PLINE
003295 1536 0A35                    .DW TRLG
003296 1538 0A07                    .DW TYPE
003297 153A 0379                    .DW SEMIS
003298 153C                         ;
003299 153C 835553C5                .DB H'83,"US",H'C5  ; USE (ADDR OF
003300 1540 152A                    .DW DLINE - 8       ; NEXT BUFFER TO USE)
003301 1542 05CD            USE:    .DW VAR
003302 1544 4000                    .DW FIRSTB
003303 1546                         ;
003304 1546 84505245D6              .DB H'84,"PRE",H'D6 ; PREV   (ADDR OF
003305 154B 153C                    .DW USE - 6         ; PREVIOUSLY USED BUFFER)
003306 154D 05CD            PREV:   .DW VAR
003307 154F 4000                    .DW FIRSTB
003308 1551                         ;
003309 1551 842B4255C6              .DB H'84,"+BU",H'C6 ; +BUF
003310 1556 1546                    .DW PREV - 7        ; ADVANCE
003311 1558 05C2            PBUF:   .DW NEST            ;    BUFFER
003312 155A 0639                    .DW BBUF
003313 155C 0086                    .DW LIT
003314 155E 0004                    .DW H'0004
003315 1560 03F7                    .DW PLUS
003316 1562 03F7                    .DW PLUS
003317 1564 04B6                    .DW DUP
003318 1566 062D                    .DW LIMIT
003319 1568 07CD                    .DW EQL
003320 156A 00D0                    .DW ZBRCH
003321 156C 1572                    .DW PBUF1
003322 156E 048D                    .DW DROP
003323 1570 0621                    .DW FIRST
003324 1572 04B6            PBUF1:  .DW DUP
003325 1574 154D                    .DW PREV
```

```
003326 1576 04FA                    .DW AT
003327 1578 07C1                    .DW MINS
003328 157A 0379                    .DW SEMIS
003329 157C                         ;
003330 157C 865550444154C5          .DB H'86,"UPDAT",H'C5 ; UPDATE
003331 1583 1551                    .DW PBUF - 7
003332 1585 05C2            UPDAT:  .DW NEST
003333 1587 154D                    .DW PREV
003334 1589 04FA                    .DW AT
003335 158B 04FA                    .DW AT
003336 158D 0086                    .DW LIT
003337 158F 8000                    .DW H'8000
003338 1591 030B                    .DW FFOR
003339 1593 154D                    .DW PREV
003340 1595 04FA                    .DW AT
003341 1597 051E                    .DW EX
003342 1599 0379                    .DW SEMIS
003343 159B                         ;
003344 159B 8D454D5054592D42        .DB H'8D,"EMPTY-BUFFER",H'D3 ; EMPTY-BUFFER
       15A3 5546464552D3
003345 15A9 157C                    .DW UPDAT - 9
003346 15AB 05C2            MTBUF:  .DW NEST
003347 15AD 0621                    .DW FIRST
003348 15AF 062D                    .DW LIMIT
003349 15B1 0473                    .DW OVER
003350 15B3 07C1                    .DW MINS
003351 15B5 0B6C                    .DW ERS
003352 15B7 0379                    .DW SEMIS
003353 15B9                         ;
003354 15B9                         ;
003355 15B9 864255464645D2          .DB H'86,"BUFFE",H'D2 ; BUFFER
003356 15C0 159B                    .DW MTBUF - 16
003357 15C2 05C2            BUFFE:  .DW NEST
003358 15C4 1542                    .DW USE
003359 15C6 04FA                    .DW AT
003360 15C8 04B6                    .DW DUP
003361 15CA 039D                    .DW GR
003362 15CC 1558            BUFF1:  .DW PBUF
003363 15CE 00D0                    .DW ZBRCH
003364 15D0 15CC                    .DW BUFF1
003365 15D2 1542                    .DW USE
003366 15D4 051E                    .DW EX
003367 15D6 03BD                    .DW R
003368 15D8 04FA                    .DW AT
003369 15DA 03EB                    .DW ZLESS
003370 15DC 00D0                    .DW ZBRCH
003371 15DE 15F2                    .DW BUFF2
003372 15E0 03BD                    .DW R
003373 15E2 0775                    .DW PLUS2
003374 15E4 03BD                    .DW R
003375 15E6 04FA                    .DW AT
003376 15E8 0086                    .DW LIT
003377 15EA 7FFF                    .DW H'7FFF
003378 15EC 02F3                    .DW FAND
003379 15EE 05F2                    .DW ZERO
003380 15F0 1668                    .DW RSLW
003381 15F2 03BD            BUFF2:  .DW R
```

```
003382 15F4 051E                    .DW EX
003383 15F6 03BD                    .DW R
003384 15F8 154D                    .DW PREV
003385 15FA 051E                    .DW EX
003386 15FC 03AD                    .DW RG
003387 15FE 0775                    .DW PLUS2
003388 1600 0379                    .DW SEMIS
003389 1602                         ;
003390 1602 85424C4F43CB            .DB H'85,"BLOC",H'CB ; BLOCK
003391 1608 15B9                    .DW BUFFE - 9
003392 160A 05C2            BLOCK:  .DW NEST
003393 160C 06F2                    .DW OFST
003394 160E 04FA                    .DW AT
003395 1610 03F7                    .DW PLUS
003396 1612 039D                    .DW GR
003397 1614 154D                    .DW PREV
003398 1616 04FA                    .DW AT
003399 1618 04B6                    .DW DUP
003400 161A 04FA                    .DW AT
003401 161C 03BD                    .DW R
003402 161E 07C1                    .DW MINS
003403 1620 04B6                    .DW DUP
003404 1622 03F7                    .DW PLUS
003405 1624 00D0                    .DW ZBRCH
003406 1626 165A                    .DW BLOC1
003407 1628 1558            BLOC2:  .DW PBUF
003408 162A 03D2                    .DW ZEQAL
003409 162C 00D0                    .DW ZBRCH
003410 162E 1642                    .DW BLOC3
003411 1630 048D                    .DW DROP
003412 1632 03BD                    .DW R
003413 1634 15C2                    .DW BUFFE
003414 1636 04B6                    .DW DUP
003415 1638 03BD                    .DW R
003416 163A 05FA                    .DW ONE
003417 163C 1668                    .DW RSLW
003418 163E 0602                    .DW TWO
003419 1640 07C1                    .DW MINS
003420 1642 04B6            BLOC3:  .DW DUP
003421 1644 04FA                    .DW AT
003422 1646 03BD                    .DW R
003423 1648 07C1                    .DW MINS
003424 164A 04B6                    .DW DUP
003425 164C 03F7                    .DW PLUS
003426 164E 03D2                    .DW ZEQAL
003427 1650 00D0                    .DW ZBRCH
003428 1652 1628                    .DW BLOC2
003429 1654 04B6                    .DW DUP
003430 1656 154D                    .DW PREV
003431 1658 051E                    .DW EX
003432 165A 03AD            BLOC1:  .DW RG
003433 165C 048D                    .DW DROP
003434 165E 0775                    .DW PLUS2
003435 1660 0379                    .DW SEMIS
003436 1662                         ;
003437 1662 83522FD7                .DB H'83,H'52,H'2F,H'D7 ; R/W
003438 1666 1602                    .DW BLOCK - 8
```

```
003439 1668 05C2            RSLW:   .DW NEST
003440 166A 0499                    .DW SWAP
003441 166C 0086                    .DW LIT
003442 166E 00FA                    .DW H’00FA
003443 1670 118C                    .DW SLMOD
003444 1672 04B6                    .DW DUP
003445 1674 0086                    .DW LIT
003446 1676 0003                    .DW H’0003
003447 1678 07E5                    .DW GTR
003448 167A 0086                    .DW LIT
003449 167C 0005                    .DW H’0005
003450 167E 08BE                    .DW QERR
003451 1680 0499                    .DW SWAP
003452 1682 0086                    .DW LIT
003453 1684 0008                    .DW H’0008
003454 1686 117D                    .DW STAR
003455 1688 0086                    .DW LIT
003456 168A 0001                    .DW H’0001
003457 168C 03F7                    .DW PLUS
003458 168E 0086                    .DW LIT
003459 1690 001A                    .DW H’001A
003460 1692 118C                    .DW SLMOD
003461 1694 075F                    .DW DV
003462 1696 0535                    .DW CEX
003463 1698 05FA                    .DW ONE
003464 169A 07C1                    .DW MINS
003465 169C 0499                    .DW SWAP
003466 169E 0086                    .DW LIT
003467 16A0 0040                    .DW H’0040
003468 16A2 117D                    .DW STAR
003469 16A4 03F7                    .DW PLUS
003470 16A6 075F                    .DW DV
003471 16A8 0768                    .DW PLUS1
003472 16AA 0535                    .DW CEX
003473 16AC 05F2                    .DW ZERO
003474 16AE 075F                    .DW DV
003475 16B0 0775                    .DW PLUS2
003476 16B2 0535                    .DW CEX
003477 16B4 075F                    .DW DV
003478 16B6 0639                    .DW BBUF
003479 16B8 07F3                    .DW ROT
003480 16BA 00D0                    .DW ZBRCH
003481 16BC 16C4                    .DW RWELSE
003482 16BE 16D5                    .DW BLKRD
003483 16C0 00BD                    .DW BRCH
003484 16C2 16C6                    .DW RWEND
003485 16C4 171F            RWELSE: .DW BLKWT
003486 16C6 0379            RWEND:  .DW SEMIS
003487 16C8 0A424C4F434B2D52        .DB H’0A,"BLOCK-REA",H’C4 ; BLOCKREAD
       16D0 4541C4
003488 16D3 1662                    .DW RSLW - 6
003489 16D5 16D7            BLKRD:  .DW * + 2
003490 16D7 F883                    LDI H’83
003491 16D9 B4                      PHI R4
003492 16DA B5                      PHI R5
003493 16DB F864                    LDI H’64
003494 16DD A4                      PLO R4
```

```
003495 16DE F874                    LDI H’74
003496 16E0 A5                      PLO R5
003497 16E1                         ;
003498 16E1 E2                      SEX R2
003499 16E2 9C                      GHI RC
003500 16E3 73                      STXD
003501 16E4 8C                      GLO RC
003502 16E5 73                      STXD
003503 16E6 49                      LDA R9
003504 16E7 B7                      PHI R7
003505 16E8 09                      LDN R9
003506 16E9 A7                      PLO R7
003507 16EA 29                      DEC R9
003508 16EB 29                      DEC R9
003509 16EC 09                      LDN R9
003510 16ED AC                      PLO RC
003511 16EE 29                      DEC R9
003512 16EF 09                      LDN R9
003513 16F0 BC                      PHI RC
003514 16F1 1C                      INC RC
003515 16F2 1C                      INC RC
003516 16F3 29                      DEC R9
003517 16F4 09                      LDN R9
003518 16F5 A8                      PLO R8
003519 16F6 29                      DEC R9
003520 16F7 09                      LDN R9
003521 16F8 B8                      PHI R8
003522 16F9 29                      DEC R9
003523 16FA 29                      DEC R9
003524 16FB D4              BLKRD2: SEP R4
003525 16FC 8502                    .DW H’8502
003526 16FE 9F                      GHI RF
003527 16FF 58                      STR R8
003528 1700 18                      INC R8
003529 1701 27                      DEC R7
003530 1702 97                      GHI R7
003531 1703 CA16FB                  LBNZ BLKRD2
003532 1706 87                      GLO R7
003533 1707 CA16FB                  LBNZ BLKRD2
003534 170A                         ;
003535 170A E2                      SEX R2
003536 170B 60                      IRX
003537 170C 72                      LDXA
003538 170D AC                      PLO RC
003539 170E F0                      LDX
003540 170F BC                      PHI RC
003541 1710 DC                      SEP RC
003542 1711                         ;
003543 1711 0B424C4F434B2D57        .DB H’0B,"BLOCK-WRIT",H’C5 ; BLOCK-WRITE
       1719 524954C5
003544 171D 16C8                    .DW BLKRD - 13
003545 171F 1721            BLKWT:  .DW * + 2
003546 1721 F883                    LDI H’83
003547 1723 B4                      PHI R4
003548 1724 B5                      PHI R5
003549 1725 F864                    LDI H’64
003550 1727 A4                      PLO R4
```

```
003551 1728 F874                    LDI H’74
003552 172A A5                      PLO R5
003553 172B                         ;
003554 172B E2                      SEX R2
003555 172C 9C                      GHI RC
003556 172D 73                      STXD
003557 172E 8C                      GLO RC
003558 172F 73                      STXD
003559 1730 49                      LDA R9
003560 1731 B7                      PHI R7
003561 1732 09                      LDN R9
003562 1733 A7                      PLO R7
003563 1734 29                      DEC R9
003564 1735 29                      DEC R9
003565 1736 09                      LDN R9
003566 1737 AC                      PLO RC
003567 1738 29                      DEC R9
003568 1739 09                      LDN R9
003569 173A BC                      PHI RC
003570 173B 1C                      INC RC
003571 173C 1C                      INC RC
003572 173D 29                      DEC R9
003573 173E 09                      LDN R9
003574 173F A8                      PLO R8
003575 1740 29                      DEC R9
003576 1741 09                      LDN R9
003577 1742 B8                      PHI R8
003578 1743 29                      DEC R9
003579 1744 29                      DEC R9
003580 1745 48              BLKWT2: LDA R8
003581 1746 BF                      PHI RF
003582 1747 D4                      SEP R4
003583 1748 8500                    .DW H’8500
003584 174A 27                      DEC R7
003585 174B 97                      GHI R7
003586 174C CA1745                  LBNZ BLKWT2
003587 174F 87                      GLO R7
003588 1750 CA1745                  LBNZ BLKWT2
003589 1753                         ;
003590 1753 E2                      SEX R2
003591 1754 60                      IRX
003592 1755 72                      LDXA
003593 1756 AC                      PLO RC
003594 1757 F0                      LDX
003595 1758 BC                      PHI RC
003596 1759 DC                      SEP RC
003597 175A                         ;
003598 175A 844C4F41C4              .DB H’84,"LOA",H’C4 ; LOAD
003599 175F 1711                    .DW BLKWT - 14
003600 1761 05C2            LOAD:   .DW NEST
003601 1763 06C8                    .DW BLK
003602 1765 04FA                    .DW AT
003603 1767 039D                    .DW GR
003604 1769 06D1                    .DW FIN
003605 176B 04FA                    .DW AT
003606 176D 039D                    .DW GR
003607 176F 05F2                    .DW ZERO
```

```
003608 1771 06D1                    .DW FIN
003609 1773 051E                    .DW EX
003610 1775 0645                    .DW BSCR
003611 1777 117D                    .DW STAR
003612 1779 06C8                    .DW BLK
003613 177B 051E                    .DW EX
003614 177D 0EE1                    .DW INPT
003615 177F 03AD                    .DW RG
003616 1781 06D1                    .DW FIN
003617 1783 051E                    .DW EX
003618 1785 03AD                    .DW RG
003619 1787 06C8                    .DW BLK
003620 1789 051E                    .DW EX
003621 178B 0379                    .DW SEMIS
003622 178D                         ;
003623 178D C32D2DBE                .DB H'C3,"--",H'BE ; -->
003624 1791 175A                    .DW LOAD - 7
003625 1793 05C2            ARROW:  .DW NEST
003626 1795 0937                    .DW QLDG
003627 1797 05F2                    .DW ZERO
003628 1799 06D1                    .DW FIN
003629 179B 051E                    .DW EX
003630 179D 0645                    .DW BSCR
003631 179F 06C8                    .DW BLK
003632 17A1 04FA                    .DW AT
003633 17A3 0473                    .DW OVER
003634 17A5 11AC                    .DW MODD
003635 17A7 07C1                    .DW MINS
003636 17A9 06C8                    .DW BLK
003637 17AB 04C6                    .DW PLUSS
003638 17AD 0379                    .DW SEMIS
003639 17AF                         ;
003640 17AF                         ;
003641 17AF 834452B0                .DB H'83,H'44,H'52,H'B0 ; DR0
003642 17B3 178D                    .DW ARROW - 6
003643 17B5 05C2            DRZER:  .DW NEST
003644 17B7 05F2                    .DW ZERO
003645 17B9 06F2                    .DW OFST
003646 17BB 051E                    .DW EX
003647 17BD 0379                    .DW SEMIS
003648 17BF                         ;
003649 17BF                         ;
003650 17BF 834452B1                .DB H'83,H'44,H'52,H'B1 ; DR1
003651 17C3 17AF                    .DW DRZER - 6
003652 17C5 05C2            DRONE:  .DW NEST
003653 17C7 0645                    .DW BSCR
003654 17C9 0086                    .DW LIT             ; 250 SCREENS/DISK
003655 17CB 00FA                    .DW H'00FA
003656 17CD 117D                    .DW STAR
003657 17CF 06F2                    .DW OFST
003658 17D1 051E                    .DW EX
003659 17D3 0379                    .DW SEMIS
003660 17D5                         ;
003661 17D5                         ;
003662 17D5 844C4953D4              .DB H'84,"LIS",H'D4 ; LIST
003663 17DA 17BF                    .DW DRONE - 6
003664 17DC 05C2            LIST:   .DW NEST
```

```
003665 17DE 09B2                    .DW MDCML
003666 17E0 05AB                    .DW CR
003667 17E2 04B6                    .DW DUP
003668 17E4 06E5                    .DW FSCR
003669 17E6 051E                    .DW EX
003670 17E8 0A68                    .DW PDQ
003671 17EA 06534352202320          .DB H'06,"SCR # "
003672 17F1 1415                    .DW DOT
003673 17F3 0086                    .DW LIT
003674 17F5 0010                    .DW H'0010
003675 17F7 05F2                    .DW ZERO
003676 17F9 0150                    .DW PDO
003677 17FB 05AB            LIST1:  .DW CR
003678 17FD 14C4                    .DW I
003679 17FF 0086                    .DW LIT
003680 1801 0003                    .DW H'0003
003681 1803 13F6                    .DW DOTR
003682 1805 0807                    .DW SPC
003683 1807 14C4                    .DW I
003684 1809 06E5                    .DW FSCR
003685 180B 04FA                    .DW AT
003686 180D 1532                    .DW DLINE
003687 180F 0597                    .DW QTERM
003688 1811 00D0                    .DW ZBRCH
003689 1813 1817                    .DW LIST2
003690 1815 0389                    .DW LVE
003691 1817 00EC            LIST2:  .DW LUPE
003692 1819 17FB                    .DW LIST1
003693 181B 05AB                    .DW CR
003694 181D 0379                    .DW SEMIS
003695 181F                         ;
003696 181F                         ;
003697 181F 85494E4445D8            .DB H'85,"INDE",H'D8 ; INDEX
003698 1825 17D5                    .DW LIST - 7
003699 1827 05C2            INDEX:  .DW NEST
003700 1829 05AB                    .DW CR
003701 182B 0768                    .DW PLUS1
003702 182D 0499                    .DW SWAP
003703 182F 0150                    .DW PDO
003704 1831 05AB            INDE1:  .DW CR
003705 1833 14C4                    .DW I
003706 1835 0086                    .DW LIT
003707 1837 0003                    .DW H'0003
003708 1839 13F6                    .DW DOTR
003709 183B 0807                    .DW SPC
003710 183D 05F2                    .DW ZERO
003711 183F 14C4                    .DW I
003712 1841 1532                    .DW DLINE
003713 1843 0597                    .DW QTERM
003714 1845 00D0                    .DW ZBRCH
003715 1847 184B                    .DW INDE2
003716 1849 0389                    .DW LVE
003717 184B 00EC            INDE2:  .DW LUPE
003718 184D 1831                    .DW INDE1
003719 184F 0379                    .DW SEMIS
003720 1851                         ;
003721 1851 8554524941C4            .DB H'85,"TRIA",H'C4 ; TRIAD
```

```
003722 1857 181F                    .DW INDEX - 8
003723 1859 05C2            TRIAD:  .DW NEST
003724 185B 05AB                    .DW CR
003725 185D 0086                    .DW LIT
003726 185F 0003                    .DW H'0003
003727 1861 119C                    .DW SLASH
003728 1863 0086                    .DW LIT
003729 1865 0003                    .DW H'0003
003730 1867 117D                    .DW STAR
003731 1869 0086                    .DW LIT
003732 186B 0003                    .DW H'0003
003733 186D 0473                    .DW OVER
003734 186F 03F7                    .DW PLUS
003735 1871 0499                    .DW SWAP
003736 1873 0150                    .DW PDO
003737 1875 05AB            TRIA1:  .DW CR
003738 1877 14C4                    .DW I
003739 1879 17DC                    .DW LIST
003740 187B 0597                    .DW QTERM
003741 187D 00D0                    .DW ZBRCH
003742 187F 1883                    .DW TRIA2
003743 1881 0389                    .DW LVE
003744 1883 00EC            TRIA2:  .DW LUPE
003745 1885 1875                    .DW TRIA1
003746 1887 05AB                    .DW CR
003747 1889 0086                    .DW LIT
003748 188B 000F                    .DW H'000F
003749 188D 148C                    .DW MSG
003750 188F 05AB                    .DW CR
003751 1891 0379                    .DW SEMIS
003752 1893                         ;
003753 1893                         ;
003754 1893 85464C5553C8            .DB H'85,"FLUS",H'C8 ; FLUSH
003755 1899 1851                    .DW TRIAD - 8
003756 189B 05C2            FLUSH:  .DW NEST
003757 189D 062D                    .DW LIMIT
003758 189F 0621                    .DW FIRST
003759 18A1 07C1                    .DW MINS
003760 18A3 0639                    .DW BBUF
003761 18A5 0086                    .DW LIT
003762 18A7 0004                    .DW H'0004
003763 18A9 03F7                    .DW PLUS
003764 18AB 119C                    .DW SLASH
003765 18AD 05F2                    .DW ZERO
003766 18AF 0150                    .DW PDO
003767 18B1 0086            FL1:    .DW LIT
003768 18B3 7FFF                    .DW H'7FFF
003769 18B5 15C2                    .DW BUFFE
003770 18B7 048D                    .DW DROP
003771 18B9 00EC                    .DW LUPE
003772 18BB 18B1                    .DW FL1
003773 18BD 0379                    .DW SEMIS
003774 18BF                         ;
003775 18BF                         ;
003776 18BF 84544153CB              .DB H'84,"TAS",H'CB ; TASK
003777 18C4 1893                    .DW FLUSH - 8
003778 18C6 05C2            TASK:   .DW NEST
```

```
003779 18C8 0379                    .DW SEMIS
003780 18CA                         ;
003781 18CA                         ;
003782 1900                         .PAGE
003783 1900                         ;
003784 1900                         ;
003785 1900 F800            COLD:   LDI (START + 12) >> 8
003786 1902 B7                      PHI R7
003787 1903 F86A                    LDI (START + 12)
003788 1905 A7                      PLO R7
003789 1906 F80F                    LDI (FRTH + 14) >> 8
003790 1908 B8                      PHI R8
003791 1909 F868                    LDI ((FRTH + 14) & H'00FF)
003792 190B A8                      PLO R8
003793 190C 47                      LDA R7
003794 190D 58                      STR R8
003795 190E 18                      INC R8
003796 190F 47                      LDA R7
003797 1910 58                      STR R8
003798 1911 F816                    LDI H'16
003799 1913 3017                    BR PUTF
003800 1915 F810            WARM:   LDI H'10
003801 1917 AF              PUTF:   PLO RF
003802 1918 F800                    LDI (START + H'10) >> 8
003803 191A B7                      PHI R7
003804 191B F86E                    LDI (START + H'10)
003805 191D A7                      PLO R7
003806 191E 47                      LDA R7
003807 191F BD                      PHI RD
003808 1920 B8                      PHI R8
003809 1921 07                      LDN R7
003810 1922 AD                      PLO RD
003811 1923 A8                      PLO R8
003812 1924 F86A                    LDI (START + 12)
003813 1926 A7                      PLO R7
003814 1927 47              WRMLP:  LDA R7
003815 1928 58                      STR R8
003816 1929 18                      INC R8
003817 192A 2F                      DEC RF
003818 192B 8F                      GLO RF
003819 192C 3A27                    BNZ WRMLP
003820 192E F800                    LDI NEXT >> 8
003821 1930 BC                      PHI RC
003822 1931 F892                    LDI NEXT
003823 1933 AC                      PLO RC
003824 1934 F8C0                    LDI ((ABORT + 2) & H'00FF)
003825 1936 AA                      PLO RA
003826 1937 F80F                    LDI (ABORT + 2) >> 8
003827 1939 BA                      PHI RA
003828 193A C00367                  LBR (RP1 + 2)
003829 193D                         ;
003830 193D C4              LEND:   NOP                 ; INITIAL FENCE IS HERE
003831 193E                         ;
003832 193E                         ; TO EXTEND THIS PORTION TO INCLUDE
003833 193E                         ; NEW WORDS, FIRST USE
003834 193E                         ;    HERE.  TO FIND END OF YOUR NEW VERSION
003835 193E                         ; THEN USE THE FOLLOWING:
```

```
003836 193E                          ;
003837 193E                          ;    LATEST 12 +ORIGIN !
003838 193E                          ;    HERE  28 +ORIGIN !
003839 193E                          ;    HERE  30 +ORIGIN !
003840 193E                          ;    HERE FENCE !
003841 193E                          ;
003842 193E                          ; THEN USE   BYE TO GET BACK TO YOUR MONITOR
003843 193E                          ; THEN USE YOUR MONITOR ROUTINES TO SAVE MEMORY
003844 193E                          ; FROM 0000 TO END ADDRESS
003845 193E                          ;
003846 193E                          ; THIS PROCEDURE WILL ALLOW YOU TO CONTINUE
003847 193E                          ; BUILDING ON YOUR FORTH VOCABULARY WITHOUT THE
003848 193E                          ; DISC INTERFACE
003849 193E                          ;
003850 193E                          ;
003851 193E                          ;
003852 193E                          ;
003853 193E                          ;
003854 193E                          .END
ABORT   =0FBE  DCSP    =08AB  FOUT    =06DB  MSLAS   =1157  R2      =0002  TYP1    =0A17
ABS     =1106  DDOT    =1407  FRST    =0215  MSMOD   =11DF  R3      =0003  TYPE    =0A07
AGAIN   =12BB  DDOTR   =13D3  FRTH    =0F5A  MSTAR   =113A  R4      =0004  UDOT    =142E
ALLOT   =0794  DELIM   =0238  FSCR    =06E5  MTBUF   =15AB  R5      =0005  UNTIL   =1299
ARROW   =1793  DFN     =0F7A  FSPAT   =033D  NEST    =05C2  R6      =0006  UOUT    =02A5
AT      =04FA  DGT     =016B  FXOR    =0324  NEXCHR  =01AD  R7      =0007  UPDAT   =1585
BACK    =1218  DIG     =1390  GR      =039D  NEXT    =0092  R8      =0008  USE     =1542
BAD     =0188  DIG1    =13AC  GTR     =07E5  NFA     =0880  R9      =0009  USER    =05DF
BAD2    =0186  DIGS    =13BB  HERE    =0784  NMB1    =0C71  RA      =000A  USLSH   =02B2
BADCHR  =01D4  DIGS1   =13BD  HLD     =0756  NMB2    =0C97  RB      =000B  USR     =0E2F
BADLEN  =01D3  DLINE   =1532  HOLD    =0B8C  NMB3    =0CA1  RBK     =0975  USTAR   =0278
BASE    =0725  DLTL    =0E98  I       =14C4  NMBR    =0C53  RC      =000C  VAR     =05CD
BBUF    =0639  DLTL1   =0EA8  ID      =0D2D  NO      =00DD  RD      =000D  VARB    =0E1E
BCOMP   =104A  DMIN    =0458  IFF     =12EB  NONE    =03DE  RE      =000E  VB1     =0F52
BDIGS   =1352  DO      =125A  IMMED   =1068  NULL    =022A  REPEA   =12D2  VBLY    =0F32
BEGIN   =1226  DOK     =017C  INDE1   =1831  OFST    =06F2  RF      =000F  VL      =06BE
BL      =060B  DOSEG   =0E53  INDE2   =184B  ONE     =05FA  RG      =03AD  VLIS1   =1450
BLK     =06C8  DOT     =1415  INDEX   =1827  ORGN    =0652  RNU     =074C  VLIS2   =1464
BLKRD   =16D5  DOTQ    =1014  INPT    =0EE1  OVER    =0473  RO      =0676  VLIST   =143E
BLKRD2  =16FB  DOTQ1   =1034  KEY     =0578  PABRT   =0CD7  ROT     =07F3  WARM    =1915
BLKWT   =171F  DOTQ2   =103C  KEY1    =057E  PAD     =0BA4  RP1     =0365  WBR     =0096
BLKWT2  =1745  DOTR    =13F6  LB      =0967  PAREN   =1078  RSLW    =1668  WD1     =0BCB
BLNK    =0B7D  DP      =06AF  LBLD    =0E43  PBUF    =1558  RSTACK  =0002  WD2     =0BCF
BLOC1   =165A  DPL     =072F  LEND    =193D  PBUF1   =1572  RWELSE  =16C4  WHILE   =1324
BLOC2   =1628  DPLUS   =042A  LESS    =07D9  PCODE   =09C8  RWEND   =16C6  WIDTH   =068C
BLOC3   =1642  DPM     =10F4  LFA     =0862  PDO     =0150  SEMIC   =1001  WORD    =0BB7
BLOCK   =160A  DPM1    =10FE  LIMIT   =062D  PDQ     =0A68  SEMIS   =0379  WRM     =14D9
BOK     =01E7  DRONE   =17C5  LIMITB  =6C2C  PFA     =0896  SIGN    =137A  WRMLP   =1927
BRANCH  =00BF  DROP    =048D  LIST    =17DC  PLINE   =150C  SIGN1   =138A  WRNG    =069A
BRCH    =00BD  DRZER   =17B5  LIST1   =17FB  PLOOP   =1283  SKIP    =0232  X       =0B0F
BSCR    =0645  DUP     =04B6  LIST2   =1817  PLUPE   =0124  SKP9A   =02A0  X1      =0B3B
BUFF1   =15CC  DUZ1    =0E5F  LIT     =0086  PLUS    =03F7  SKPD8   =02DE  X2      =0B3F
BUFF2   =15F2  DV      =075F  LOAD    =1761  PLUS1   =0768  SLASH   =119C  XEND    =0B43
BUFFE   =15C2  EDIGS   =1361  LOOP    =126D  PLUS2   =0775  SLMOD   =118C  ZBRCH   =00D0
BYE     =120B  ELSEE   =1302  LOOP1   =01A4  PLUSS   =04C6  SMDG    =098A  ZEQAL   =03D2
CAT     =050D  EMIT    =054A  LOOP2   =01B6  PM      =10E2  SNEG    =14FB  ZERO    =05F2
CCMA    =07B1  ENCL    =01FE  LOP1    =020D  PM1     =10EC  SO      =066D  ZLESS   =03EB
CEND    =0113  END2    =026E  LOP2    =0220  PNM1    =0C40  SP1     =0350  ZONE    =03DA
```

```
CEX     =0535  ENDD    =12AD  LP7B    =0281  PNM2    =0C46  SPACS   =1335
CFA     =0872  ENDIFF  =1238  LPC5    =02CB  PNMBR   =0C08  SPAX1   =134B
CL      =0615  EQL     =07CD  LT1     =0E8B  POP     =04D7  SPAX2   =1345
CLD     =14E5  ERR1    =0CF3  LTL     =0E7B  PORGN   =0660  SPC     =0807
CMOVE   =0246  ERROR   =0CE5  LTST    =0852  PREV    =154D  SSKP    =14FD
CMPL    =0951  ERS     =0B6C  LUPE    =00EC  PT1     =0F05  SSLA    =11CD
CNST    =0E09  EX      =051E  LUPE1   =0140  PT2     =0EFB  SSMOD   =11BC
CNT     =09F4  EXC     =08F0  LUUP    =0265  PT3     =0EFF  STAR    =117D
CNTX    =0700  EXE     =00A6  LVE     =0389  PT4     =0F1F  START   =005E
CODE    =09DE  EXPT    =0A85  MAX     =1123  PT5     =0F19  STOD    =14F1
COLD    =1900  EXPT1   =0ABF  MAX1    =1131  PT6     =0F1D  STOR    =03E0
COLON   =0DC4  EXPT2   =0AE5  MDCML   =09B2  PUTF    =1917  STT     =071A
COMMA   =07A0  EXPT3   =0AD7  MDUP    =0816  Q1      =0FB0  SWAP    =0499
COMP    =0100  EXPT4   =0A8F  MESS1   =14B2  Q2      =0F97  TASK    =18C6
CONST   =05D6  EXPT5   =0AD9  MESS2   =14AE  QCMP    =08D8  TGLE    =04E3
CR      =05AB  FAND    =02F3  MESS3   =14BE  QCSP    =091A  THEN    =124F
CR1     =05AE  FFOR    =030B  MF1     =0CCB  QERR    =08BE  THREE   =1086
CR2     =05B7  FILL    =0B4C  MFIND   =0CAB  QLDG    =0937  TIB     =0680
CRNT    =070E  FIN     =06D1  MHEX    =099C  QPR     =0907  TICK    =108E
CRT1    =0D8A  FIND    =0196  MIN     =0D15  QSTK    =0EB3  TR1     =0831
CRTE    =0D64  FIRST   =0621  MINOS   =0416  QTERM   =0597  TRIA1   =1875
CSEND   =0552  FIRSTB  =4000  MINS    =07C1  QUER    =0AF7  TRIA2   =1883
CSEND1  =0565  FL1     =18B1  MINUS   =0412  QUES    =1421  TRIAD   =1859
CSP     =0743  FLD     =0739  MN1     =0D23  QUIT    =0F8D  TRL1    =0A3D
CSTACK  =0009  FLUSH   =189B  MODD    =11AC  R       =03BD  TRLG    =0A35
DABS    =1115  FNCE    =06A6  MON     =11FB  R0      =0000  TRVS    =082D
DCODE   =0DEC  FORG    =10A7  MSG     =148C  R1      =0001  TWO     =0602

003293 1532 05C2            DLINE:  .DW NEST
003294 1534 150C                    .DW PLINE
003295 1536 0A35
```

8.2. Rc/Forth

Popisuji implementace Rc/Forth ROM version 0.1.

Forth je uložen v paměti ROM od adresy `0e000h` výše.

```
xxx
```

FIXME:

```
0000:                       org     0e000h
e000: f8 e0                 ldi     high start          ; get address of main start
e002: b3                    phi     r3                  ; and place into standard PC
e003: f8 07                 ldi     low start
e005: a3                    plo     r3
e006: d3                    sep     r3                  ; transfer control to main PC
```

FIXME:

```
e007: e2          start:    sex     r2                ; set X to stack
e008: f8 ff                 ldi     high call         ; get address of standard call
e00a: b4                    phi     r4                ; place into r4
e00b: f8 e0                 ldi     low call
e00d: a4                    plo     r4
```

```
e00e: f8 ff               ldi     high ret          ; get address of standard return
e010: b5                  phi     r5
e011: f8 f1               ldi     low ret
e013: a5                  plo     r5
e014: f8 00               ldi     0                 ; get temporary stack
e016: b2                  phi     r2
e017: f8 ff               ldi     0ffh
e019: a2                  plo     r2
e01a: d4                  sep     scall             ; copy definitions to low memory
e01b: ea da               dw      copy
e01d: f8 50               ldi     50h
e01f: b2                  phi     r2
e020: f8 00               ldi     0
e022: a2                  plo     r2
e023: d4                  sep     scall             ; call bios to set terminal speed
e024: ff 2d               dw      f_setbd           ; function to set terminal
e026: f8 01               ldi     high himem        ; get page of data segment
e028: b9                  phi     r9                ; place into r9
e029:
e029: f8 eb               ldi     high hello        ; address of signon message
e02b: bf                  phi     rf                ; place into r6
e02c: f8 c1               ldi     low hello
e02e: af                  plo     rf
e02f: d4                  sep     scall             ; call bios to display message
e030: ff 09               dw      f_msg             ; function to display a message
```

FIXME:

```
e032:             ; ***********************************************
e032:             ; **** Determine how much memory is installed ****
e032:             ; ***********************************************
e032: f8 06               ldi     low freemem       ; free memory pointer
  :
e077: 59                  str     r9                ; and store
```

Hlavní smyčka interpretu.

```
e078:             ; ************************
e078:             ; *** Main program loop ***
e078:             ; ************************
e078: f8 eb       mainlp:     ldi     high prompt       ; address of prompt
e07a: bf                      phi     rf                ; place into r6
e07b: f8 d0                   ldi     low prompt
e07d: af                      plo     rf
e07e: d4                      sep     scall             ; display prompt
e07f: ff 09                   dw      f_msg             ; function to display a message
e081: f8 00                   ldi     high buffer       ; point to input buffer
e083: bf                      phi     rf
e084: f8 00                   ldi     low buffer
e086: af                      plo     rf
e087: d4                      sep     scall             ; read a line
e088: ff 0f                   dw      f_input           ; function to read a line
e08a: f8 eb                   ldi     high crlf         ; address of CR/LF
e08c: bf                      phi     rf                ; place into r6
e08d: f8 cd                   ldi     low crlf
e08f: af                      plo     rf
e090: d4                      sep     scall             ; call bios
e091: ff 09                   dw      f_msg             ; function to display a message
e093: d4                      sep     scall             ; call tokenizer
e094: e1 ea                   dw      tknizer
e096:
e096: f8 06                   ldi     low freemem       ; get free memory pointer
e098: a9                      plo     r9                ; place into data segment
e099: 49                      lda     r9                ; get free memory pointer
e09a: bb                      phi     rb                ; place into rF
e09b: 09                      ldn     r9
e09c: ab                      plo     rb
e09d: 1b                      inc     rb
e09e: 1b                      inc     rb
e09f: d4                      sep     scall
```

```
e0a0: e3 65                 dw      exec
e0a2:
e0a2: 30 78                 br      mainlp              ; return to beginning of main loop
```

FIXME:

```
e365:             ; ***************************************************
e365:             ; *** Execute forth byte codes, RB points to codes ***
e365:             ; ***************************************************
e365: 0b          exec:     ldn     rb                  ; get byte from codestream
e366: 32 f2                 bz      execdn              ; jump if at end of stream
e368: ff ff                 smi     T_NUM               ; check for numbers
e36a: 32 aa                 bz      execnum             ; code is numeric
e36c: 0b                    ldn     rb                  ; recover byte
e36d: ff fe                 smi     T_ASCII             ; check for ascii data
e36f: 32 bc                 bz      execascii           ; jump if ascii
e371: f8 01                 ldi     high jump
e373: b8                    phi     r8
e374: f8 0a                 ldi     low jump
e376: a8                    plo     r8
e377: 18                    inc     r8
e378: 0b                    ldn     rb                  ; recover byte
e379: fa 7f                 ani     07fh                ; strip high bit
e37b: ff 01                 smi     1                   ; reset to origin
e37d: fe                    shl                         ; addresses are two bytes
e37e: e2                    sex     r2                  ; point X to stack
e37f: 52                    str     r2                  ; write offset for addtion
e380: f8 83                 ldi     low cmdvecs
e382: f4                    add                         ; add offset
e383: a7                    plo     r7
e384: f8 ec                 ldi     high cmdvecs        ; high address of command vectors
e386: 7c 00                 adci    0                   ; propagate carry
e388: b7                    phi     r7                  ; r[7] now points to command vector
e389: 47                    lda     r7                  ; get high byte of vector
e38a: 58                    str     r8
e38b: 18                    inc     r8
e38c: 47                    lda     r7                  ; get low byte of vector
e38d: 58                    str     r8
e38e: 1b                    inc     rb                  ; point r6 to next command
e38f: 8b                    glo     rb                  ; save RF
e390: 73                    stxd
e391: 9b                    ghi     rb
e392: 73                    stxd
e393: c0 01 0a              lbr     jump                ; jump vector

e396: e2          execret:  sex     r2                  ; be sure X poits to stack
e397: a7                    plo     r7                  ; save return code
e398: 60                    irx                         ; recover RF
e399: 42                    lda     r2
e39a: bb                    phi     rb
e39b: 02                    ldn     r2
e39c: ab                    plo     rb
e39d: 87                    glo     r7                  ; get result code
e39e: 32 65                 bz      exec                ; jump if no error
e3a0: f8 eb                 ldi     high msempty        ; get error message
e3a2: bf                    phi     rf
e3a3: f8 d4                 ldi     low msempty
e3a5: af                    plo     rf
e3a6: d4          execrmsg: sep     scall
e3a7: ff 09                 dw      f_msg
e3a9: d5                    sep     sret                ; return to caller
e3aa:
e3aa: 1b          execnum:  inc     rb                  ; point to number
e3ab: 9b                    ghi     rb
e3ac: b7                    phi     r7
e3ad: 8b                    glo     rb
e3ae: a7                    plo     r7
e3af: 47                    lda     r7
e3b0: bb                    phi     rb
```

```
e3b1: 47                    lda     r7
e3b2: ab                    plo     rb
e3b3: d4                    sep     scall
e3b4: e0 d8                 dw      push
e3b6: 97                    ghi     r7
e3b7: bb                    phi     rb
e3b8: 87                    glo     r7
e3b9: ab                    plo     rb
e3ba: 30 65                 br      exec                ; execute next code

e3bc: 1b          execascii: inc     rb                ; move past ascii code
e3bd: 9b                    ghi     rb                ; transfer name to R8
e3be: b8                    phi     r8
e3bf: 8b                    glo     rb
e3c0: a8                    plo     r8
e3c1: d4                    sep     scall             ; find entry
e3c2: e1 1c                 dw      findname
e3c4: 3b ce                 bnf     ascnoerr          ; jump if name was found
e3c6: f8 eb       ascerr:   ldi     high msgerr       ; get error message
e3c8: bf                    phi     rf
e3c9: f8 e2                 ldi     low msgerr
e3cb: af                    plo     rf
e3cc: 30 a6                 br      execrmsg
e3ce: 17          ascnoerr: inc     r7                ; point to type
e3cf: 17                    inc     r7
e3d0: 07                    ldn     r7                ; get type
e3d1: ff 86                 smi     86h               ; check for variable
e3d3: 32 e9                 bz      execvar           ; jump if so
e3d5: 07                    ldn     r7                ; get type
e3d6: ff 87                 smi     87h               ; check for function
e3d8: 3a c6                 bnz     ascerr            ; jump if not
e3da: e2                    sex     r2                ; be sure X is pointing to stack
e3db: 88                    glo     r8                ; save position
e3dc: 73                    stxd                      ; and store on stack
e3dd: 98                    ghi     r8
e3de: 73                    stxd
e3df: d4                    sep     scall             ; call exec to execute stored program
e3e0: e3 65                 dw      exec
e3e2: 60                    irx                       ; recover pointer
e3e3: 72                    ldxa
e3e4: bb                    phi     rb
e3e5: f0                    ldx
e3e6: ab                    plo     rb
e3e7: 30 65                 br      exec              ; and continue execution
e3e9: d4          execvar:  sep     scall             ; push var address to stack
e3ea: e0 d8                 dw      push
e3ec: 98                    ghi     r8                ; transfer address back to rb
e3ed: bb                    phi     rb
e3ee: 88                    glo     r8
e3ef: ab                    plo     rb
e3f0: 30 65                 br      exec              ; execute next code
e3f2:
e3f2: d5          execdn:   sep     sret              ; return to caller
```

8.3. Cosmac 1802 handheld computer

Obsazení registrů

- R3 — parameter stack (the X register is setup with **SEX R3** instruction
- R4 — return stack
- R5 — i (interpreter pointer)

- R6 — w (current word pointer)
- R8 — address of next, the inner interpreter
- R9 — program counter

Code definitions: Forth words defined in assembly code should have the contents of their code-field address (CFA) pointing to their parameter-field address (PFA); the definition should terminate with the **SEP 8** instruction.

Inner interpretermechanism, or 'next': code words control to next by setting R8 as the program counter (**SEP R8**). When the inner interpreter has advanced the I pointer and set the W pointer to the PFA of the next word to be executed then R9 is set as the program counter to execute the code associated with that word and R8 will be reset to the start address of NEXT again.

Stack requirements: stack space allocation is dictated by the application processing and nesting requirements. If memory is restricted try some low nominal values, eg. parameter stack space >= 40h bytes ? return stack space >= 40h bytes ?

Header structure: the word header structure is based on the Forth-79 dictionary model:

- byte 1 — length L of the word name (msb set) [NFA]
- byte 2 — name bytes (final byte indicated by msb set)
- byte L+2 — 2 byte link to NFA of previous named word in dictinary [LFA]
- byte L+4 — 2 byte pointer to code definition for this word [CFA]
- byte L+6 ... — threaded list of word addresses or parameters for definition [PFA]
-

Kapitola 9. Forth na procesoru 6502

* *chapter id="implementace.6502" xreflabel="Forth na procesoru 6502"*
* *rcsinfo="$Header: /home/radek/cvs/forth-book/ch-6502.xml,v 1.2 2005/10/20 05:33:42 radek Exp $"*

9.1. FigForth 65

* *$Header: /home/radek/cvs/forth-book/sec-figforth65.xml,v 1.2 2004/12/23 21:07:28 radek Exp $*

FigForth pro procesor 6502 používá tradiční model Nepřímo zřetězený kód.

```
  ;                                    LIT          SCREEN 13 LINE 1
0224 83 4C 49 D4        L22       .BYTE $83,'LI',$D4            ; <--- name field
  ;                          <----- link field
0228 00 00                          .WORD 00       ; last link marked by zero
022A 2C 02              LIT         .WORD *+2      ; <----- code address field
022C B1 AE                          LDA (IP),Y     ; <----- start of parameter field
022E 48                             PHA
022F E6 AE                          INC IP
0231 D0 02                          BNE L30
0233 E6 AF                          INC IP+1
0235 B1 AE              L30         LDA (IP),Y
0237 E6 AE              L31         INC IP
0239 D0 02                          BNE PUSH
023B E6 AF                          INC IP+1
                         ;
023D CA                 PUSH        DEX
023E CA                             DEX
023F 95 01              PUT         STA 1,X
0241 68                             PLA
0242 95 00                          STA 0,X
                        ;
                        ;    NEXT is the address interpreter that moves from machine
                        ;    level word to word.
                        ;    Uses 25 bytes of memory.
0244 A0 01              NEXT        LDY #1
0246 B1 AE                          LDA (IP),Y     ; Fetch code field address pointed
0248 85 B2                          STA W+1        ; to by IP.
024A 88                             DEY
024B B1 AE                          LDA (IP),Y
024D 85 B1                          STA W
                        ;@          JSR TRACE      ; Remove this when all is well
024F 18                             CLC            ; Increment IP by two.
0250 A5 AE                          LDA IP
0252 69 02                          ADC #2
0254 85 AE                          STA IP
0256 90 02                          BCC L54
0258 E6 AF                          INC IP+1
025A 4C B0 00           L54         JMP W-1        Jump to an indirect jump (W) which
                        ;                          vectors to code pointed to by a code
                        ;                          field.
```

9.2. Forth 65

* *$Header: /home/radek/cvs/forth-book/sec-forth65.xml,v 1.2 2004/12/23 21:07:28 radek Exp $*

Forth65 je má implementace Forthu pro procesor 6502 a mikropočítač Atari800XL. Je založena na technologii přímo zřetězeného kódu Přímo zřetězený kód.

```
NEXT:           ; W=(IP), IP=IP+2
                Wh = (IP)
                IP = IP+1
                Wl = (IP)
                IP = IP+1

                JMP (W)



BOS = $8E
            TOS = $C6
            N   = $F0
            IP  = N+8
            W   = IP+3
            UP  = W+2

NEXT:
            ; W=(IP)
            LDY #1
            LDA (IP),Y
            STA W+1
            DEY
            LDA (IP),Y
            STA W
            ; IP=IP+2
            CLC
            LDA IP
            ADC #2
            STA IP
            BCC +L2
            INC IP+2
L2:
            ;
            JMP W-1
```

Tento Forth používá technologii DTC (*Direct Threaded Code*).

9.2.1. Registry a paměť

Tabulka 9-1. Přidělení registrů

Reg.	CPU Reg.	Poznámka
W	zpage	
IP	zpage	ukazatel instrukcí v HL definicích

Reg.	CPU Reg.	Poznámka
PSP	X	zásobník parametrů, umístněn v zpage
RSP	S	zásobník návratových adres
UP	zpage	
TOS	memory	

Pomocný registr W

```
W   = $FB
WL  = $FB
WH  = $FC
```

Ukazatel instrukcí IP je použit v High Level slovech. Ukazuje na adresu v definici.

```
; Instruction Pointer
IPJUMP = $FD       ; Instrukce skoku JMP
IP  = $FE
IPL = $FE
IPH = $FE
```

Parameter Stack je v zpage

```
; od $8E do $C6
PSPB = $8E
PSPT = $C6
```

9.2.2. Struktura záznamu slova ve slovníku

```
NAME:   .BYTE $85       ; Dolní bity - počet znaků jména slova
        .BYTE "slovo"
LINK:   .WORD 0         ; spojovací pole ukazuje na NAME předchozího
                        ; slova.  U prvního slova obsahuje 0
CODE:   ; strojový kód slova.
```

U nízkoúrovňových slov obsahuje pole CODE přímo kód slova. U vysokoúrovňových slov obsahuje volání interpretu adres NEST

```
CODE:   CALL NEST
        .WORD slovo1
        .WORD slovo2
        .WORD slovo3
        .WORD EXIT
```

Volání je následováno seznamem adres které ukazují na kód použitých slov. Ukončení je v provádění kódu slova EXIT.

9.2.3. Vnitřní interpret NEST

Vnitřní interpret provádí interpretaci definice slova. V principu dělá to, že postupně volá kód na adresách uvedených v definici. Přesněji zapsáno provádí:

```
NEST:   -(RSP) = IP
        POP IP
        JUMP NEXT
```

Protože RSP je S bude to vypadat takto

```
NEST:   W = (RSP)+      ; POP W
        -(RSP) = IP
        IP = W
        JUMP NEXT
```

```
                ; Vyzednutí adresy uložené instrukcí JSR NEST ze zásobníku.
                NEST:
0600 68                 PLA
0601 85 FB              STA W
0603 68                 PLA
0604 85 FC              STA W+1

                        ; Uložení aktuální hodnoty IP na zásobník RSP
0606 A5 FE              LDA IP
0608 48                 PHA
0609 A5 FF              LDA IP+1
060B 48                 PHA

                        ; Adresa vyzvednutá ze zásobníku a uložená v W
                        ; je o 1 menší než potřebujeme.  Je to vlastnost
                        ; procesoru 6502

060C 18                 CLC
060D A5 FB              LDA W
060F 69 01              ADC #1
0611 85 FE              STA IP
0613 A5 FC              LDA W+1
0615 69 00              ADC #0
0617 85 FF              STA IP+1

                        ; JUMP NEXT      ; Vykonání
0619 4C 17 06           JMP NEXT
```

Varianta pro JMP místo JSR

```
NEST:
            ; -(RSP)=IP
            ; Uložení ukazatele instrukcí na zásobník návratových adres
            ; Uložení obsahu IP do zásobníku S (=RSP)
            LDA IPL
            PHA
            LDA IPH
            PHA

            ; IP=W+3
            ; Vyzvednutí první adresy v definici slova
            ; V registru W je adresa CF tohoto slova a první adresa je o tři
```

```
;  byty dále
CLC
LDA WL
ADC #3
STA IPL
LDA WH
ADC #0
STA IPH

; JUMP NEXT      ; Vykonání
JMP NEXT
```

9.2.4. Vykonání slova

Procedura `NEXT` vykoná slovo v definici na jehož adresu ukazuje registr `IP`. `IP` je před skokem opraveno (posunuto na další adresu).

```
NEXT:   W=(IP)
        IP=IP+2
        JUMP W

                        ; W=(IP)
0620 A0 00    NEXT:     LDY #0
0622 B1 FE              LDA (IP),Y      ; Fetch code address pointed by IP
0624 85 FB              STA W
0626 C8                 INY
0627 B1 FE              LDA (IP),Y
0629 85 FC              STA W+1

              ;optional JSR TRACE

                        ; Advance IP.  IP=IP+2
062B 18                 CLC
062C A5 FE              LDA IP
062E 69 02              ADC #2
0630 85 FE              STA IP
0632 90 02              BCC 1+      ; skip inc if ...
0634 E6 FF              INC IP+1
              1:
                        ; JUMP W
0636 6C FB 00           JMP (W)
```

9.2.5. Ukončení slova

Návrat ze slova definovaného standardním způsobem (pomocí „;“). Adresa této procedury je poslední adresou v definici slova. Může se však vyskytnout i uvnitř slova.

```
           EXIT:     ; IP = (RSP)+
0640 68                 PLA
0641 85 FE              STA IP
```

```
0643 68                 PLA
0644 85 FF              STA IP+1
0646 4C 17 06           JMP NEXT
0649
```

9.2.6. Startovací kód

```
                        ; Startovací kód
              START:    ; PSP = $80
0650 A2 80              LDX #$80
0652 A9 86              LDA #$86
0654 85 FE              STA IP
0656 A9 06              LDA #$06
0658 85 FF              STA IP+1
065A 4C 17 06           JMP NEXT
```

9.3. XForth

* *$Header: /home/radek/cvs/forth-book/sec-xforth.xml,v 1.1 2003/12/28 18:21:56 radek Exp $*

FIXME: obsah

Hlavička slova `LIT`

```
NFA_LIT:    BYTE $83, "LIT"
LFA_LIT:    WORD 0
CFA_LIT:    WORD PFA_LIT
```

Tělo, kód slova je uložen v poli `PFA`

```
PFA_LIT:    LDA (IP),Y  ; Lo-Byte der inline folgenden Zahl holen
            PHA         ; und auf den Stack retten
            INC IP      ; Instruction-Pointer inkrementieren
            BNE L30
            INC IP+1
L30:        LDA (IP), Y ; Hi-Byte der inline folgenden Zahl holen
L31:        INC IP      ; Instruction-Pointer inkrementieren
            BNE PUSH
            INC IP+1

PUSH:       DEX         ; Datenstack-Pointer dekrementieren
            DEX

PUT:        STA 1,X     ; Hi-Byte (ist in A) auf den Datenstack legen
            PLA         ; Lo-Byte vom Stack holen
            STA 0,X     ; und auf den Datenstack legen
```

Der Adressinterpreter NEXT holt die Adresse des naechsten Secondaries und fuehrt es aus.

```
NEXT:       LDY #1
            LDA (IP),Y  ; Hi-Byte der Wortadresse
            STA W+1     ; im W-Register ablegen
            DEY
            LDA (IP),Y  ; Lo-Byte der Wortadresse
            STA W       ; im W-Register ablegen
            CLC
            LDA IP      ; IP um 2 inkrementieren
            ADC #2
            STA IP
            BCC L54
            INC IP+1
L54:        JMP W-1     ; indirekten Sprung nach (W) ausfuehren

DOCOL:
            ; -(RSP)=IP
            LDA IP+1
            PHA
            LDA IP
            PHA

            ; IP=W+2
            CLC
            LDA W
            ADC #2
            STA IP
            TYA
            ADC W+1
            STA IP+1

            ; JUMP NEXT
            JMP NEXT

EXIT:
            ; IP=(RSP)+
            PLA
            STA IP
            PLA
            STA IP+1
            ; JUMP NEXT
            JMP NEXT
```

9.4. FIG6502

Ze začátku souboru `FIG6502.ASX`

```
;
;                       Through the courtesy of
;
;                        FORTH INTEREST GROUP
;                           P.O. BOX  2154
;                        OAKLAND, CALIFORNIA
;                               94621
;
```

```
;
;
;                          Release 1.k0010
;
;                      with compiler security
;                                and
;                      variable length names
;
;    Further dstribution need not include this notice
;    The FIG installation Manual is required as it contains
;    the model of FORTH and glossary of the system.
;    Might be available from FIG at the above address for $95.00 postpaid.
;
;    Translated from FIG model by W.F. Ragsdale with input-
;    output given for Rockwell System-65. Transportation to
;    other systems requires only the alteration of :
;                XEMIT, XKEY, XQTER, XCR, AND RSWL
;


;    Equates giving memory assignments, machine
;    registers, and disk parameters.
;
SSIZE     EQU 128           ; sector size in bytes
NBUF      EQU 8             ; number of buffers desired in RAM
;                             (SSIZE*NBUF >= 1024 bytes)
SECTR     EQU 800           ; sector per drive
;                             forcing high drive to zero
SECTL     EQU 1600          ; sector limit for two drives
;                             of 800 per drive.
BMAG      EQU 1056          ; total buffer magnitude, in bytes
;                             expressed by (SSIZE+4)*NBUF
;
BOS       EQU $20           ; bottom of data stack, in zero-page.
TOS       EQU $9E           ; top of data stack, in zero-page.
N         EQU TOS+8         ; scratch workspace
IP        EQU N+8           ; interpretive pointer
W         EQU IP+3          ; code field pointer
UP        EQU W+2           ; user area pointer          ❶
XSAVE     EQU UP+2          ; temporary for X register.
;
TIBX      EQU $0100         ; terminal input buffer of 84 bytes.
ORIG      EQU $0200         ; origin of FORTH's Dictionary.
MEM       EQU $4000         ; top of assigned memory+1 byte.
UAREA     EQU MEM-128       ; 128 bytes of user area
DAREA     EQU UAREA-BMAG    ; disk buffer space.
;
;         Monitor calls for terminal support
;
OUTCH     EQU $D2C1         ; output one ASCII char. to term.
INCH      EQU $D1DC         ; input one ASCII char. to term.
TCR       EQU $D0F1         ; terminal return and line feed.
;
;    From DAREA downward to the top of the dictionary is free
;    space where the user's applications are compiled.
```

❶ Terminal Input Buffer

```
;
;    Boot up parameters. This area provides jump vectors
;    to Boot up  code, and parameters describing the system.
;
;
          ORG   ORIG
;
                        ; User cold entry point
ENTER     NOP           ; Vector to COLD entry
          JMP COLD+2    ;
REENTR    NOP           ; User Warm entry point
          JMP WARM      ; Vector to WARM entry
          .WORD $0004   ; 6502 in radix-36
          .WORD $5ED2   ;
          .WORD NTOP    ; Name address of MON
          .WORD $7F     ; Backspace Character
          .WORD UAREA   ; Initial User Area
          .WORD TOS     ; Initial Top of Stack
          .WORD $1FF    ; Initial Top of Return Stack
          .WORD TIBX    ; Initial terminal input buffer
;
;
          .WORD 31      ; Initial name field width
          .WORD 0       ; 0=no disk, 1=disk
          .WORD TOP     ; Initial fence address
          .WORD TOP     ; Initial top of dictionary
          .WORD VL0     ; Initial Vocabulary link ptr.



;
;    NEXT is the address interpreter that moves from machine
;    level word to word.
;
NEXT      LDY #1
          LDA (IP),Y    ; Fetch code field address pointed
          STA W+1       ; to by IP
          DEY
          LDA (IP),Y
          STA W
          JSR TRACE     ; Remove this when all is well
          CLC           ; Increment IP by two.
          LDA IP
          ADC #2
          STA IP
          BCC L54
          INC IP+1
L54       JMP W-1       ; Jump to an indirect jump (W) which
;                         vectors to code pointed to by a code
;                         field.
```

9.5. Poznámky

Tato část obsahuje různé poznámky.

Odkazy:

- Pountain, R. Object Oriented Forth. (http://www.tutorials-blog.com/forth/ObjectOriented-Forth/) — části blogu
-

9.5.1. Fastest ROMable 6502 NEXT

V následujícím kódu je `JV` adresa v zero page. Na ní je uložen kód instrukce `JMP (addr)` který je zde uložen v rámci inicializace. Za ní na adresách `JV+1` a `JV+2`, které slouží jako 16-bitový registr, je je uložana adresa následující instrukce.

```
NEXT:
        INC JV+1
        BEQ NEXT1:
        INC JV+1
        BEQ NEXT2
        JMP JV
NEXT1:
        INC JV+1
NEXT2:
        INC JV+2
        JMP JV

ENTER:  ; HL def begins with JMP ENTER
        LDA JV+2
        PHA
        JDA JV+1
        PHA
        LDA #$6C
        STA JV
        LDY #0
        LDA (JV+1),Y
        CLC
        ADC #3
        PHA
        INY
        LDA (JV+1),Y
        ADC #0
        STA JV+2
        PLA
        STA JV+1
        JMP JV

EXIT:
        PLA
        STA JV+1
        PLA
        STA JV+2
NEXT:
        INC JV+1
        BEQ NEXT1:
```

```
        INC JV+1
        BEQ NEXT2:
        JMP JV
NEXT1:
        INC JV+1
NEXT2:
        INC JV+2
        JMP JV
```

On the other end of the spectrum, if you want multiple stacks for multiple threads, you can provide 32 deep data stacks and 16 deep return stacks with separate high byte and low byte stacks, giving 8 independent threads with three dedicated pages of RAM:

```
DL -- low data stack byte, aligned on a page boundary
DH = DL+256
RL = RH+256
RH = RL+128
```

If you have 8K RAM, with the zero page and hardware stack, this is 1 1/4K, add a page per thread for user variables, PAD, and Pictured numeric input, and you have allocated 5 1/4K, leaving 2 3/4K for I/O and other uses (block buffers, input buffer, serial port buffer)

```
ENTER:  ; HL def begins with JSR ENTER
        DEY
        LDA JV+2
        STA RH,Y
        LDA JV+1
        STA RL,Y
        LDA #$6C
        STA JV
        PLA
        CLC
        ADC #1
        STA JV+1
        PLA
        ADC #0
        STA JV+2
        JMP JV
ENTER1:
        INC JV+2
        JMP JV

EXIT:
        LDA RH,Y
        STA JV+2
        LDA RL,Y
        STA JV+1
        INY
NEXT:
        INC JV+1
        BEQ NEXT1:
        INC JV+1
        BEQ NEXT2:
        JMP JV
NEXT1:
        INC JV+1
NEXT2:
```

```
            INC JV+2
            JMP JV
```

Kapitola 10. ARM

Informace o implementacích Forthu pro processor ARM.

- Riscy Pygness — Pygmy Forth for the ARM (http://pygmy.utoh.org/riscy/)
-

10.1. Pygmy Forth for the ARM

Pygmy Forth pro ARM je 32 bitový (cell) forth pro processor ARM. Implementace požívá slova velikosti 32 bitů a HL slova jsou posloupnosti 16-ti bitových tokenů. Je kódován pro prostředí processoru LPC2106 který má 128kB flash (0x00000000 - 0x0001FFFF) a 64kB RAM (0x40000000 - 0x4000FFFF). Tokeny mají vnitřní strukturu jejíž součástí je 13-ti bitový ukazatel do tabulky adres.

FIXME:HL slova jsou posloupnosti 16-ti bitových čísel. Horních 15 bitů ukazuje na slova, spodní bit má význam JUMP/CALL. V horních 15-ti bitech je jaště „zakódována“ informace, zdali cílové (volané) slovo je HL či LL. Tato informace se získá porovnáním 15 bitové hodnoty s hranicí. Slova pod touto hranicí jsou LL a slova nad touto hranící pak HL. To nám dovolí skrátit definici slova o pole CFA. Protože všechna LL obsahují kód a všechna HL se vykonávají interpretem `docol`.

Implementace NEXT. High-Level slova jsou posloupnosti 16-ti bitových tokenů. Tokeny mají vnitřní strukturu:

- bity 15..3 — vstupní číslo
- bit 2 — příznak primitive (LL) / High Level
- bit 1 — příznak v které paměti se definice slova nachází RAM/flash
- bit 0 — příznak exit CALL/JUMP. Tímto příznakem rozlišíme, jestli se na cílové slovo přechází skokem (tedy ukončujeme vykonávání stávajícího slova) nebo se přechází voláním s návratem k vykonávání stávajícího slova. Je to exit (";") zakompilovaný do definice slova.

Makra

```
; nxt -- move from one word to the next following IP
        .macro nxt              ; select correct version of inner interpreter
            b nxtTab
        .endm

nxtTab:
        ldrh W, [IP], #2        ; read unsigned half-word then bump IP by 2
nxtexec: ; convenient entry point for use by EXECUTE
        ; 16-bit token is now in W
        ; handle Exit flag
        movs W, W, lsr #1       ; set C flag from original bit 0 (i.e. the jump flag)
        ldrcs IP, [RSTK], #4    ; pop rstack into IP ("unnest") inly if jump=1
        ; handle RAM/flash table flag
        movs W, W, lsr #1       ; set C flag from original bit1
        ; Then load the address of the correct token table into TEMPREG
        ldrcc TEMPREG, ptokens  ; Load temporary register with base
        ldrcs TEMPREG, prtokens ;  address of the chosen token table.
                                ;  If RAM flag was set, use the RAM
                                ;  table.  Otherwise, use the flash table.

        ; Remember the Primitive flag (in C)
```

```
        movs W, W, lsr #1       ; set C flag from original bit2
                                ; we will test this flag after looking
                                ; the token's address in the token table

        ; Lookup word's address in token table.  The 13 bits of the token
        ;  number were originally in bits 15..2 of W but are now in
        ;  bits 12..0 because we shifted W right 3 bits.  Now, shift W
        ;  left 2 bits to convert to a byte offset then add offset to
        ;  start of table, leaving address of entry table item in TEMPREG.
        add TEMPREG, TEMPREG, W, lsl #2

        ; Handle Primitive
        ldrcs pc, [TEMPREG]     ; jump to the primitive if primitive flag
                                ;  was set

        ; Otherwise, handle nesting down to called high-level word
        str IP, [RSTK, #-4]!    push IP to return stack
        ldr IP, [TEMPREG]       load IP with address of new word
        b nxtTab                jump back to nxt to begin handling new word.

        .ltorg  ; force dumping of literal pool

; ; semicolon
exit:
EXIT:
        /* unnest by popping return stack into IP */
        ldr IP, [RSTK], #4      ; pop rstack into IP
        nxt

; NOP ( - )  This serves mainly the purpose of safely occupying
;               16-bits in a high-level word list, for aligning a
;               label in a 4-byte boundary.
NOP:
        nxt
```

10.1.1. MMC rozhraní

```
( SPI interface to MMC disk )
 : BITS-ON  ( mask -)  IOSET !  ;
 : BITS-OFF ( mask -)  IOCLR !  ;
 : PIN13 ( - mask)     $00002000  ;
 : DISABLE-MMC ( -)    PIN13 BITS-ON   ;
 : ENABLE-MMC  ( -)    PIN13 BITS-OFF  ;
 : SPI!@ ( c - c)  ( send a byte then collect the returned byte)
   SPDR !  BEGIN SPSR @ $80 AND UNTIL SPDR @  ;
 : SPI!  ( c -)  ( sned a byte but throw away returned byte)
   SPI!@ DROP  ;
 : SPI@ ( - c) ( send a dumm byte and collect returned byte)
   $FF SPI!@ ;
 : DUMMY-SPI-BYTES ( # -) FOR $FF SPI! NEXT  ;

 : GET-MMC-RESPONSE ( - f)
   ( Try up to 256 times to get a response.  A zero response          )
   (  means no error.  A valid response is a byte whose MSBit is zero.)
   (  Return -1 in case of a time-out. )
   256 ( remaining tries)   ( fall through to next word)
```

```
: (GET-MMC-RESPONSE ( remaining# - f)    (  PAUSE  )
  SPI@ DUP $80 AND NOT IF NIP ( cardResponse)  ; THEN DROP
  ( remaining#)  1- DUP 0= IF DROP -1 ; THEN
  (GET-MMC-RESPONSE  ;
```

FIXME:...

Kapitola 11. Ostatní implemetace

* *chapter*
* *rcsinfo="$Header: /home/radek/cvs/forth-book/ch-ostatni_implementace.xml,v 1.4 2005/10/20 19:19:37 radek Exp $"*
* *print="**psselect -p186-192 forth.ps|foldprn -s8**"*

Seznam Forthů v Debian GNU/Linux Woody

- `gforth`
- `kforth`
- `pforth`
- `yforth`
- `pfe`

Další implementace

- `bigforth` — http://www.jwdt.com/~paysan/bigforth.html
- `4th` — http://www.xshall.nl/~thebeez/4tH/foldertree.html
- IsForth (http://isforth.clss.net)

Seznam Forthů pro platformu Palm

- `QuartusForth`
- `DragonForth`
- `pp forth`

Nyní si v samostatných částech popíšeme některé konkrétní implementace Forthu.

11.1. Z80 Forth

* *$Header: /home/radek/cvs/forth-book/sec-z80_forth.xml,v 1.1 2003/12/28 18:21:56 radek Exp $*

FIXME: obsah

```
; Vnitřní interpret
NEXT:   LD A,(BC)
        LD L,A
        INC BC
        LD A,(BC)
        LD H,A
        INC BC
EXE:    LD E,(HL)
        INC HL
        LD D,(HL)
        EX DE,HL
        JP (HL)

EXECUTE:
        POP HL
        JR EXE

COLON:  LD HL,(RSP)
        DEC HL
        LD (HL),B
        DEC HL
        LD (HL),C
        LD (RSP),HL
```

```
        INC DE
        LD B,D
        LD C,E
        JP (IX)

EXIT:   LD HL,(RSP)
        LD C,(HL)
        INC HL
        LD B,(HL)
        INC HL
        LD (RSP),HL
        JP (IX)
```

11.2. nanoForth

* *section id="nanoforth" xreflabel="nanoForth"*
* *rcsinfo="$Header: /home/radek/cvs/forth-book/sec-nanoforth.xml,v 1.3 2005/10/20 05:33:42 radek Exp $"*

FIXME: obsah

Odkazy a zdroje:

- qUark (quite Utilitarian arkitecture) (http://www.rdrop.com/~cary/mirror/quark.txt)
- .. ()

ToDo

1. První úkol.

11.2.1. Tabulka instrukcí

Tabulka 11-1. Tabulka instrukcí

kód	instr.	slovo	PSP efekt	RSP efekt	popis
0	add	+	(n_1 n_2 ⟶ n_3)	(⟶)	addition
1	nand	NAND	(n1 n2 ⟶ n3)	(⟶)	negative and
2	xor	XOR	(n1 n2 ⟶ n3)	(⟶)	exclusive-or
3	ashr	2/	(n1 ⟶ n2)	(⟶)	arithmetic shift right, sign retained
4	fetch	@	(addr ⟶ n)	(⟶)	fetch cell at addr
5	store	!	(n addr ⟶)	(⟶)	store n to cell at addr
6	lit	LIT	(⟶ n) "value"	(⟶)	push inline cell, PSP++
7	dup	DUP	(n ⟶ n n)	(⟶)	duplicate
8	swap	SWAP	(n_1 n_2 ⟶ n2 n1)	(⟶)	exchange
9	drop	DROP	(n ⟶)	(⟶)	discard
A	tor	>R	(n ⟶)	(⟶ n)	pop parameter stack to return stack
B	rfrom	R>	(⟶ n)	(n ⟶)	pop return stack to parameter stack
C	enter	CALL	(⟶)	(⟶ PC)	DOCOL, NEST - call subroutine
D	exit	SEMIS	(⟶)	(addr ⟶)	RETURN - return from subroutine

kód	instr.	slovo	PSP efekt	RSP efekt	popis
E	zjmp	JZ	(n ⟶)	(⟶)	jump if false
F	iptor	BEGIN	(⟶)	(⟶ ip)	push IP (next instruction fetch addr) to R
	NOP		(⟶)	(⟶)	no operation

11.2.2. Rozšiřující definice

```
ddrop:  drop drop exit
rot:    tor swap rfrom swap exit
over:   tor dup rfrom swap exit
not:    dup and exit
and:    nand dup nand exit
or:     dup nand swap dup nand nand exit
negate: dup nand lit 1 add exit
subtract: dup nand lit 1 add add exit
nondextructive-subtract: enter over enter over enter subtract exit
shl:    dup add exit       ; AKA '2*', "two-star", '<<', shift-left
```

11.2.3. Nezapracované texty

11.2.3.1. nFORTH v2.3

```
Date: Sat, 20 Feb 1999 12:54:34 +0300
From: "Stas Pereverzev"
To: MISC
Subject: Re: nFORTH v2.3
Content-Type: text/plain;
 charset="koi8-r"
Content-Transfer-Encoding: 7bit

>Comments, folks?


You need only five instructions, not 16. They are:

ALU:
1. nand
2. shr

RAM:
3. store ( addr n -- )
4. lit   ( -- n )

CONTROL:
5. ncret  \ JUMP to addr in N, if carry flag isn't set in T,
          \ also drop both T and N


Also, if PC is memory variable (or can be addressed as memory variable)
```

```
we can awoid "ncret" instruction:
In that case we sholud use NCSTORE instead STORE:
ncstore (addr n flag -- )

ncstore (addr n 0 ) - same as store,
ncstore (PC n -1 )  - same as ncret

We need only 2 bits per instruction in that case.

That all folks ;-)

Stas.
```

11.3. Color FORTH

* *section id="colorforth" xreflabel="Color FORTH"*
* *rcsinfo="$Header: /home/radek/cvs/forth-book/sec-colorforth.xml,v 1.2 2005/10/20 05:33:42 radek Exp $"*

Odkazy:

- Dispelling the User Illusion, Charles Moore 5/22/99 (http://www.ultratechnology.com/cm52299.htm)
- 1x Forth by Charles Moore 4/13/99 (http://www.ultratechnology.com/1xforth.htm)
- . ()

ToDo

1. úkol

Color Forth pochází z ruky Charlese Moora. Jedná se o přepracovanou, velmi promyšlenou a zjednodušenou (nikoliv na úkor kvality) implementaci. Pokusím se zde ve stručnosti shrnout své poznatky o Color Forthu.

Jak již název napovídá, hrají barvy důležitou roli. Charles zrušil řadu slov a některá z nich nahradil barvou. Například definice nového slova.

11.3.1. Definice nového slova

Standardní slova „:“ a „;“ nahradila barva. Zápis

```
: WORD ... ;
```

je nahrazen

```
WORD ...
```

* *Vytvořit barevné obrázky a vložit místo popisných*

kde slova WORD je vyvedeno v barvě červené a závěrečný středník chybí, neb je jasné kde definice slova končí. Končí definicí slova dalšího, nebo koncem textu.

Text definice slova je pak vyveden v barvě zelené (kompiluje se), černé na bílém pozadí (vykonává se) a červené (jméno definovaného slova)

Středník se zajisté v kódu ještě vyskytuje a často, ale má jiný význam. Místo ukončení definice funguje jako návrat (exit, return) z vykonávaného slova.

11.3.2. Manipulace se zásobníkem

Manipulace se zásobníkem je také jednodušší. Charles používá jen slova **DUP**, **DROP**, **OVER** a slovo **SWAP** jenž není atomickou instrukcí. Úplně zavrhuje slova jako **PICK**, **ROLL** a další jenž provádějí příliš komplexní změny na zásobníku.

11.3.2.1. Komentáře

Jedním slovem žádné. Kód má být natolik jednoduchý, vždy maximálně dva řádky na slovo a samovysvětlující. Rovněž nepoužívá zásobníkové komentáře.

11.3.2.2. Programové konstrukce

Myšlen cykly, větvení, ...

Instrukce pro větvení je jen jedna a to

```
IF ... ; THEN
```

část **ELSE** byla úplně vypuštěna.

Po zralé úvaze byly všechny konstrukce cyklů vypuštěny. Jsou nahrazeny „rekurzivním“ voláním definovaného slova.

```
WORD ... IF ... WORD ; THEN --- ;
```

Sémantika **IF** byla také změnena. Teď již neodstraňuje logickou hodnotu ze zásobníku ale ji tam ponechává. Její případné odstranění (**DROP**) je pak na programátorovi. Přibyla však varianta **-IF** která testuje znaménko.

11.3.2.3. Adresový registr

Charles přidal nové registry. Registr `A` a odpovídající operace **@+**, **!+**, **A!** a **A**.

A registr `R` je specifický a používá se prakticky jen při přesunech v paměti. Má automatickou inkrementace při použití. Ovládá se slovy **@R**, **!R**.

11.3.2.4. Grafické rozhraní

Velikost zobrazované plochy je 1024x768 bodů. Na této ploše je 40x24 znaků každý veliký 16x24 bodů

11.4. Implemetace pro 8-mi bitové procesory

FIXME:

11.4.1. AVR

FIXME:

Odkazy:

- avrforth (http://krue.net/avrforth/)
- ???: Forth AVR Stamp Hardware & Compiler (http://www.mpeltd.demon.co.uk/avrstamp.htm)
- ???: SX-Forth for AVR Butterfly ... and others AVR with 8KB Flash (http://www.avrfreaks.net/index.php?func=viewItem&item_id=716&module=Freaks%20Tools)
- COMERCIAL: Forth for the Atmel AVR microcontrollers (http://www.ram-tech.co.uk/avr.htm)

11.4.1.1. PFAVR

Odkazy:

- PFAVR -- An ANS Forth Implementation for the Atmel AVR (http://claymore.engineer.gvsu.edu/~steriana/Software/pfavr/rationale.html)

PFAVR (http://claymore.engineer.gvsu.edu/~steriana/Python/pfavr/index.html) je implementace ANS Forthu pro procesory AVR firmy Atmel. Jedná se o 16-ti bitovou implementaci kde velikost buňky je 16 bitů (2 bajty). Ke svému běhu potřebuje 13K words paměti FLASH a 32KB externí paměti RAM. Jediné podporované procesory jsou tedy ATmega64, ATmega128 a vyšší.

11.4.1.2. amforth

Odkazy:

- amforth: Forth for AVR ATmega (http://sourceforge.net/projects/amforth/)
- amforth (http://amforth.sourceforge.net/)

11.5. Implemetace pro 16-ti bitové procesory

FIXME:

11.6. Implemetace pro 32-ti bitové procesory

FIXME:

11.6.1. Procesor i386

FIXME:

11.6.1.1. SwiftForth

Implementace SwiftForth (http://www.forth.com/swiftforth/index.html) od firmy FORTH, Inc. (http://www.forth.com/index.html).

Tato implementace je založena na modelu STC. Spouští se jako GUI aplikace pod subsystémem Win32 operačních systému Windows-95 a Windows-NT.

Tabulka 11-2. Význam registrů

registr	použití
EBX	top of stack
ESI	user area pointer
EDI	executable base address
EBP	data stack pointer
ESP	return stack pointer

Kapitola 12. Různé

* *chapter*
* *rcsinfo="$Header: /home/radek/cvs/forth-book/ch-misc.xml,v 1.4 2005/10/20 05:33:42 radek Exp $"*

FIXME:

12.1. Adresový register

FIXME:

```
\ fetch
: @   ( a ⟶ n )
    A! @A ;

\ store
: !   ( n a ⟶ )
    A! !A ;


A! ( a ⟶ )    \ A ⟵ TOS
@A ( ⟶ n )    \ TOS ⟵ mem[A]
!A ( n ⟶ )    \ mem[A] ⟵ TOS
```

Dalším rozšířením je adresní register s autoinkrementací/dekrementací.

```
+!   ( n a ⟶ )   \ Add n to mem[A] and store to mem[A]
```

12.2. Řetězce znaků

* *$Header: /home/radek/cvs/forth-book/sec-strings.xml,v 1.1 2003/12/28 18:21:56 radek Exp $*

FIXME: obsah

$@ — string fetch
$! — string store
$+! — string append
$c! — string char append
$/ — string slash ???
$^ — string where

III. Palm OS

Kapitola 13. Palm OS

* *rcsinfo="$Header: /home/radek/cvs/forth-book/ch-palmos.xml,v 1.3 2005/10/20 05:33:42 radek Exp $"*

epigram

Text kapitoly

13.1. Obsazení registrů procesoru

Podle PalmOSCallingConventions (http://sleepless-night.com/cgi-bin/twiki/view/Main/PalmOSCallingConventions) na webu Sleepless-Night Wiki (http://kristopherjohnson.net/cgi-bin/twiki/view/Main/).

Tabulka 13-1. Obsazení registrů procesoru

Registry	Obsah
D0, D1	Vrácená hodnota se nachází v registru D0. Pokud má více něž 32 bitů, tedy 48 či 64 je i v registru D1.
A0	Pokud je vrácená hodnota ukazatelem, není v D0, D1 ale v A0.
A0, A1 a D0 - D2	Mohou být libovolně použity.
A7	Slouží jako systémový zásobník návratových adres a parametrů.
A5	Je ukazatel na blok globálních parametrů programu. Je nastaven operačním systémem při startu aplikace. Hodnota v A5 musí být obnovena před voláním jakékoliv rutiny, jenž používá globální proměnné.

Kapitola 14. Analýza několik aprogramů pro Palma

* *rcsinfo="$Header: /home/radek/cvs/forth-book/ch-palmprogs.xml,v 1.3 2005/10/20 05:33:42 radek Exp $"*

epigram

Tato kapitola se yabývá analýzou několika náhodně vzbraných programů pro PalmOS. Snažím se zjistit jaká je vnitřní struktura těchto aplikací, jaký je jejich kód. Cílem je poučit se jak se dělá program pro Palma.

14.1. Keyring

Tato aplikace je vyvýjena pomocí gcc.

14.1.1. Část *code 0*

```
0000: 0000 0028
0004: 0000 01DC
0008: 0000 0008
000C: 0000 0020
0010: 0000 3F3C
0014: 0001 A9F0
```

14.1.2. Část *code 1*

Příklad 14-1. Začátek kódu v části *code 1*

```
0000: 4E56 FFF4           START:          LINK    A6, #-12
0004: 48E7 1F00                           MOVEM.L D3-D7, -(A7)
0008: 486E FFFC                           PEA     -4(A6)
000C: 486E FFF8                           PEA     -8(A6)
0010: 486E FFF4                           PEA     -12(A6)
0014: 4E4F A08F                           SYSTRAP SysAppStartup
0018: 4EEF 000C                           LEA     12(A7), A7
001C: 4A40                                TST     D0
001E: 670E                              .-<BEQ     $002E
0020: 1F3C 0003                         |  MOVE.B  #3, -(A7)
0024: 4E4F A234                         |  SYSTRAP SndPlaySystemSound
0028: 70FF                              |  MOVEQ   #-1, D0
002A: 6000 006A                       .-|-<BRA     $0096
002E: 206E FFF4                       | ‘->MOVE.L  -12(A6), A0
0032: 3C10                            |    MOVE    (A0), D6
0034: 2A28 0002                       |    MOVE.L  2(A0), D5
0038: 3828 0006                       |    MOVE    6(A0), D4
003C: 3604                            |    MOVE    D4, D3
003E: 0243 0004                       |    ANDI    #4, D3
0042: 6704                            | .-<BEQ     $0048
0044: 6100 42CE                       | |  BSR     $4314
0048: 3F04                            | ‘->MOVE    D4, -(A7)
004A: 2F05                            |    MOVE.L  D5, -(A7)
```

```
004C: 3F06                          |    MOVE    D6, -(A7)
004E: 6100 4314                     |   =BSR     $4364
0052: 508F                          |    ADDQ.L  #8, A7
0054: 4A43                          |    TST     D3
0056: 6704                          | .-<BEQ     $005C
0058: 6100 43A6                     | |  BSR     $4400
005C: 3F04                          | '->MOVE    D4, -(A7)
005E: 6100 4414                     |   =BSR     $4474
0062: 3F04                          |    MOVE    D4, -(A7)
0064: 2F05                          |    MOVE.L  D5, -(A7)
0066: 3F06                          |    MOVE    D6, -(A7)
0068: 6100 0240                     |   =BSR     $02AA
006C: 2E00                          |    MOVE.L  D0, D7
006E: 4FEF 000A                     |    LEA     10(A7), (A7)
0072: 4A43                          |    TST     D3
0074: 6704                          | .-<BEQ     $007A
0076: 6100 43C2                     | | =BSR     $443A
007A: 3F04                          | '->MOVE    D4, -(A7)
007C: 2F05                          |    MOVE.L  D5, -(A7)
007E: 3F06                          |    MOVE    D6, -(A7)
0080: 6100 4320                     |   =BSR     $43B2
0084: 2F2E FFFC                     |    MOVE.L  -4(A6), -(A7)
0088: 2F2E FFF8                     |    MOVE.L  -8(A6), -(A7)
008C: 2F2E FFF4                     |    MOVE.L  -12(A6), -(A7)
0090: 4E4F A090                     |    SYSTRAP SysAppExit
0094: 2007                          |    MOVE.L  D7, D0
0096: 4CEE 00F8 FFE0                '--->MOVEM.L -32(A6), D3-D7
009C: 4E45                               UNLK    A6
009E: 4E75                               RTS
```

Příklad 14-2. Procedura *$00A0*

```
00A0: 4E56 FFE4           SUB_00A0:      LINK    A6, #-28
00A4: 48E7 1F00                          MOVEM.L D3,D7, -(A7)
00A8: 7CE8                               MOVEQ   #-24, D6
00AA: DC8E                               ADD.L   A6, D6
00AC: 4247                               CLR     D7
00AE: 4267                _loop:.--------->CLR     -(A7)
00B0: 4E4F A039                 |        SYSTRAP MemHeapCheck
00B4: 3F3C 0001                 |        MOVE    #1, -(A7)
00B8: 4E4F A039                 |        SYSTRAP MemHeapCheck
00BC: 4878 FFFF                 |        PEA     -1
00C0: 2F06                      |        MOVE.L  D6, -(A7)
00C2: 4E4F A11D                 |        SYSTRAP EvtGetEvent
00C6: 2F06                      |        MOVE.L  D6, -(A7)
00C8: 6100 4208                 |        BSR     $42D2
00CC: 2F06                      |        MOVE.L  D6, -(A7)
00CE: 4E4F A0A9                 |        SYSTRAP SysHandleEvent
00D2: 4EEF 0014                 |        LEA     20(A7), A7
00D6: 4A00                      |        TST.B   D0
00D8: 6668                      | .------<BNE     $0142
00DA: 486E FFE6                 | |      PEA     -26(A6)
00DE: 2F06                      | |      MOVE.L  D6, -(A7)
00E0: 42A7                      | |      CLR.L   -(A7)
00E2: 4E4F A1BF                 | |      SYSTRAP MenuHandleEvent
00E6: 4FEF 000C                 | |      LEA     12(A7), A7
00EA: 4A00                      | |      TST.B   D0
```

```
00EC: 6654                    | +-------<BNE     $0142
00EE: 4205                    | |        CLR.B   D5
00F0: 0C6E 0017 FFE8          | |        CMPI    #23, -24(A6)
00F6: 663C                    | | .------<BNE     $0134
00F8: 362E FFF0               | | |      MOVE    -16(A6), D3
00FC: 3F03                    | | |      MOVE    D3, -(A7)
00FE: 4E4F A16F               | | |      SYSTRAP FrmInitForm
0102: 280F                    | | |      MOVE.L  A0, D4
0104: 2F04                    | | |      MOVE.L  D4, -(A7)
0106: 4E4F A174               | | |      SYSTRAP FrmSetActiveForm
010A: 5C8F                    | | |      ADDQ.L  #6, A7
010C: 0C43 03E8               | | |      CMPI    #1000, D3
0110: 670A                    | | |   .-<BEQ     $011C
0112: 0C43 03E9               | | |   |  CMPI    #1001, D3
0116: 670C                    | | | .-|-<BEQ     $0124
0118: 6000 001A               | | +-|-|-<BRA     $0134
011C: 41FA 1CAC               | | | | '->LEA     7340(PC), AO    ;1DCA
0120: 6000 0006               | | | | .-<BRA     $0128
0124: 41FA 0DDC               | | | '-|->LEA     3548(PC), A0    ;0F02
0128: 2F08                    | | |   '->MOVE.L  A0, -(A7)
012A: 2F04                    | | |      MOVE.L  D4, -(A7)
012C: 4E4F A19F               | | |      SYSTRAP FrmSetEventHandle...
0130: 7A01                    | | |      MOVEQ   #1, D5
0132: 508F                    | | |      ADDQ.L  #8, A7
0134: 1E05                    | | '----->MOVE.B  D5, D7
0136: 4A07                    | |        TST.B   D7
0138: 6608                    | +-------<BNE     $0142
013A: 2F06                    | |        MOVE.L  D6, -(A7)
013C: 4E4F A1A0               | |        SYSTRAP FrmDispatchEvent
0140: 588F                    | |        ADDQ.L  #4, A7
0142: 0C6E 0016 FFE8          | '------->CMPI    #22, -24(A6)
0148: 6600 FF64               '---------<BNE     $00AE
014C: 4CEE 00F8 FFD0                     MOVEM.L -48(A6), D3-D7
0152: 4E5E                               UNLK    A6
0154: 4E75                               RTS
```

14.2. QED

14.2.1. Část *code 0*

Příklad 14-3. ***code 0***

```
0000 0028
0000 02BC
0000 0008
0000 0020
0000 3F3C
0001 A9F0
```

14.2.2. Část *code 1*

Příklad 14-4. Začátek části *code 1*

```
0000:                       START:          ORI.B   #1, D0
0004:                                       LINK    A6, #-12
0008:                                       MOVEM.L D3-D7, -(A7)
000C:                                       PEA     -4(A6)
0010:                                       PEA     -8(A6)
0014:                                       PEA     -12(A6)
0018:                                       SYSTRAP SysAppStartup
001C:                                       LEA     12(A7), A7
0020:                                       TST     D0
0022:                                     .- BEQ     $0032
0024:                                     |  MOVE.B  #3, -(A7)
0028:                                     |  SYSTRAP SndPlaySystemSound
002C:                                     |  MOVEQ   #-1, D0
002E:                                   .-|- BRA     $0092
0032:                                   | ‘> MOVE.L  -12(A6), A0
0036:                                   |    MOVE    (A0), D6
0038:                                   |    MOVE.L  2(A0), D5
003C:                                   |    MOVE    6(A0), D4
0040:                                   |    MOVE    D4, D3
0042:                                   |    ANDI    #4, D3
0046:                                   | .- BEQ     $004C
0048:                                   | |  BSR     5054
004C:                                   | ‘> MOVE    D4, -(A7)
004E:                                   |    MOVE.L  D5, -(A7)
0050:                                   |    MOVE    D6, -(A7)
0052:                                   |    BSR     $50A4
0056:                                   |    ADDQ.L  #8, A7
0058:                                   |    TST     D3
005A:                                   | .- BEQ     $0060
005C:                                   | |  BSR     $5140
0060:                                   | ‘> MOVE    D4, -(A7)
0062:                                   |    MOVE.L  D5, -(A7)
0064:                                   |    MOVE    D6, -(A7)
0066:                                   |    BSR     $009C
006A:                                   |    MOVE.L  D0, D7
006C:                                   |    ADDQ.L  #8, A7
006E:                                   |    TST     D3
0070:                                   | .- BEQ     $0076
0072:                                   | |  BSR     $517A
0076:                                   | ’> MOVE    D4, -(A7)
0078:                                   |    MOVE.L  D5, -(A7)
007A:                                   |    MOVE    D6, -(A7)
007C:                                   |    BSR     $50F2
0080:                                   |    MOVE.L  -4(A6), -(A7)
0084:                                   |    MOVE.L  -8(A6), -(A7)
0088:                                   |    MOVE.L  -12(A6), -(A7)
008C:                                       SYSTRAP SysAppExit
0090:                                       MOVE.L  D7, D0
0092:                                       MOVEM.L -32(A6), D3-D7
0098:                                       UNLK    A6
009A:                                       RTS
```

```
009C:                                   LINK    A6, #0
00A0:                                   MOVE.L  D3, -(A7)
00A2:                                   MOVE    8(A6), D0
00A6:                                   MOVE.L  10(A6), D1
00AA:                                   TST     D0
00AC:                                 .--- BNE     $00D2
00AE:                                 |    BSR     $010A
00B2:                                 |    TST.B   D0
00B4:                                 | .- BEQ     $00BE
00B6:                                 | |  MOVE    D3, D0
00B8:                                 | |  EXT.L   D0
00BA:                               .-|-|- BRA     $00FA
00BE:                               | | ‘> MOVE    #1000, -(A7)
00C2:                               | |    SYSTRAP FrmGotoForm
00C6:                               | |    BSR     $0216
00CA:                               | |    BSR     $031C
00CE:                               | | .- BRA     $00F8
00D2:                               | ‘-|> CMPI    #1, D0
00D6:                               | .-|- BNE     $00EC
00D8:                               | | |  MOVE    14(A6), D0
00DC:                               | | |  BTST    #4, D0
00E0:                               | | +- BEQ     $00F8
00E2:                               | | |  MOVE.L  D1, -(A7)
00E4:                               | | |  BSR     $4908
00E8:                               | | +- BRA     $00F8
00EC:                               | ‘-|> CMPI    #2, D0
00F0:                               |   +- BNE     $00F8
00F2:                               |   |  MOVE.L  D1, -(A7)
00F4:                               |   |  BSR     $4B60
00F8:                               |   ‘- MOVEQ   #0, D0
00FA:                               ‘----- MOVE.L  -4(A6), D3
00FE:                                      UNLK    A6
0100:                                      RTS


0102:                                   SUBQ    #8, D5
```

14.3. Scripts

14.3.1. Část *code 0*

Příklad 14-5. Část *code 0*

```
0000: 0000 0028
0004: 0000 0128
0008: 0000 0008
000C: 0000 0020
0010: 0000 3F3C
0014: 0001 A9F0
```

14.3.2. Část *code 1*

Příklad 14-6. Úvod části *code 1* programu Scripts

```
0000: 0000 0001           START:          ORI.B   #1, D0
0004: 4E56 FFF4                           LINK    A6, #-12
0008: 48E7 1F00                           MOVEM.L D3-D7, -(A7)
000C: 486E FFFC                           PEA     -4(A6)
0010: 486E FFF8                           PEA     -8(A6)
0014: 486E FFFC                           PEA     -12(A6)
0018: 4E4F A08F                           SYSTRAP SysAppStartup
001C: 4FEF 000C                           LEA     12(A7), A7
0020: 4A40                                TST     D0
0022: 670E                              .-<BEQ    $0032
0024: 1F3C 0003                         |  MOVE.B  #3, -(A7)
0028: 4E4F A234                         |  SYSTRAP SndPlaySystemSound
002C: 70FF                              |  MOVEQ   #-1, D0
002E: 6000 0062                       .-|-<BRA    $0092
0032: 206E FFF4                       | ‘> MOVE.L  -12(A6), A0
0036: 3C10                            |    MOVE    (A0), D6
0038: 2A28 0002                       |    MOVE.L  2(A0), D5
003C: 3828 0006                       |    MOVE    6(A0), D4
0040: 3604                            |    MOVE    D4, D3
0042: 0243 0004                       |    ANDI    #4, D3
0046: 6704                            | .-<BEQ     $004C
0048: 6100 0EFE                       | |  BSR     $0F48
004C: 3F04                            | ‘> MOVE    D4, -(A7)
004E: 2F05                            |    MOVE.L  D5, -(A7)
0050: 3F06                            |    MOVE    D6, -(A7)
0052: 6100 0F44                       |    BSR     $0F98
0056: 508F                            |    ADDQ.L  #8, A7
0058: 4A43                            |    TST     D3
005A: 6704                            | .-<BEQ     $0060
005C: 6100 0FD6                       | |  BSR     $1034
0060: 3F04                            | ‘->MOVE    D4, -(A7)
0062: 2F05                            |    MOVE.L  D5, -(A7)
0064: 3F06                            |    MOVE    D6, -(A7)
0066: 6100 063E                       |    BSR     $06A6
006A: 2E00                            |    MOVE.L  D0, D7
006C: 508F                            |    ADDQ.L  #8, A7
006E: 4A43                            |    TST     D3
0070: 6704                            | .-<BEQ     $0076
0072: 6100 0FFA                       | |  BSR     $106E
0076: 3F04                            | ‘->MOVE    D4, -(A7)
0078: 2F05                            |    MOVE.L  D5, -(A7)
007A: 3F06                            |    MOVE    D6, -(A7)
007C: 6100 0F68                       |    BSR     $0FE6
0080: 2F2E FFFC                       |    MOVE.L  -4(A6), -(A7)
0084: 2F2E FFF8                       |    MOVE.L  -8(A6), -(A7)
0088: 2F2E FFF4                       |    MOVE.L  -12(A6), -(A7)
008C: 4E4F A090                       |    SYSTRAP SysAppExit
0090: 2007                            |    MOVE.L  D7, D0
0092: 4CEE 00F8 FFE0                  ‘--->MOVE.L  -32(A6), D3-D7
0098: 4E5E                                UNLK    A6
009A: 4E75                                RTS
```

Příklad 14-7. Procedura $009C

```
009C: 4E56 FFF4           SUB_009C:       LINK    A6, #-12
00A0: 2F03                                MOVE.L  D3, -(A7
00A2: 262E 0008                           MOVE.L  8(A6), D3
00A6: 42A7                                CLR.L   -(A7)
00A8: 486E FFFE                           PEA     -2(A6)
00AC: 42A7                                CLR.L   -(A7)
00AE: 42A7                                CLR.L   -(A7)
00B0: 486E FFFA                           PEA     -6(A6)
00B4: 2F03                                MOVE.L  D3, -(A7)
00B6: 4E4F A04C                           SYSTRAP DmOpenDatabaseIn...
00BA: 4FEF 0018                           LEA     24(A7), A7
00BE: 4A40                                TST     D0
00C0: 662C                              .-<BNE     $00EE
00C2: 42A7                              |  CLR.L   -(A7)
00C4: 42A7                              |  CLR.L   -(A7)
00C6: 42A7                              |  CLR.L   -(A7)
00C8: 486E FFF6                         |  PEA     -10(A6)
00CC: 42A7                              |  CLR.L   -(A7)
00CE: 42A7                              |  CLR.L   -(A7)
00D0: 42A7                              |  CLR.L   -(A7)
00D2: 42A7                              |  CLR.L   -(A7)
00D4: 42A7                              |  CLR.L   -(A7)
00D6: 42A7                              |  CLR.L   -(A7)
00D8: 42A7                              |  CLR.L   -(A7)
00DA: 2F2E FFFA                         |  MOVE.L  -6(A6), -(A7)
00DE: 3F2E FFFE                         |  MOVE    -2(A6), -(A7)
00E2: 4E4F A046                         |  SYSTRAP DmDatabaseInfo
00E6: 4FEF 0032                         |  LEA     50(A7), A7
00EA: 4A40                              |  TST     D0
00EC: 6708                            .-|-<BEQ     $00F6
00EE: 303C 0203                       | '->MOVE    #515, D0
00F2: 6000 0078                     .-|---<BRA     $016C
00F6: 4AAE FFF6                     | '--->TST.L   -10(A6)
00FA: 664A                          | .---<BNE     $0146
00FC: 4878 0116                     | |    PEA     278
0100: 2F03                          | |    MOVE.L  D3, -(A7)
0102: 4E4F A059                     | |    SYSTRAP DmNewHandle
0106: 2008                          | |    MOVE.L  A0, D0
0108: 508F                          | |    ADDQ.L  #8, A7
010A: 6608                          | | .-<BNE     $0114
010C: 303C 0201                     | | |  MOVE    #513, D0
0110: 6000 005A                     +-|-|-<BRA     $016C
0114: 2F00                          | | '->MOVE.L  D0, -(A7)
0116: 4E4F A020                     | |    SYSTRAP MemHandleToLocal
011A: 2D40 FFF6                     | |    MOVE.L  D0, -10(A6)
011E: 42A7                          | |    CLR.L   -(A7)
0120: 42A7                          | |    CLR.L   -(A7)
0122: 42A7                          | |    CLR.L   -(A7)
0124: 486E FFF6                     | |    PEA     -10(A6)
0128: 42A7                          | |    CLR.L   -(A7)
012A: 42A7                          | |    CLR.L   -(A7)
012C: 42A7                          | |    CLR.L   -(A7)
012E: 42A7                          | |    CLR.L   -(A7)
0130: 42A7                          | |    CLR.L   -(A7)
0132: 42A7                          | |    CLR.L   -(A7)
```

```
0134: 42A7                          | |    CLR.L   -(A7)
0136: 2F2E FFFA                     | |    MOVE.L  -6(A6), -(A7)
013A: 3F2E FFFE                     | |    MOVE    -2(A6), -(A7)
013E: 4E4F A047                     | |    SYSTRAP DmSetDatabaseInfo...
0142: 4FEF 0036                     | |    LEA     54(A7), A7
0146: 3F2E FFFE                     | '--->MOVE    -2(A6), -(A7)
014A: 2F2E FFF6                     |      MOVE.L  -10(A6), -(A7)
014E: 4E4F A036                     |      SYSTRAP MemLocalIDToLocke...
0152: 2608                          |      MOVE.L  A0, D3
0154: 4878 0116                     |      PEA     278
0158: 486D FFDE                     |      PEA     -290(A5)
015C: 42A7                          |      CLR.L   -(A7)
015E: 2F03                          |      MOVE.L  D3, -(A7)
0160: 4E4F A076                            SYSTRAP DmWrite
0164: 2F03                          |      MOVE.L  D3, -(A7)
0166: 4E4F A035                     |      SYSTRAP MemPtrUnlock
016A: 4240                          |      CLR     D0
016C: 262E FFF0                     '----->MOVE.L  -16(A6), D3
0170: 4E5E                                 UNLK    A6
0172: 4E75                                 RTS
```

14.4. SmartDoc

14.4.1. Část *code 0*

Příklad 14-8. Část *code 0*

```
0000: 0000 0178
0004: 0000 06D8
0008: 0000 0008
000C: 0000 0020
0010: 0000 3F3C
0014: 0001 A9F0
```

14.4.2. Část *code 1*

Příklad 14-9. Úvod části *code 1*

```
0000: 0000 0001           START:           ORI.B   #1, D0
0004: 487A 0004                            PEA     4(PC)       ;000A
0008: 0697 0000 00FA                       ADDI.L  #250, (A7)
000E: 4E75                                 RTS
```

IV. Quartus Forth

V této části je popsána implementace jazyka Forth určená pro počítače Palm.

Kapitola 15. Quartus Forth

* *rcsinfo="$Header: /home/radek/cvs/forth-book/ch-quartusforth.xml,v 1.11 2005/10/20 05:33:42 radek Exp $"*

Quartus je latinsky čtyři.

Odkazy:
- „Site“ (http://www.quartus.net)
- Sleepless Night Wiki (http://sleepless-night.com/cgi-bin/twiki/view/Main) — už dávno neexistuje
- PilotForth (http://www.daveltd.com/pilot/PalmOS/)
- Tutorial for a System Programming Example with Quartus Forth (http://www.erwin-schomburg.homepage.t-online.de/MakingNoAlarms.htm) © 2005 Erwin Schomburg

Quartus Forth (http://www.quartus.net/products/forth) je implementace jazyka Forth pro počítače Palm Pilot. Jedná se o kompilátor schopný vytvářet samostatné aplikace. Stojí necelých $100, ale je k dispozici omezená verze zdarama.

15.1. Startup

Popis startování programu QuartusForth a použití poznámky („mema“) `startup.quartus`. Volně podle SleeplessNightWiki (http://sleepless-night.com/cgi-bin/twiki/view/Main/StartupQuartus) *(uz neexistuje)*.

Když QuartusForth startuje, hledá memo s názvem `startup.quartus`. Najde-li ho, nahraje a vykoná. Registrovaní uživatelé zde mají umístněn svůj registrační klíč, který říká programu QuartusForth že je vše v pořádku a zpřístupní jim všechny možnosti vymoženosti.

Vykonávání příkazů v `startup.quartus` s výhodou využijeme pří úpravě prostředí programu. Například při ladění konkrétního programu sem můžeme umístnit jeho spouštění.

```
\ startup.quartus
registered XXX-X-XXXXX-XXXX
needs snowflakes
go
```

nebo si sem umístníme zkratku pro spouštění našeho laděného programu

```
\ startup.quartus
registered XXX-X-XXXXX-XXXX
: snow
  s" needs snowflakes" evaluate
  s" go" evaluate
;
```

Poznámka: Doplnit

15.2. Kostra

Příklad 15-1. Kostra aplikace s obsluhou událostí

```
\ applikace

: dispatch-event ( ekey -- ekey )
  ;

: handle-events ( -- )
  begin
    ekey dispatch-event drop
  again ;

: go ( -- )
  page
  handle-events ;

\ app-1

needs Events
needs ondo    \ Kris's totally awesome dispatching words

: penDown ( -- )
  coords@ ." penDown: " . space . cr ;

: penMove ( -- )
  coords@ ." penMove: " . space . cr ;

: penUp ( -- )
  coords@ ." penUp: " . space . cr ;

: dispatch-event ( ekey -- ekey )
  on: penDownEvent do: penDown
  on: penMoveEvent do: penMove
  on: penUpEvent do: penUp ;

: handle-events ( -- )
  begin
    ekey dispatch-event drop
  again ;

: go ( -- )
  page
  handle-events ;
```

15.3. Grafické uživatelske rozhraní

Popis obsluhy událostí. Volně podle „Quartus Forth Manual“, Sleepless Night Wiki (http://sleepless-night.com/cgi-bin/twiki/view/Main) a dalších zdrojů.

Odkazy

- EventLoop (http://sleepless-night.com/cgi-bin/twiki/view/Main/EventLoop)

Smyčka obsluhy událostí (Event Loop). je částí programu která vybírá (přijímá) události od PalmOs, a dále je zpracovává. Všechny programy *musí* mít tuto smyčku, aby jste se mohli přepnout do jiné aplikace. Nepotřebujete-li obsluhovat žádné události, můžete použít jednoduchou „prázdnou“ smyčku.

Příklad 15-2. Prázdná smyčka obsluhy událostí

```
: default-event-loop ( -- )
  begin
    ekey drop
  again ;
```

> **Varování**
>
> Aby se program choval standardně, musí mít obsluhu událostí, jinak by nešel ani ukončit.

Příklad 15-3. Prázdná smyčka obsluhy událostí

```
: default-event-loop ( -- )
  begin
    ekey drop
  again ;
```

Slovo **ekey** čeká na událost až 500ms. Chceme-li rychlejší smyčku můžeme použít slovo (ekey).

Příklad 15-4. Rychlejší *FIXME:smyčka/cyklus obsuhy událostí

```
: faster-event-loop ( -- )
    begin
        10. (ekey)      ( ⟶ ekey) \ wait max of 100ms
        do-my-thing     ( ekey ⟶ ekey )
        drop            ( ekey ⟶ )
    again ;
```

Úsporná smyčka s „nekonečným“ čekáním.

```
: event-loop ( -- )
  begin
    -1. (ekey)
\   dispatch-event
    drop
  again ;
```

Hodnota -1. slova (ekey) přepne procesor do „doze mode“. Z hlediska našeho programu tento nedostane řízení, dokud nenastane nějaká událost. Rovněž není nikdy vrácena událost nilEvent.

V programu psaném v QuartusForth se nemusíte zajímat o obsluhu většiny událostí, Forth to udělá za vás, kdykoliv použijete **KEY** nebo **EKEY**. Automaticky jsou obsluhovány události

- Systémové události (*System events*)
- Menu (*Menu events*)
- Nahrávání a otevírání formulářů (*Form load and Form open events*)
- _____ (*Form event dispatching*)

Události které nebyli obslouženy automaticky jsou vráceny na zásobník, kde je můžeme přečíst a obsloužit.

Základ obsluhy událostí vypadá takto

```
begin ekey drop again
```

Takováto obsluha ignoruje všechny události. Ve skutečnosti nejsou všechny události ignorovány ale řada z nich je obsloužena standardně.

Příklad 15-5. Ukázka kódu který reaguje na dotek pera

```
\ eh
needs events
: go ( -- )
  begin ekey
    dup penDownEvent = if
      ." Pen Down detected" cr
    else dup penUpEvent = if
      ." Pen Up detected" cr
      then
    then  drop
  again ;
```

Zajímavé je, jakým způsobem je ošetřena událost *appStopEvent*. Quartus Forth vám dává plnou kontrolu nad tím co se bude dít při ukončení programu. Když Quartus Forth obdrží *appStopEvent*, vygeneruje výjimku **-257 THROW**. Za normálních (běžných) okolností je tato výjimka ošetřena standardní obsluhou výjimek a program jednoduše skončí voláním příkazu **(bye)**. Nicméně můžete tuto vájimku odchytit a zařídit si třeba úklid použitých prostředků před voláním **(bye)**

Příklad 15-6. Ukázka obsluhy

```
-257 constant byeThrow
: go ( -- )
  MainForm
  ." Go ahead, start another app." cr
  begin
    ['] key catch
    byeThrow = if
      ." Exiting in 5 seconds!"
      500. SysTaskDelay
      (bye)
    then  drop
  again ;
```

číslo	událost	popis

Tabulka 15-1. Seznam událostí (returned by EKEY) [1:3:7]

číslo	událost	popis
0	nilEvent	žádná událost, nenastala žádná událost
1	penDownEvent	
2	penUpEvent	
3	penMoveEvent	
4	keyDownEvent	
5	winEnterEvent	
6	winExitEvent	
7	ctlEnterEvent	
8	ctlExitEvent	
9	ctlSelectEvent	
10	ctlRepeatEvent	
11	lstEnterEvent	tato událost v QF nenastane
12	lstSelectEvent	
13	lstExitEvent	
14	popSelectEvent	
15	fldEnterEvent	
16	fldHeightChangedEvent	
17	fldChangedEvent	
18	tblEnterEvent	
19	tblSelectEvent	
20	daySelectEvent	
21	menuEvent	
22	appStopEvent	
23	frmLoadEvent	
24	frmOpenEvent	
25	frmGotoEvent	
26	frmUpdateEvent	
27	frmSaveEvent	
28	frmCloseEvent	
29	frmTitleEnterEvent	
30	frmTitleSelectEvent	
31	tblExitEvent	
32	sclEnterEvent	tato událost v QF nenastane
33	sclExitEvent	
34	sclRepeatEvent	tato událost v QF nenastane
32767	firstUserEvent	

15.3.1. Konstra programu zpracování události

Dobře „refaktorizovaný“ program pro zpracování událostí

Import.

```
needs ids
needs Resources
needs OnDo
needs Events
```

Připojení zdrojové databáze.

```
(ID) MyF1 (ID) rsrc use-resources
```

Definice konstant.

```
1000 constant MainForm
2001 constant AboutMenuItem
3000 constant AboutBox
```

Zpracování jednotlivých událostí.

```
: penDown ( ekey ⟶ ekey )
    coords@ ." penDown: " . space . cr ;

: penUp ( ekey ⟶ ekey )
    coords@ ." penUp: " . space . cr ;
```

Dispatch/rozdělění/přidělení události.

```
: dispatch-event ( ekey ⟶ ekey )
    on: penDownEvent    do: penDown
    on: penUpEvent      do: penUp ;
```

Cyklus *FIXME:příjmu/výběru událostí.

```
: handle-events ( ⟶ )
    begin
        ekey dispatch-event drop
    again ;
```

Otevření formuláře.

```
: go ( ⟶ )
    BlankFormID ShowForm
    handle-events ;
```

15.3.2. Výběr události

Příkaz ekey nebo pro výběr událostí s fronty nám vrátí jen typ události. Vlastní popis události musíme získat jinak.

Příklad 15-7. EventType

```
typedef struct {
        eventsEnum      eType;
        Boolean         penDown;
        UInt8           tapCount;
        Int16           screenX;
        Int16           screenY;
        union {
                ...
        } data;
} EvenType;
```

15.3.3. Jak zjistit který prvek událost vyvolal

Nestačí nám jen znát jaká událost vzikla, jak jsme si ukázali v předchozích odílech, my také potřebujeme vědět který že to objekt událost způsobil. Tedý které tlačítko bylo stlačeno, které pole získalo zaměření (focus).

Za použití OnDo rozšíříme rozdělování (dispatch) událostí o událost ctlSelectEvent

```
: dispatch-event ( ekey ⟶ ekey )
 ...
        on: ctlSelectEvent     do: ctlSelect
 ... ;
```

V proceduře **ctlSelect** se pak ptáme které že to tlačítko bylo zmáčknuto. Ke zjištění ID zmáčknutého tlačítka použijeme kód

```
event >abs ItemId
```

. Procedura výběru tlačítka pak vypadaá takto

```
: ctlSelect (ekey ⟶ ekey )
 event >abs ItemID
 on: ButtonOK           do: pushButtonOK
        on: ButtonBackLight    do: pushButtonBacklight
        drop ;  \ musíme zahodit ID získané na začátku slova.
```

Příklad 15-8. Zjištění ID zmáčknutého tlačítka

```
1202 constant ButtonOK
1203 constant ButtonBackLight

: Event. ( ⟶ addr. )
 event >abs ;

: ctlSelect (ekey ⟶ ekey )
 Event. ItemID
 on: ButtonOK           do: pushButtonOK
        on: ButtonBackLight    do: pushButtonBacklight
 ;

: dispatch-event ( ekey ⟶ ekey )
        on:  ctlSelectEvent     do: ctlSelect
```

```
          ;
```

15.3.4. Label

Objekt typu Label a práce s ním.

```
needs Forms

: str>abs ( a n ⟶ a. n )
        r> >abs r> ;

s" HELL" str>abs drop
MyLabel SetLabel
```

15.4. Automated Test Suite

Automatizované testování slov je jedním z důležitých nástrojů při ladění, nebo spíše projektování bezchybných slov. Je založeno na množině testovacích vektorů, které na kterých testujeme zdali se chování našeho nového slova odchyluje od požadavků, či specifikace.

15.5. select ... end-select

select slouží k vytváření pole funkcí. Nejdříve příklad:

```
: zero ." Zero" ;
: one ." One" ;
: two ." Two" ;

: go
  select
    xt zero
    xt one
    xt two
  end-select  execute ;
```

Tabulka 15-2. Slova

Slovo	Modifikace zásobníku	Popis
select	(index --)	
xt	("name" --)	
end-select	(-- xt)	

15.6. Moduly

Popis několika zajímavých modulů a příklady jejich použití.

.

15.7. Práce s pamětí

Prostředí Palm OS je vystavěno na 32-bitové architekruře. Systém používa 32-bitové adresy a základní datové typy jsou 8, 16 a 32 bitů veliké.

Každá paměťová karta má k dispozici adresový prostor 256MB. Paměťová karta 0 začíná na adrese $1000000, paměťová karta 1 na adrese $2000000, atd.

Palm OS dělí pamět RAM na dvě logické části. dynamic RAM a storage RAM. Dynamic RAM je použita jako pracovní prostor apluikací a je ekvivalentem RAM v pracovních stanicích. Zbytek paměti RAM je věnován/určen pro storage RAM a je analogický diskové paměti v pracovní stanici.

Calá *dynamic RAM* je použita pro implementaci jedné haldy (heap) která poskytuje aplikacím paměť pro dynamickou alokaci/přidělení. Paměť z této haldy je možno získat voláním funkce MemHandleNew.

15.7.1. Struktura chunku

Každý chunk začíná 8 bytovou hlavičkou následovanou daty. Hlavička sestává z:

- Flags:sizeAdj (byte). Tento bajt obsahuje příznaky (flags) v horním nibble a opravu velikosti (size adjustment) v dolním nibble.

  Nibble příznaků má momentálně definovaný jen jeden bit jenž je nastaven u volných chunků.

  Nibble sizeAdj může být použit při ... FIXME:
- informace o velikosti 3 bajty
- lock:owner bajt
- hOffset informace 3 bajty

15.7.2. Práce s kouskem (chunk) paměti v dynamické haldě (heap)

Získání paměti provedeme žádostí o paměť MemHandleNew které předáváme jeden parametr a to velikost požadovaného kousku paměti.

```
160
MemHandleNew            ( 160 ⟶ handle. )
```

Před použitím získaného kousku paměti po zápis a čtení je třeba jej uzamknout voláním MemHandleLock které nám vrátí adresu (32bit) na daný kousek (chunk). Jeden chunk smí být uzamčen nejvýše 14 krát. Volání MemHandleUnlock naopak odemkne daný kousek paměti a umožní tak systému jeho dynamické posouvání v paměti na haldě.

Velikost posouvatelného kousku (movable chunk) je možno zjistit voláním MemHandleSize a změnit se dá voláním MemHandleResize. Obecně není možno zvětšit velikost kousku je-li uzamčen. Je potřeba jej odemknout.

Až už není kousku paměti třeba vrátíme jej systému voláním MemHandleFree. Toto volání uvolní/zruší kousek a v případě že je zamknut.

15.7.3. Sumář volání modulu Memory Management

Alokování a uvolňování paměti

MemHandleNew
MemHandleLock

Tabulka 15-3. Allocating and Freeing Memory

MemHandleNew	MemPtrNew

15.8. Databáze

- *The Data Manager* udržuje uživatelská data která jsou uložena v databázích.
- *The Resource Manager*
- *File Streaming Application Program Interface*

Všechna data v Palmovi jsou v databázích, i výkonné programy jsou zapsány jako databáze. Zkrátka databáze je jediný způsob jak na Palmovi ukládat data.

Co to je databáze? Databáze v Palmovi je jednoduše množina záznamů které obsahují data.

15.8.1. Hlavička databáze

Každá databáze ma svoji hlavičku která ji identifikuje a nese pomocné informace. Hlavička sestáva z několika polí, které si dále popíšeme.

Hlavička databáze obsahuje tyto pole:

name

Obsahuje název databáze

attributes

Obsahuje příznaky databáze (*flags*)

version

Číslo verze databáze

modificationNumber

Je zvětšeno pokaždé když je do databáze přidán, smazán či modifikován záznam.

appInfoID

Volitelné pole kam může aplikace ukládat své specifické informace o databázi. Například může být použito k uložení uživatelových preferencí pro zobrazení databáze.

sortInfoID

Další nepovinné pole kam si může aplikace uloži lokální ID třídicí tabulky.

type

pole obsahující 4 bytový identifikátor typu

creator

Identifikační kód tvůrce o velikosti 4 byty.

numRecords

počet položek uložených v hlavičce databáze

15.8.2. Vytvoření a odstranění databáze

Před tím, než budeme s databází pracovat, musíme ji vytvořit. Jediný způsob jak to udělat je voláním `DmCreateDatabase`. Volání vrací příznak chyby *`Err`*. Je-li nenulový, nastala chyba. Uveďme si krátky příklad.

```
# $Id: example.createDict.ses,v 1.1 2003/12/28 18:21:58 radek Exp $
needs zstring
: createDict ( -- Err )
 FALSE
 [ID] Dict
 [ID] SFth
 z" SFDictionary" DROP >ABS
 0
 DmCreateDatabase ;
```

Příznak na řádku 3 určuje vytvoříme li databázi zdrojů, nebo obyčejnou databázi. Je-li `true` bude vytvořena databáze zdrojů (*Resource Database*), je-li `false` bude vytvořena obyčejná databáze.

15.8.3. Záznam v databázi

Záznam, či věta databáze je posloupnost bajtů bez jakékoliv definované struktury. Samotná databáze, ani palm, neurčují jejich význam. Je na programátorovi, aby jim ve své aplikaci přidělil význam a dal řád. Takových záznamů může být va databázi asi 64K. Velikost záznamu je limitována 64KB.

S každým zaznamem je spojena informace (hlavička záznamu) která obsahuje

Každý záznam má svůj index, přes který k němu přistupujeme. První záznam má index `0`.

15.8.4. Práce s databází s použitím *Data Manageru*

Pro vytvoření a zrušení databáze máme volání DmCreateDatabase a DmDeleteDatabase

Kapitola 16. Quartus Forth zevnitř

* *rcsinfo="$Header: /home/radek/cvs/forth-book/ch-qf-internals.xml,v 1.16 2005/10/20 05:33:42 radek Exp $"*

vtipný epigraf

Tato kapitola je o tom, jak funguje Quartus Forth zevnitř.

Tabulka 16-1. Použití registrů CPU

* ***:[1:1:5]***

Registr	Symbolické jméno	Obsah
A2	CS	*Codespace segment pointer* - bázová adresa segmentu s kódem
A4	SP	*Data stack pointer* - ukazatel na zásobník dat, ukazuje na NOS, TOS je v D7
A5	DS	*Dataspace segment pointer* - bázová adresa datového segmentu
A6	FP	*Frame pointer* - ukazatel na aktivační záznam procedury/funkce
A7	RP	*Return stack pointer* - ukazatel na zásobník návratových adres, tento je shodný se systémovým zásobníkem v A7 zvaným SP. Pozor, zde označujeme názvem SP datový zásobník!
D7	TOS	*Top of Stack* - vrchol zásobníku dat je udržován v registru D7 pro rychlejší práci se zásobníkem. Ukazatel zásobníku dat SP ukazuje na prvek pod vrcholem, NOS.

16.1. Mapa paměti

Mapa obsazení paměti

Tabulka 16-2. Mapa paměti

* ***:[1:1:1:6]***

adresa	hex	typ	popis
78	004E	?	window-bounds
86	0056	W	currentx
88	0058	W	currenty
94	005E	L	SOURCE
190	00BE	W	eventhandler
196	00C4	W	SOURCE-ID
280	0118	W	event
374	0176	W	STATE
376	0178	W	BASE
406	0196	W	>IN
420	01A4	W	proměnná HERE obsahuje adresu první volné buňky v datovém segmentu

16.2. Analýza Quartus.PRC version 1.2.1U

Příklad 16-1. Resource *code 0*

```
0000= 000000C0
0004= 00000000
```

Tabulka 16-3. Resource `code 0`

* ***:[1:2:6]***

offset	value	description
0	000000C0	initialized data size
4	00000000	uninitialized data size

16.2.1. Část *code 1*

Analyzovaný kód je rozdělený do kousků/úseků podle podprogramů či slov. Kód slov se pak nachází v části *Slovník ANSI forthu*, a zde jsou uvedeny jen počáteční a koncová adresa a odkaz na slovo do části *Slovník ANSI forthu*.

Quartus Forth je napsán v assembleru PILA a začátek je tedy *Startup* kód.

Příklad 16-2. Startovací kód, začátek programu

* ***[W:80]***

Začátek PRC souboru je standardní kód tak jak je doporučený v assembleru PILA. Je to kód přímo ze souboru *FIXME:`jméno souboru`. Dvojí adresa na začátku řádku je proto, že první adresa je adresa v PRC souboru a druhá adresa pak adresa na které se kód nachází z hlediska procesoru v době běhu programu.

```
;ADR  WORDS           LABEL        MNEMO   ARGS
;---  ---------  -----------  ------- -------------------------
                              ; Analýza Quartus Forth verze 1.2.1U
0000:                         .ORG    0         ; PRC začíná na adrese 0000.
                 __Startup__: PROC __Startup__()
                              LOCAL pappi.L
                              LOCAL prevGlobals.L
                              LOCAL globalsPtr.L
0000: 4E56 FFF4                 = LINK    FP, #-12   12 slabik pro lokální proměnné;
                              SysAppStartup(&pappi(A6), &prevGlobals(A6), &globalsPtr(a6))
0004: 486E FFF4                 = PEA     globalsPtr(FP)
0008: 486E FFF8                 = PEA     prevGlobals(FP)
000C: 486E FFFC                 = PEA     papi(FP)
0010: 4E4F A08F                 = SYSTRAP SysAppStartup
0014: 4FEF 000C               LEA     12(RP), RP    ; Odstranění parametrů ze zásobníku
0018: 4A40                    TST.W   D0               ; Test na výsledek SysAppStartup
001A: 670E                  .--BEQ.S   _SU1            ; +14=+$0E ⟶002A
                            |
                            |  ; V případě neúspěchu pípneme a ukončíme program
                            |  SndPlaySystemSound(#sndError.B)
001C: 1F3C 0003             |    = MOVE.B  #3, -(RP)
0020: 4E4F A234             |    = SYSTRAP SndPlaySystemSound
0024: 548F                  |    = ADDQ.L  #2, RP
0026: 70FF                  |  MOVEQ   #-1, D0
0028: 602A                .-|--BRA.S   _SUExit          ; Konec
                          | |
                          | |  ; Příprava argumentů a volání procedury PilotMain
002A: 206E FFFC  _SU1:    | '->MOVE.L  pappi(FP), A0
```

```
                          |    PilotMain(SysAppInfoType.cmd(A0).W,
                          |              SysAppInfoType.cmdPBP(A0).L,
                          |              SysAppInfoType.launchFlags(A0).W)
002E: 3F28 0006           |      = MOVE    6(A0), -(RP)
0032: 2F28 0002           |      = MOVE.L  2(A0), -(RP)
0036: 3F10                |      = MOVE    (A0), -(RP)
0038: 4EBA 001E           |      = JSR     PilotMain
003C: 508F                |      = ADDQ.L  #8, RP
                          |    SysAppExit(appi(A6).L, prefGlobal(A6).L, globalsPtr(A6).L)
003E: 2F2E FFF4           |      = MOVE.L  globalsPtr(FP), -(RP)
0042: 2F2E FFF8           |      = MOVE.L  prefGlobal(FP), -(RP)
0046: 2F2E FFFC           |      = MOVE.L  appi(FP), -(RP)
004A: 4E4F A090           |      = SYSTRAP SysAppExit
004E: 4FEF 000C           |      = LEA     12(RP), RP
0052: 7000                |    MOVEQ.W #0, D0
0054:             _SUExit:‘--->ENDPROC                    ; Return to PalmOS
0054: 4E5E                       = UNLK    FP
0056: 4E75                       = RTS
```

Příklad 16-3. Funkce `PilotMain` *:[95]

Toto je vlastní funkce main, přesněji PilotMain která se volá ze startup kódu uvedeného výše.

```
                                 ; PROC PilotMain (cmd.W, cmdPBP.L, launchFlags.W)
0058= 805A: 4E56 0000            PilotMain:   LINK    FP, #0           ;Rámec bez lokálních proměnných
005C= 805E: 4A6E 0008                         TST.W   8(FP)            ;ARG.
0060= 8062: 6638                           .--BNE     PmReturn         ; $+56 = 009A
0062= 8064: 48E7 3F7E                      |  MOVEM.L D2-D7/A1-A6, -(RP) ; Úschova registrů s použitím RP=A7
                                           |  ; ?Žádost o pamět’?
0066= 8068: 200F                           |  MOVE.L  RP, D0           ; Uložení ukazatele zásobníku
0068= 806A: 2B40 000E                      |  MOVE.L  D0, 14(DS)       ;+
                                           |  MemHandleNew(#2048L)
006C= 806E: 2F3C 0000 0800                 |      MOVE.L  #2048, -(RP)
0072= 8074: 4E4F A01E                      |      SYSTRAP MemHandleNew
0076= 8078: 588F                           |      ADDQ.L  #4, RP
                                           |  MemHandleLock(A0.L)
0078= 807A: 2F08                           |      MOVE.L  A0, -(RP)
007A= 807C: 4E4F A021                      |      SYSTRAP MemHandleLock
007E= 8080: 588F                           |      ADDQ.L  #4, RP
0080= 8082: 49E8 07FE                      |  LEA     2046(A0), SP     ;Nastavení datového zásobníku
0084= 8086: 2B4C 0012                      |  MOVE.L  SP, 18(DS)       ;Uložení SP do datového segmentu
                                        ; Mapování segmentu kódu
0088= 808A: 45FA 7F74                      |  LEA     $7FFE(PC), CS    ;Spočte adresu segmentu kódu a uloží ji do CS
008C= 808E: 2B4A 0026                      |  MOVE.L  CS, 38(DS)       ;Uložení CS do datového segmentu
0090= 8092: 610C                           |  BSR.S   INIT             ;=$80A0
0092= 8094: 2E6D 000E                      |  MOVE.L  14(DS), RP
0096= 8098: 4CDF 7EFC                      |  MOVEM.L (RP)+, D2-D7/A1-A6 ;Load saved registers from (RP)
009A= 809C: 4E5E                 PmReturn: ‘->UNLK    FP               ;Remove stack frame
009C= 809E: 4E75                           RTS
```

Příklad 16-4. Kód 009E - 00DC

*** *:[80]**

```
                                    ; Inicializece volaná z PilotMain
009E= 80A0:                         INIT:        FplInit()
009E= 80A0: 4E4F A0DF                              = SYSTRAP FplInit
00A2= 80A4: 3E3C 0212                            MOVE    #530, TOS
00A6= 80A8: 0647 0100                            ADDI    #256, TOS
00AA= 80AC: 3B47 00AE                            MOVE    TOS, 174(DS)      ;
00AE= 80B0: 4EAA 8142                            JSR     $8142(CS)         ; -32446(A2)
00B2= 80B4: 3E1C                                 MOVE    (SP)+, TOS        ;= DROP
00B4= 80B6: 4EAA 8AAA                            JSR     $8AAA(CS)        ;= MainForm
00B8= 80BA: 4E4F A296                            SYSTRAP GetCharCaselessVal...
00BC= 80BE: 2B48 0182                            MOVE.L  A0, 386(DS)
00C0= 80C2: 4E4F A294                            SYSTRAP GetCharSortValue
00C4= 80C6: 2B48 0186                            MOVE.L  A0, 390(DS)
00C8= 80CA: 42AD 01C8                            CLR.L   456(DS)
```

```
00CC= 80CE: 4EAA C0C6                       JSR     -16186(CS)
00D0= 80D2: 4E75                            RTS

00D2= 80D4: 3907                LBL_5:      MOVE    TOS, -(SP)        ;= 3
00D4= 80D6: 7E03                            MOVEQ   #3, TOS           ;+
00D6= 80D8: 4E75                            RTS

00D8= 80DA: 6100 FFF8           LBL_6:      BSR     LBL_5             ; $-8 = 00D2
   00DC= 80DE: 4E75                            RTS
```

Následující tabulka je seznam slov ze slovníku. Příslušný kód se nachází u daného slova v části *Slovník ANSI forthu*. Zde jsou jen uvedeny adresy začátku a konce kódu, jakožto i startovací adresa. Adresy jsou uvedeny nejdříve ve vztahu k PRC souboru a poté kde se skutečně nachází v paměti bráno jako posunutí (offset) k registru A2 (= CS). Adresy jsou rovněž brány za celé slova (16 bitů) nikilov za slabiky (8 bitů). Odtud 00DE-**00E6** a **00E8**-010E

Tabulka 16-4. Slovník *:[1:1:1:1:1:3]

začátek	konec	začátek	konec	start	slovo
00DE	00E6	80E0	80E8	80E8	noop
00E8	010E	80EA	8110	80F2	(bye)
0110	013E	8112	8140	8118	BYE

Příklad 16-5. Kód následující po BYE, je volán z inicializace instrukcí JSR na adrese 00AE (80B0)

***** ****:[72]***

```
                                ; ???
0140:                   ________:   MOVEQ   #0, D0
0142:                               MOVE    TOS, D0
0144:                               MOVE    #-1, TOS
0148:                               ADDI    #530, D0
                                    MemHandleNew(D0)
014C:                                 = MOVE.L  D0, -(RP)
014E:                                   SYSTRAP MemHandleNew
0152:                                   ADDQ.L  #4, RP          ;= UNLOOP
0154:                               MOVE.L  A0, A3
0156:                               MOVE.L  A0, 176(DS)
015A:                               CMPA.L  #0, A0
0160:                             .--BNE.S  $8168
0162:                             |  CLR.W   TOS
0164:                             |  RTS                        ;= EXIT
0166:                             ‘->MemHandleLock(A3)
0166:                                   MOVE.L  A3, -(RP)
0168:                                   SYSTRAP MemHandleLock
016C:                                   ADDQ.L  #4, RP
016E:                               MOVE.L  A0, A3
0170:                               MemMove(DS, A3, #192)
0170:                                 = MOVE.L  #192, -(RP)
0176:                                   MOVE.L  DS, -(RP)
0178:                                   MOVE.L  A3, -(RP)
017A:                                   SYSTRAP MemMove dstP, sP, numBytes
017E:                                   LEA     12(RP), RP
0182:                               MOVE.L  A3, DS
0184:                               MOVE.W  #532, 420(DS)
018A:                               RTS
```

Tabulka 16-5. Pokračování předešlé tabulky Slovník *:[1:1:1:1:1:3]

začátek	konec	začátek	konec	start	slovo
018C	019E	818E	81A0	8198	EXECUTE
01A0	01AE	81A2	81B0	81AA	STATE
01B0	01C6	81B2	81C8	81BC	SOURCE
01C8	01D4	81CA	81D6	81D0	>IN
01D6	01E4	81D8	81E6	81E0	HERE
01E6	01FC	81E8	81FE	81F8	window-bounds
01FE	020C	8200	820E	8208	event
020E	021C	8210	821E	8218	BASE
021E	0230	8220	8232	822C	currentx
0232	0244	8234	8246	8240	currenty
0246	0256	8248	8258	8254	SOURCE-ID
025A	0270	825C	8272	826C	eventhandler
0272	0286	8274	8288	8282	BlankFormID
0288	029E	828A	82A0	829A	TitledFormID
02A0	02B4	82A2	82B6	82B0	MainFormID
02B6	02CC	82B8	82CE	82C0	DEPTH
02CE	02DC	82D0	82DE	82D8	SWAP
02DE	02F8	82E0	82FA	82E8	2SWAP
02FA	030C	82FC	830E	8302	ROT
____	____	BE30	____	BE36	.S
____	____	969E	96AA	96A2	:

Příklad 16-6. Konec části *code 1*

```
362A= B62C: B604                              DW      $B604
362C= B62E: 0554 4852 4F57                    DB      5, "THROW"
3632= B634: 4A47                  "THROW":    TST.W   D7
3634= B636: 6606                            .-BNE     $+6
3636= B638:                                 | MOVE    (A4)+, D7
3638= B63A:                                 | BRA     $+44  ;=3666
363C= B63E:                                 '>TST.L   158(A5)
3640= B642:                                   BEQ     $+16  ;=3652
3642= B644:                                   MOVE.L  158(A5), A7
3646= B648:                                   MOVE.L  (A7)+, 158(A5)
364A= B64C:                                   MOVE    (A7)+, 372(A5)
364E= B650:                                   MOVE.L  (A7)+, A4
3650= B652:                                   BRA     $+20  ;=3666
3652= B654:                                  >CMP     #-257, D7
3656= B658:                                   BNE     $+4  ;=365C
3658= B65A:                                   JSR     -32526(A2)        ;= (bye)
365C= B65E:                                   TST     12(A5)
3660= B662:                                   BEQ     $+6  ;=3668
3662= B664:                                   JSR     -32526(A2)        ;= (bye)
3666= B668: 4E75                              RTS                          ;= EXIT
3668= B66A: BE7C FFFE                        >CMP     #-2, D7
366C= B66E: 6616                              BNE     $+22  ;=3684
```

```
366E= B670:                                    MOVE    (A4)+, D7
3670= B672:                                    MOVE    D7, -(A4)
3672= B674:                                    MOVE    484(A5), D7
3676= B678:                                    MOVE    D7, -(A4)
3678= B67A:                                    MOVE    482(A5), D7
367C= B67E:                                    JSR     -30644(A2)        ;=884C
3680= B682:                                    JSR     -18966(A2)
3684= B686:                                    CMP     #-1, D7
3688= B68A:                                  .-BNE     $+4  ;=368E
368A= B68C:                                  | JSR     -18966(A2)
368E= B690:                                  ‘>JSR     -30736(A2)
3692= B694:                                    CMP     #-259, D7
3696= B698:                                    BLT     $+144  ;=3728
369A= B69C:                                    CMP     #-58, D7
369E= B6A0:                                    BGE     $+8  ;=36A8
36A0=                                          CMP     #-256, D7
36A4=                                          BGT     $+130  ;=3728
36A8=                                          CMP     #0, D7
36AC=                                          BGE     $+122  ;=3728
36B0=                                          MOVE    #1, -(A7)
36B4=                                          MOVE.L  #1953002103, -(A7)
36BA=                                          SYSTRAP DmGetResource
36BE=                                          ADDQ.L  #6, A7
36C0=                                          CMPA.L  #0, A0
36C6=                                          BEQ     $+96  ;=3728
36C8=                                          MOVE.L  A0, -(A7)
36CA=                                          CMP     #-1, D7
36CE=                                          BEQ     $+10  ;=36DA
36D0=                                          CMP     #-2, D7
36D4=                                          BEQ     $+4   ;=36DA
36D6=                                          JSR     -20764(A2)
36DA=                                         >MOVE    D7, -(A4)
36DC=                                          MOVEQ   #63, D7
36DE=                                          JSR     -30536(A2)
36E2=                                          JSR     -30512(A2)
36E6=                                          MOVE.L  (A7)+, A3
36E8=                                          MOVE.L  A3, -(A7)
36EA=                                          SYSTRAP MemHandleLock
36EE=                                          ADDQ.L  #4, A7
36F0=                                          JSR     -31508(A2)
36F4=                                          CMP     #58, D7
36F8=                                          BLE     $+4  ;=36FE
36FA=                                          ADDI    #-198, D7
36FE=                                          ASL     1, D7
3700=                                          MOVE    0(A0,D7), D1
3704=                                          LEA     0(A0,D1), A6
3708=                                          MOVE.L  A6, -(A7)
370A=                                          SYSTRAP StrLen
37E0=                                          ADDQ.L  #4, A7
3710=                                          MOVE.L  A6, A0
3712=                                          JSR     -30674(A2)
3716=                                          MOVE.L  A3, -(A7)
3718=                                          SYSTRAP MemHandleUnlock
371C=                                          ADDQ.L  #4, A7
371E=                                          MOVE.L  A3, -(A7)
3720=                                          SYSTRAP DmReleaseResource
3724=                                          ADDQ.L  #4, A7
```

```
3726=                                   BRA     $+76  ;=3774
3728=                                   MOVE    #10, 376(A5)
372E=                                   MOVE    D7, -(A7)
3730=                                   JSR     -31508(A2)
3734=                                   JSR     -31640(A2)
3738=                                   JSR     -25178(A2)
373C=                                   JSR     -31640(A2)
3740=                                   JSR     -25276(A2)
3744=                                   JSR     -25164(A2)
3748=                                   MOVE    (A7)+, D0
374A=                                   MOVE    D7, -(A4)
374C=                                   MOVE    D0, D7
374E=                                   JSR     -25252(A2)
3752=                                   JSR     -25198(A2)
3756=                                   MOVE    (A4)+, D7
3758=                                   MOVE    D7, D0
375A=                                   MOVE    (A4)+, D7
375C=                                   LEA     0(A5,D0), A0
3760=                                   MOVE.L  A0, -(A7)
3762=                                   MOVE.L  A0, -(A7)
3764=                                   MOVE.L  A0, -(A7)
3766=                                   MOVE    #1005, -(A7)
376A=                                   SYSTRAP FrmCustomAlert
376E=                                   LEA     14(A7), A7
3772=                                 .-BRA     $+4  ;=3778
3774=                                 | JSR     -29164(A2)
3778=                                 '>JSR     -18966(A2)
377C=                                   RTS


377E= B780: B62C                        DW      $B62C
3780= B782: 0541 424F 5054              DB      5, "ABORT"
3786= B788: 3907                "ABORT" MOVE    D7, -(A4)
3788= B78A:                             MOVE    #-1, D7
378C=                                   JSR     -18892(A2)
3790= B792:                             RTS                     ;= EXIT

3792=                                   MOVE    D7, 482(A5)
3796=                                   MOVE    (A4), 484(A5)
379A=                                   MOVE.L  (A4)+, D7
379C=                                   MOVE    D7, -(A4)
379E=                                   MOVEQ   #-2, D7
37A0=                                   JSR     -18892(A2)
37A4=                                   RTS

37A6= B7A8: B780                        DW      $B780
37A8= B7AA: 4641 424F 5254 2200         DB      6, 'ABORT"', 0
37B0= B7B2: 4EAA BC4A           ABORT"  JSR     -17334(A2)      ;= IF
37B4= B7B6: 4EAA 9A30                   JSR     -26064(A2)      ;= S"
37B8= B7BA:                             MOVE    D7, -(A4)       ;= DUP
37BA= B7BC:                             MOVE    #-18540, D7
37BE=                                   JSR     -26712(A2)      ;= COMPILE,
37C2=                                   JSR     -17298(A2)      ;= THEN
37C6=                                   RTS                     ;= EXIT

.
.
```

```
.


400A= C00C: BFF6                                DW      $BFF6
400C= C00E: 4728 666C 6F61 7429                 DB      $40+7, "(float)"
4014= C016: 3F2D 0178           "(float)"       MOVE    376(A5), -(A7)
4018=                                           MOVE    #10, 376(A5)
401E=                                           JSR     -20646(A2)        ;= parse-word
4022=                                           JSR     -22004(A2)        ;= >FLOAT
4026=                                           MOVE    D7, D0
4028=                                           MOVE    (A4)+, D7         ;= DROP
402A=                                           TST     D0
402C=                                         .-BNE     $+10  ;= 4038
402E=                                         | MOVE    D7, -(A4)         ;= DUP
4030=                                         | MOVE    #-46, D7
4034=                                         | JSR     -18892(A2)        ;= THROW
4038=                                         '>CMPI    #0,374(A5)
403E=                                         .-BEQ     $+4  ;=4044
4040=                                         | JSR     -26660(A2)        ;= LITERAL
4044=                                         '>MOVE    (A7)+, 376(A5)
4048=                                           RTS                       ;= EXIT


                                ; Následující slova slouží pro generování nového PRC souboru
                                ; a fungují jen v registrované verzi.

404A= C04C: C00C                                DW      $C00C
404C= C04E: 1067 656E 6572 6174                 DB      16, "generate-symbols", 0
4054= C056: 652D 7379 6D62 6F6C 7300
405E= C060:                     "generate-symbols":
405E= C060: 600A                                BRA     $+10  ;=406A

                                ; : MakePRC -259 THROW ;
4060= C062: C04C                                DW      $C04C
4062= C064: 074D 616B 6560 5243                 DB      7, "MakePRC"
406A= C06C: 3907                "MakePRC":  >MOVE    D7, -(A4)         ;=
406C= C06E: 3E3C FEFD                           MOVE    #-259, D7         ;  -259
4070= C072: 4EAA B634                           JSR     -18892(A2)        ;= THROW


4074= C076: C062                                DW      $C062
4076= C078: 0744 566C 5273 7263                 DB      7, DelRsrc
407E= C080: 60EA                                BRA     $-22  ;=406A


4080= C082: C076                                DW      $C076
4084= C086: 074E 5677 5273 7263                 DB      7, "NewRsrc"
408A= C08C: 60DE                "NewRsrc":   BRA     $-34  ;=406A


408C= C08E: C082                                DW      $C082
408E= C090: 0843 6F70 7952 7372 6300            DB      8, "CopyRsrc", 0
4098= C09A: 60D0                "CopyRsrc":  BRA     $-48  ;=406A


409A= C09C: C08E                                DW      $C08E
409C= C09E: 0A72 6567 6973 7465 7265 6400       DB      10, "registered", 0
```

```
40A8= C0AA: 60C0                "registered":BRA.S   $C06C  ;$-64 ;=406A


40AA= C0AC: 576F 726B 7370 6163 6500          DB      "Workspace", 0
40B4= C0B6: 486A                "Workspace": PEA     -16212(A2)
40B8= C0BA:                                  MOVE    #0, -(A7)
40BC= C0BE:                                  SYSTRAP DmFindDatabase
40C0= C0C2:                                  ADDQ.L  #6, A7
40C2= C0C4:                                  RTS


40C4= C0C6: 246D 0026                        MOVE.L  38(A5), A2
40C8= C0CA: 41EA C306                        LEA     -15610(A2), A0
40CC= C0CF: 303C 0025                        MOVE    #37, D0
40D0= C0D2: 4EAA 882E                        JSR     -30674(A2)
40D4= C0D6: 4EAA C0B6                        JSR     -16202(A2)
40D8= C0DA: 4A80                             TST.L   D0
40DA= C0DC: 670C                           .-<BEQ     $+12   ;=40E8
40DC= C0DE: 2F00                           |  MOVE.L  D0, -(A7)
40DE= C0E0: 3F3C 0000                      |  MOVE    #0, -(A7)
40E2= C0E4: 4E4F A042                      |  SYSTRAP DmDeleteDatabase...
40E6= C0E8: 5C8F                           '->ADDQ.L  #6, A7
40E8= C0EA: 3F3C 0000                        MOVE    #0, (A7)
40EC= C0EF: 2F3C 576F 726B                   MOVE.L  #'Work', -(A7)
40F2= C0F4: 2F3C 7034 7072                   MOVE.L  #'p4pr', -(A7)
40F8= C0FA: 486A C0AC                        PEA     -16212(A2)
40FC= C0FE: 3F3C 0000                        MOVE    #0, -(A7)
4100= C102: 4E4F A041                        SYSTRAP DmCreateDatabase
4104= C106: 4FEF 0010                        LEA     16(A7), A7
4108= C10A: 4A40                             TST     D0
410A= C10C: 6704                           .-<BEQ     $+4  ;=4110
410C= C10E: 4EAA 80F2                      |  JSR     -32526(A2)
4110= C112: 4EAA C0B6                      '->JSR     -16202(A2)
4114= C116: 3F3C 0003                        MOVE    #3, -(A7)
4118= C11A: 2F00                             MOVE.L  D0, -(A7)
411A= C11C: 3F3C 0000                        MOVE    #0, -(A7)
411E= C120: 4E4F A049                        SYSTRAP DmOpenDatabase
4122= C124: 508F                             ADDQ.L  #8, A7
4124= C126: 2B48 0048                        MOVE.L  A0, 72(A5)
4128= C12A: 2F3C 0000 C428                   MOVE.L  #50216, -(A7)
412E= C130: 486D 002A                        PEA     42(A5)
4132= C134: 2F08                             MOVE.L  A0, -(A7)
4134= C136: 4E4F A055                        SYSTRAP DmNewRecord
4138= C13A: 4FEF 000C                        LEA     12(A7), A7
413C= C13E: 2B48 0016                        MOVE.L  A0, 22(A5)
4140= C142: 2F08                             MOVE.L  A0, -(A7)
4142= C144: 4E4F A021                        SYSTRAP MemHandleLock
4146= C148: 588F                             ADDQ.L  #4, A7
4148= C14A: 2648                             MOVE.L  A0, A3
414A= C14C: 2B48 001A                        MOVE.L  A0, 26(A5)
414E= C150: 2F3C 0000 4428                   MOVE.L  #17448, -(A7)
4154= C156: 486A 8002                        PEA     -32766(A2)
4158= C15A: 2F3C 0000 0000                   MOVE.L  #0, -(A7)
415E= C160: 2F08                             MOVE.L  A0, -(A7)
4160= C162: 4E4F A076                        SYSTRAP DmWrite
4164= C166: 4FEF 0010                        LEA     16(A7), A7
4168= C16A: 2217                             MOVE.L  (A7), D1
```

```
416A= C16C: 928A                    SUB.L   A2, D1
416C= C16E: 0681 0000 7FFE          ADDI.L  #32766, D1
4172= C174: D28B                    ADD.L   A3, D1
4174= C176: 2E81                    MOVE.L  D1, (A7)
4176= C178: 45EB 7FFE               LEA     32766(A3), A2
417A= C17C: 2B4A 0022               MOVE.L  A2, 34(A5)
417E= C180: 3B7C C42A 01A6          MOVE    #-15318, 422(A5)
4184= C186: 3B7C 0214 01A4          MOVE    #532, 420(A5)
418A= C18C: 4EAA 9406               JSR     -27642(A2)
418E= C190: 41ED 0174               LEA     372(A5), A0
4192= C194: 91CD                    SUBA.L  A5, A0
4194= C196: 3B48 0174               MOVE    A0, 372(A5)
4198= C19A: 4EAA 86C6               JSR     -31034(A2)
419C= C19E: 3F3C 03ED               MOVE    #1005, -(A7)
41A0= C1A2: 2F3C 5462 6D70          MOVE.L  #'Tbmp', -(A7)
41A6= C1A8: 4E4F A05F               SYSTRAP DmGetResource
41AA= C1AC: 5C8F                    ADDQ.L  #6, A7
41AC= C1AE: 2648                    MOVE.L  A0, A3
41AE= C1B0: 2F08                    MOVE.L  A0, -(A7)
41B0= C1B2: 4E4F A021               SYSTRAP MemHandleLock
41B4= C1B6: 588F                    ADDQ.L  #4, A7
41B6= C1B8: 3F2D 0058               MOVE    88(A5), -(A7)
41BA= C1BC: 3F2D 0056               MOVE    86(A5), -(A7)
41BE= C1C0: 2F08                    MOVE.L  A0, -(A7)
41C0= C1C2: 4E4F A226               SYSTRAP WinDrawBitmap
41C4= C1C6: 508F                    ADDQ.L  #8, A7
41C6= C1C8: 2F0B                    MOVE.L  A3, -(A7)
41C8= C1CA: 4E4F A022               SYSTRAP MemHandleUnlock
41CC= C1CE: 588F                    ADDQ.L  #4, A7
41CE= C1D0: 2F0B                    MOVE.L  A3, -(A7)
41D0= C1D2: 4E4F A061               SYSTRAP DmReleaseResource...
41D4= C1D6: 588F                    ADDQ.L  #4, A7
41D6= C1D8: 3B7C 0021 0058          MOVE    #33,88(A5)
41DC= C1DE: 41EA C32B               LEA     -15573(A2), A0
41E0= C1E2: 303C 0020               MOVE    #32, D0
41E4= C1E6: 4EAA 882E               JSR     -30674(A2)
41E8= C1EA: 4EAA 8724               JSR     -30940(A2)
41EC= C1EE: 41EA C38D               LEA     -15475(A2), A0
41F0= C1F2: 303C 001C               MOVE    #28, D0
41F4= C1F6: 4EAA 882E               JSR     -30674(A2)
41F8= C1FA: 4EAA 8724               JSR     -30940(A2)
41FC= C1FE: 41EA C34B               LEA     -15541(A2), A0
4200= C202: 303C 001B               MOVE    #27, D0
4204= C206: 4EAA 882E               JSR     -30674(A2)
4208= C20A: 4EAA 8724               JSR     -30940(A2)
420C= C20E: 41EA C366               LEA     -15514(A2)
4210= C212: 303C 0014               MOVE    #20, D0
4214= C216: 4EAA 882E               JSR     -30674(A2)
4218= C21A: 4EAA 8724               JSR     -30940(A2)
421C= C21E: 4EAA 8724               JSR     -30940(A2)
4220= C222: 41EA C37A               LEA     -15494(A2), A0
4224= C226: 303C 0013               MOVE    #19, D0
4228= C22A: 4EAA 882E               JSR     -30674(A2)
422C= C22E: 4EAA 8724               JSR     -30940(A2)
4230= C232: 4EAA 8724               JSR     -30940(A2)
4234= C236: 2F3C 0000 0012          MOVE.L  #18, -(A7)
423A= C23C: 4E4F A013               SYSTRAP MemPtrNew
```

```
423E= C240: 588F                        ADDQ.L  #4, A7
4240= C242: 2648                        MOVE.L  A0, A3
4242= C244: 2B48 01F0                   MOVE.L  A0, 496(A5)
4246= C248: 486D 01F6                   PEA     502(A5)
424A= C24C: 486D 01F4                   PEA     500(A5)
424E= C250: 3F3C FFFF                   MOVE    #-1, -(A7)
4252= C254: 2F3C 6D65 6D6F              MOVE.L  #'memo', -(A7)
4258= C25A: 2F3C 6170 706C              MOVE.L  #'appl', -(A7)
425E= C260: 2F0B                        MOVE.L  A3, -(A7)
4260= C262: 3F3C FFFF                   MOVE.L  #-1, -(A7)
4264= C266: 4E4F A078                   SYSTRAP DmGetNextDataba...
4268= C26A: 4FEF 0018                   LEA     24(A7), (A7)
426C= C26E: 3F3C 0000                   MOVE.W  #0, -(A7)
4270= C272: 2F3C 0000 0012              MOVE.L  #18, -(A7)
4276= C278: 2F0B                        MOVE.L  A3, -(A7)
4278= C27A: 4E4F A027                   SYSTRAP MemSet
427C= C27E: 4FEF 000A                   LEA     10(A7), A7
4280= C282: 202D 01F6                   MOVE.L  502(A5), D0
4284= C286: 2740 0004                   MOVE.L  D0, 4(A3)
4288= C28A: 377C FFFF 0008              MOVE    #-1, 8(A3)
428E= C290: 3B7C FFFF 01C6              MOVE    #-1, 454(A5)
4294= C296: 42AD 01C2                   CLR.L   450(A5)
4298= C29A: 4EAA B594                   JSR     -19052(A2)
429C= C29E: 3907                        MOVE    D7, -(A4)
429E= C2A0: 3E3C 0066                   MOVE    #102, D7
42A2= C2A4: 3907                        MOVE    D7, -(A4)
42A4= C2A6: 7E0F                        MOVEQ   #15, D7
42A6= C2A8: 3907                        MOVE    D7, -(A4)
42A8= C2AA: 3E3C B930                   MOVE    #-18128, D7
42AC= C2AE: 4EAA B60C                   JSR     -18932(A2)
42B0= C2B2: 4A47                        TST     D7
42B2= C2B4: 670E                    .---BEQ     $+14  ;=42C2
42B4= C2B6: BE7C FFDA               |   CMP     #-38, D7
42B8= C2BA: 6704                    | .-BEQ     $+4   ;=42BE
42BA= C2BC: 4EAA B634               | | JSR     -18892(A2)
42BE= C2C0: 548C                    | '>ADDQ.L  #2, A4
42C0= C2C2: 3E1C                    |   MOVE    (A4)+, D7
42C2= C2C4: 3E1C                    '-->MOVE    (A4)+, D7
42C4= C2C6: 426D 019C                   CLR     412(A5)
42C8= C2CA: 4EAA B104                   JSR     -20220(A2)
42CC= C2CE: 4E75                        RTS                                 ;=EXIT

                                ;******************
                                ; Cold Start
                                ; ==========
                                        ; cold ( ⟶ )
42CE= C2D0: C09C                        DW      $C09C
42D0= C2D2: 0463 6F6C 6400      "cold":     DB      4, "cold"
42D6= C2D8: 486D 017A                   PEA     378(DS)
42DA= C2DC: 486D 01BA                   PEA     442(DS)
42DE= C2E0: 4E4F A0AC                   SYSTRAP SysCurAppDatabase..
42E2= C2E4: 508F                        ADDQ.L  #8, RP
42E4= C2E6: 2F3C 0000 0000              MOVE.L  #0, -(RP)
42EA= C2EC: 3F3C 0000                   MOVE.W  #0, -(RP)
42EE= C2F0: 2F2D 017A                   MOVE.L  378(DS), -(RP)
42F2= C2F4: 3F2D 01BA                   MOVE.W  442(DS), -(RP)
42F6= C2F8: 4E4F A0A7                   SYSTRAP SysUIAppSwitch
```

```
42FA= C2FC: 4FEF 000C                    LEA     12(RP), RP
42FE= C300: 4EAA 8118                    JSR     -32488(CS)        ;= BYE
4302= C304: 4E75                         RTS

4304= C306: 5175                         DW      $5175
4306= C308:                              DB      "Quartus Forth initializing workspace"
4328= C32A:                              DB      $85, "Welcome to Quartus Forth 1.2.1U.)
                                         DB      "Neal Bridges, 1998, 1999.All rights res
                                         DB      "Evaluation version.Build: 1999.03.06  2
43A8= C3AA: 8000 8000 8000 8000          DW      $8000, $8000, $8000, $8000
                                         ...
4420= C422: 8000 8000 8000 8000          DW      $8000, $8000, $8000, $8000


                                         MOVEQ.L #2, D7
                                         JSR     -12232(A2)     ;= approximate
                                         BRA.L   63154
....= F6A2: 4EAA DADA                    JSR     -9510(A2)      ;= .source
      F6A6: 4EAA DB4E                    JSR     -24B2(A2)      ;= ,dreg
      F6AA: 3907                         MOVE.W  D7, -(A4)      ;= DUP
      F6AC: 7E04                         MOVEQ.L #4, D7
      F6AE: 3907                         MOVE.W  D7, -(A4)      ;= DUP
      F6B0:                              MOVEQ.L #2, D7
      F6B2:                              JSR     $-2A92(A2)     ;= +if-long
      F6B6:                              RTS                    ;= EXIT

      F6B8: 1234 0000 EBE6 0B69
```

16.2.2. Část *imag 1000*

Příklad 16-7. Resource `imag 1000`

```
00B4= 0000:  LINK    A6, #-12
00B8= 0004:   PEA -12(A6)
00BC= 0008:   PEA -8(A6)
00C0=         PEA -4(A6)
00C4=         SYSTRAP SysAppStartup
00C8=         LEA  12(A7), A7
00CC=         TST  D0
00CE=         BEQ  $+14
00D0=         MOVE.B  #3, -(A7)
00D4=         SYSTRAP SndPlaySystemSou
00D8=         ADDQ.L  #2, A7
00DA=         MOVEQ   #-1, D0
00DC=         BRA     $+42
00DE=        >MOVE.L  -4(A6), A0
00E2=         MOVE    6(A0), -(A7)
00E6=         MOVE.L  2(A0), -(A7)
00EA=         MOVE    (A0), -(A7)
00EC=         JSR     30(PC)
00F0=         ADDQ.L  #8, A7
00F2=         MOVE.L  -12(A6), -(A7)
00F6=         MOVE.L  -8(A6), -(A7)
00FA=         MOVE.L  -4(A6), -(A7)
00FE=         SYSTRAP SysAppExit
```

```
0102=         LEA     12(A7), A7
0106=         MOVEQ   #0, D0
0108=        >UNLK    A6
010A=         RTS

010C=         LINK    A6, #0
0110=         TST     8(A6)
0114=         BEQ     $+6
0116=         BRA     $+252
011A=         D
011C=        >MOVEM.L D2-D7/A1-A6, -(A7)
0120=         MOVE    #1, -(A7)
0124=         MOVE.L  #1145132097, -(A7)
012A=         SYSTRAP DmGet1Resource
012E=         ADDQ.L  #6, A7
0130=         CMPA.L  #0, A0
0136=         BEQ     $+216
013A=         MOVE.L  A0, A3
013C=         MOVE.L  A0, -(A7)
013E=         SYSTRAP MemHandleSize
0142=         ADDQ.L  #4, A7
0144=         MOVE.L  D0, D7
0146=         MOVE.L  A3, -(A7)
0148=         SYSTRAP MemHandleLock
014C=         ADDQ.L  #4, A7
014E=         MOVE.L  A0, D4
0150=         MOVE.L  D7, D0
0152=         ADDI.L  #40, D0
0158=         MOVE.L  D0, -(A7)
015A=         SYSTRAP MemHandleNew
015E=         ADDQ.L  #4, A7
0160=         MOVE.L  A0,-(A7)
0162=         SYSTRAP MemHandleLock
0166=         ADDQ.L  #4, A7
0168=         MOVE.L  A5, D3
016A=         MOVE.L  A0, A5
016C=         MOVE.L  D7, -(A7)
016E=         MOVE.L  D4, -(A7)
0170=         MOVE.L  A5, -(A7)
0172=         SYSTRAP MemMove
0176=         LEA     12(A7), A7
017A=         MOVE.L  D3, A0
```

16.2.3. Analýza aplikace clock.prc

Příklad 16-8. Disassemblovaný kód aplikace clock.prc

```
0000= 4E56 FFF4               LINK    FP, #-12
0004= 486E                    PEA     -12(FP)
0008=                         PEA     -8(FP)
000C=                         PEA     -4(FP)
0010=                         SYSTRAP SysAppStartup
0014=                         LEA     12(RP), RP
0018=                         TST     D0
001A=                         BEQ     $+14 = 002A
```

```
001C=                           MOVE.B  #3, -(RP)
0020=                           SYSTRAP SndPlaySystemSound
0024=                           ADDQ.L  #2, RP
0026=                           MOVEQ   #-1, D0
0028=                           BRA     $+42 = 0054
002A=                           MOVE.L  -4(FP), A0
002E=                           MOVE    6(A0), -(RP)
0032=                           MOVE.L  2(A0), -(RP)
0036=                           MOVE    (A0), -(RP)
0038=                           JSR     30(PC)          ;= 0058
003C=                           ADDQ.L  #8, RP
003E=                           MOVE.L  -12(FP), -(RP)
0042=                           MOVE.L  -8(FP), -(RP)
0046=                           MOVE.L  -4(FP), -(RP)
004A=                           SYSTRAP SysAppExit

004E=                           LEA     12(RP), RP
0052=                           MOVEQ   #0, D0
0054=                           UNLK    FP
0056=                           RTS

0058=                           LINK    FP, #0
005C=                           TST     8(FP)
0060=                           BEQ     $+6             ;= 0068
0062=                           BRA     $+270           ;= 0172

0066=
0068=                LBL???:    MOVEM.L D2-D7/A1-A6, -(RP)
006C=                           MOVE    #1, -(RP)
0070=                           MOVE.L  #1145132097, -(RP)
0076=                           SYSTRAP DmGet1Resource
007A=                           ADDQ.L  #6, RP
007C=                           CMPA.L  #0, A0
0082=                           BEQ     $+234           ;= 016E
0086=                           MOVE.L  A0, A3
0088=                           MOVE.L  A0, -(RP)
008A=                           SYSTRAP MemHandleSize
008E=                           ADDQ.L  #4, RP
0090=                           MOVE.L  D0, TOS
0092=                           MOVE.L  A3, -(RP)
0094=                           SYSTRAP MemHandleLock
0098=                           ADDQ.L  #4, RP
009A=                           MOVE.L  A0, D4
009C=                           MOVE.L  TOS, D0
009E=                           ADDI.L  #40, D0
00A4=                           MOVE.L  D0, -(RP)
00A6=                           SYSTRAP MemHandleNew
00AA=                           ADDQ.L  #4, RP
00AC=                           MOVE.L  A0, -(RP)
00AE=                           SYSTRAP MemHandleLock
00B2=                           ADDQ.L  #4, RP
00B4=                           MOVE.L  DS, D3
00B6=                           MOVE.L  A0, DS
00B8=                           MOVE.L  TOS, -(RP)
00BA=                           MOVE.L  D4, -(RP)
00BC=                           MOVE.L  DS, -(RP)
00BE=                           SYSTRAP MemMove
```

```
00C2=                       LEA     12(RP), RP
00C6=                       MOVE.L  D3, A0
00C8=                       MOVE.L  (A0), (DS)
00CA=                       MOVE.L  4(A0), 4(SP)
00D0=                       MOVE.L  A3, -(RP)
00D2=                       SYSTRAP MemHandleUnlock
00D6=                       ADDQ.L  #4, RP
00D8=                       MOVE.L  A3, -(RP)
00DA=                       SYSTRAP DmReleaseResource
00DE=                       ADDQ.L  #4, RP
00E0=                       MOVE.L  RP, 14(DS)
00E4=                       MOVE.L  #2048, -(RP)
00EA=                       SYSTRAP MemHandleNew
00EE=                       ADDQ.L  #4, RP
00F0=                       MOVE.L  A0, -(RP)
00F2=                       SYSTRAP MemHandleLock
00F6=                       ADDQ.L  #4, RP
00F8=                       LEA     2046(A0), SP
00FC=                       MOVE.L  SP, 18(DS)
0100=                       MOVE    #1, -(RP)
0104=                       MOVE.L  #1882481008, -(RP)
010A=                       SYSTRAP DmGet1Resource
010E=                       ADDQ.L  #6, RP
0110=                       MOVE.L  A0, 22(DS)
0114=                       CMPA.L  #0, A0
011A=                       BEQ     $+58            ;= 0156
011C=                       MOVE.L  A0, -(RP)
011E=                       SYSTRAP MemHandleLock
0122=                       ADQ.L   #4, RP
0124=                       MOVE.L  A0, 26(DS)
0128=                       LEA     44(PC), A1      ;= 0156
012C=                       MOVE.L  A1, 30(DS)
0130=                       LEA     32766(A0), CS
0134=                       MOVE.L  CS, 34(DS)
0138=                       MOVE    8(FP), 38(DS)
013E=                       MOVE.L  10(FP), 40(DS)
0144=                       MOVE    14(FP), 44(DS)
014A=                       SYSTRAP FplInit
014E=                       MOVE    12(DS), D0
0152=                       JSR     0(CS, D0)
0156=               LBL???: MOVE.L  22(DS), -(RP)
015A=                       SYSTRAP MemHandleUnlock
015E=                       ADDQ.L  #4, RP
0160=                       MOVE.L  22,(DS), -(RP)
0164=                       SYSTRAP DmReleaseResource
0168=                       ADDQ.L  #4, RP
016A=                       MOVE.L  14(DS), RP
016E=               LBL???: MOVEM.L (RP)+, D2-D7/A1-A6
0172=               LBL???: UNLK    FP
0174=                       RTS
```

Příklad 16-9. Clock.PRC: p4ap 1

```
0000= 3907                      MOVE    TOS, -(SP)
0002= 3E3C                      MOVE    #948, TOS
0006=                           MOVE    (SP)+, 0(DS,TOS)
000A=                           MOVE    (SP)+, TOS
000C=                           MOVE    TOS, -(SP)
000E=                           MOVE    #946, TOS
0012=                           MOVE    (SP)+, 0(DS,TOS)
0016=                           MOVE    (SP)+, TOS
0018=                           RTS

001A=                           MOVE    TOS, -(SP)
001C=                           MOVEQ   #81, TOS
001E=                           MOVE    TOS, -(SP)
0020=                           MOVEQ   #80, TOS
0022=                           JMP     -32766(CS)

0026=                           MOVE    TOS, -(SP)
0028=                           MOVEQ   #-1, TOS
002A=                           MOVE    TOS, -(SP)
002C=                           MOVE    #952, TOS
0030=                           MOVE    (SP)+, 0(DS,TOS)
0034=                           MOVE    (SP)+, TOS
0036=                           RTS

0038=                           MOVE    TOS, -(SP)
003A=                           MOVE    A0, -(SP)
003C=                           MOVE.L  A0, TOS
003E=                           SWAP    TOS
0040=                           RTS

0042=                           MOVE.L  (SP), D1
0044=                           MOVE.L  D1, 0(DS,TOS)
0048=                           ADDQ.L  #4, SP
004A=                           MOVE    (SP)+, TOS
004C=                           RTS

004E=                           MOVE    TOS, -2(SP)
0052=                           MOVE.L  SP, D0
0054=                           SUBQ.L  #2, D0
0056=                           MOVE    TOS, -(SP)
0058=                           MOVE.L  D0, TOS
005A=                           MOVE    TOS, -(SP)
005C=                           SWAP    TOS
005E=                           RTS


...
...
...


0F9C=                           MOVE    TOS, -(SP)
0F9E=                           MOVE    #10, -(SP)
0FA2=                           MOVE    #0, TOS
0FA6=                           JSR     -32034(CS)
```

```
0FAA=                           MOVE    TOS, -(SP)
0FAC=                           MOVEQ   #6, TOS
0FAE=                           MOVE    TOS, -(SP)
0FB0=                           MOVE    2(SP), TOS
0FB4=                           CMP     (SP)+, TOS
0FB6=                           SEQ     TOS
0FB8=                           EXT     TOS
0FBA=                           MOVE    TOS, D0
0FBC=                           MOVE    (SP)+, TOS
0FBE=                           TST     D0
0FC0=                           BEQ     $+12            ;= 0FCE
0FC4=                           MOVE    (SP)+, TOS
0FC6=                           JSR     -32008(CS)
0FCA=                           BRA     $+76            ;= 1018
0FCE=                           MOVE    TOS, -(SP)
0FD0=                           MOVEQ   #5, TOS
0FD2=                           MOVE    TOS, -(SP)
0FD4=                           MOVE    2(SP), TOS
0FD8=                           CMP     (SP)+, TOS

...

105E=                           JSR     -28818(CS)
1062=                           BRA     $-200           ;= 0F9C
1066=                           RTS

1068=                           JSR     -32494(CS)
106C=                           MOVE    TOS, -(SP)
106E=                           MOVEQ   #100, TOS
1070=                           JSR     -32078(CS)
1074=                           JMP     -28770(CS)
```

16.3. Analýza Quartus.PRC version 2.0.0U

PRC soubro sestává z několika částí. Zde si některé rozebereme.

16.3.1. Část code1

```
;
0000: 4E56 FFF4
0004: 486E FFF4
0008: 486E FFF8
000C: 486E FFFC
0010: 4E4F A08F
0014: 4FEF 000C
0018: 4A40
001A: 670E

001C: 1F3C 0003
```

Kapitola 17. Moduly

Moduly pro QuartusForth

Vybrané moduly pro QuartusForth.

* *Každá sekce s modulem má `id` tvaru module:`název_modulu`*

Modulem se rozumí soubor, nebo skupina souborů s definicemi slov jenž řeší nějaký ucelený problém.

17.1. DataMgr

* *rcsinfo="$Header: /home/radek/cvs/forth-book/db-qf-moduly/DataMgr,v 1.6 2005/10/20 05:33:42 radek Exp $"*

Slova v modulu definovaná:dmModeReadOnly, dmModeWrite, dmModeReadWrite, dmModeLeaveOpen, dmModeExclusive, dmModeShowSecret, cardnum, UseCard, OpenDB, CloseDB, CreateDB

V modulu jsou definovány následující konstanty:

`dmModeReadOnly`

FIXME: doplnit

`dmModeWrite`

FIXME: doplnit

`dmModeReadWrite`

FIXME: doplnit

`dmModeLeaveOpen`

FIXME: doplnit

`dmModeExclusive`

FIXME: doplnit

`dmModeShowSecret`

FIXME: doplnit

Tabulka 17-1. Slova v modulu `DataMgr` *:[1:3:4]

slovo	zásobník	popis
dmModeRead(Only)	(⟶)	
dmModeWrit(e	(⟶)	

slovo	zásobník	popis
dmModeReadWrite	(⟶)	
dmLeaveOpen	(⟶)	
dmExclusive	(⟶)	
dmShowSecret	(⟶)	
UseCard	(n ⟶)	n - číslo karty
OpenDB	(mode zaddr len ⟶ dbr.)	
CloseDB	(dbr. ⟶)	
CreateDB	(resDB? type. creator. &zname zlen ⟶)	

Příklad 17-1. Příklad použití docneeds

```
needs docinc
docneeds myapp
```

Dalším použitím může být například startování aplikace uložené v dokumentu z programu Scripts

```
needs docinc
DocInclude" můj dokument"
```

17.2. CASE

Tabulka 17-2. Slova v modulu CASE

slovo	zásobník	popis
case		
endcase		
of		
endof		

Příklad 17-2. Modul CASE

17.3. csdump

Modul je vystaven na SleeplessNightWiki (http://sleeples-night.com/cgi-bin/twiki/view/Main/CsdumpModule). Autorem modulu je Kris Johnson

Modul definuje funkci `csdump` pro „binární“ výpis definice slova.

Tabulka 17-3. Prototypy funkcí definovaných v modulu csdump

slovo	zásobník	popis
csdump	(xt n --)	Dump N cells (in hex format) from codespace starting at XT

Příklad 17-3. Příklad použití modulu csdump

```
needs csdump
' SWAP 4 csdump
```

Příklad 17-4. Modul csdump

```
\ csdump  2001/7/28 KDJ

\ Print as unsigned hex
: h. ( u -- )
  base @ hex swap u. base ! ;

\ Dump given number of cells from codespace
: csdump ( xt n -- )
  for
    cr dup u. dup cs@ h.
    cell+
  next drop ;
```

17.4. Disasm

Tabulka 17-4. Slova v modulu Disasm [1:2:6]

slovo	zásobník	popis
see	(word ⟶)	Disassembluj uvedené slovo. Vypíše okomentovaný strojový kód slova.
seecs	(xt ⟶)	
seebase	(a. ⟶)	

Module je uložen ve čtyřech souborech `disasm`, `disasm.part2`, `disasm.part3` a `disasm.part4`.

17.5. DocInc

* *$Header: /home/radek/cvs/forth-book/db-qf-moduly/docinc,v 1.4 2003/12/28 18:21:57 radek Exp $*

Volně podle OnDoModule (http:/sleepless-night.com/cgi-bin/twiki/view/Main/DocIncModule) na webu Sleepless-Night Wiki (http://kristopherjohnson.net/cgi-bin/twiki/view/Main/)

Modul **docinc** nabízí možnost požívat Doc soubory pro psaní programů. Standardně používané poznámky jsou limitovány velikostí 4kB. Tento limit je limitem programu MemoPad. Za pomoci tohoto modulu mohou býti naše zdrojové texty podstatně delší.

Tabulka 17-5. Slova v modulu `docinc` *:[1:2:4]

***** ***[1:3:4]***

slovo	zásobník	popis
docincluded	(c-addr u ⟶)	
docinclude	("jméno souboru" ⟶)	
docinclude"	("jméno souboru<uvozovky>" ⟶)	
docneeded	(c-addr u ⟶)	
docneeds	("jméno souboru" ⟶)	
docneeds"	("jméno souboru<uvozovky>" ⟶)	

Příklad 17-5. Příklad použití docneeds

```
needs docinc
docneeds myapp
```

Dalším použitím může být například startování aplikace uložené v dokumentu z programu Scripts

```
needs docinc
DocInclude" můj dokument"
```

17.6. ezUI

* *$Header: /home/radek/cvs/forth-book/db-qf-moduly/ezUI,v 1.5 2003/12/28 18:21:57 radek Exp $*

ezUI je modul a nástroje pro tvorbu grafického rozhraní aplikace na PalmOS.

Tabulka 17-6. Seznam souborů modulu ezUI

***** ***[1:3]***

soubor	obsah
`ezUIbase`	defunuje slova a hodnoty společné pro ostatní soubory modulu
ezUI	hlavni soubor, jádro modulu **ezUI**, definuje slova pro pole a ovládací prvky
ezDate	definuje slova pro výběr datumu
ezTime	definuje slova pro výběr času
arraymove	definuje slova pro posouvání bloku buňek ze zásobníku do paměti a zpět
wrapnum	funkce pro „wrapping“ čísla v specifikovaném páru limit.
string2anyfield	dovoluje přesouvání řetězců do polí jenž nejsou editovatelná

soubor	obsah
pushbuttons	definuje slova pro nastavování a získávání stavu tlačítek v množině tlačítek

Tabulka 17-7. Slova v modulu ezUI

* ***[1:3:6]***

slovo	zásobník	popis
stringfield	(GET ⟶ str u) (str u SET ⟶)	
intfield	(GET ⟶ n) (n SET ⟶)	
dblfield	(GET ⟶ n.) (n. SET ⟶)	
listfield	(GET ⟶ n) (n SET ⟶)	
checkbox	(GET ⟶ n) (n SET ⟶)	
button	( )	

Modul definuje nekolik typů prvků rozhraní. Všechny se definují podle voru

```
číslo jméno_typu jméno
```

tedy například řádek

```
1001 stringfield ClientName
```

definuje prvek s identifikačním číslem 1010 jako textové pole pojmenovane `ClientName` Definovane prvky jsou objekty kterým je možno zasílat metody. Všem ovládacím prvkm je možno zaslat metody

hide

schová, zneviditelní, ovládací prvek

show

opět zviditelní, ovládací prvek

getid

vrátí identifikační číslo **id** ovládacího prveku

Varování

Volání metody ovládacího prvku který není na aktuálním formuláři způsobí chybu PalmOS

Tip: Metodu `getid` můžeme použít v době běhu ... . Například zjištění zda-li číslo v TOS je shodné s id ovládacího prvku „mycontrol“ můžeme udělat takto:

```
[ getid mycontrol ] = if
```

slova v hranatých závorkách jsou vykonána v době kompilace a hodnota takto zjištěná je uložena do programu. Tedy je-li id „mycontrol“ 1020, uvedený příklad se zkompiluje stejně jako kód

```
1020 = if
```

Pro všechny typy polí mohou být použity metody:

`get`
: vrátí hodnotu v poli

`set`
: nastavi hodnotu v poli

dirty?
: vrátí 1 je-li pole dirty, jinak vrátí 0

dirty
: nastaví status dirty na danou hodnotu

Příklad 17-6. Modul Šablona

```
...
```

17.6.1. string2anyfield

Tabulka 17-8. Slova v modulu string2anyfield

slovo	zásobník	popis
string>Handle	(str u ⟶ handle.)	
freeHandle	(handle. ⟶)	
handle>Field	(handle. ⟶)	
string>anyField	(str u fieldID ⟶)	

17.7. OnDo

* *$Header: /home/radek/cvs/forth-book/db-qf-moduly/ondo,v 1.4 2003/12/28 18:21:57 radek Exp $*

Volně podle OnDoModule (http:/sleepless-night.com/cgi-bin/twiki/view/Main/OnDoModule) na webu Sleepless-Night Wiki (http://kristopherjohnson.net/cgi-bin/twiki/view/Main/)

Modul **ondo** nabízí podobnou funkcionalitu jako **case** nebo **cond..thens**. Má ovšem čistší syntaxi a je lépe optimalizován v QuartusForthu.

Tabulka 17-9. Slova v modulu OnDo [1:2:7]

slovo	zásobník	popis

slovo	zásobník	popis
on:	(x "word" ⟶ x)	porovnej TOS s hodnotou následujícího slova
or:	(x "word" ⟶ x)	je-li shoda, skoč na **do:**, jinak pokračuj srovnáním z dalším slovem
do:	("word" ⟶)	je-li shoda, vykonej následující slovo a potom **exit;** jina pokračuj

Věta

```
on: SOME-VALUE do: SOME-WORD
```

je ekvivalentní větě

```
dup SOME-VALUE = if SOME-WORD exit then
```

Mezi **on:** a **do:** smí být libovolné množství

```
or: SOME-VALUE
```

Poznámka: Konstrukce **on:..or:..do:** neodstraňuje nic ze zásobníku.

Příklad 17-7. Příklad použití on:..do:

```
needs Events
needs ondo

: DoCtlSelect ( -- ) ... ;
: DoMenu ( -- ) ... ;
: DoPenDown ( -- ) ... ;

: dispatch-event ( ekey -- ekey )
  on: ctlSelectEvent do: DoCtlSelect
  on: menuEvent do: DoMenu
  on: penDownEvent do: DoPenDown ;

: event-loop ( -- )
  begin
    ekey dispatch-event drop
  again ;
```

Příklad 17-8. Modul ondo

```
\ ondo  2001/8/23 KDJ
\ Provides ON:..DO: construct.
\ Copyright 2001 Kristopher D. Johnson
\
\ WARRANTY ...
\
\ USAGE ...
\
\ Relies upon undocumented features
```

```
\ of Quartus Forth 1.2.x; may not
\ be compatible with future releases.

needs condthens

\ M68K opcodes
(hex) be7c constant cmp#,tos
(hex) 6600 constant bne.w
(hex) 6700 constant beq.w

\ Compile conditional branch,
\ leaving ORIG on stack for later
\ resolution by ELSE or THEN
: (bcc-orig) ( op ⟶ ) ( C: ⟶ orig )
  cs, cshere 0 cs, ;

: (eval-word) ( "word" ⟶ i**x )
  parse-word evaluate ;

: (on:) ( x "word" ⟶ x )
  cmp#,tos cs,
  postpone [ (eval-word) postpone ]
  cs, ;

\ Compare top-of-stack with
\ value of following word
: on: ( x "word" ⟶ x )
  postpone cond
  (on:)
; immediate

\ If EQ, branch ahead to DO:,
\ else compare TOS with neot word
: or: ( x "word" ⟶ x )
  beq.w (bcc-orig)
  (on:)
; immediate

\ If EQ, jump to NAME, otherwise
\ branch over NAME
: do: ( "name" ⟶ )
  bne.w (bcc-orig) >r
  postpone thens
  (eval-word)
  postpone exit
  r> postpone then
; immediate

\ drop do:
: ddo: ( "name" ⟶ )
        bne.w (bcc-orig) >r
        postpone thens
        postpone drop
        (eval-word)
        postpone exit
        r> postpone then
; immediate
```

Kapitola 18. Kalkulačka

* *rcsinfo="$Header: /home/radek/cvs/forth-book/ch-Kalkulacka.xml,v 1.5 2005/10/20 05:33:42 radek Exp $"*

ftipný apygraf

Text kapitoly

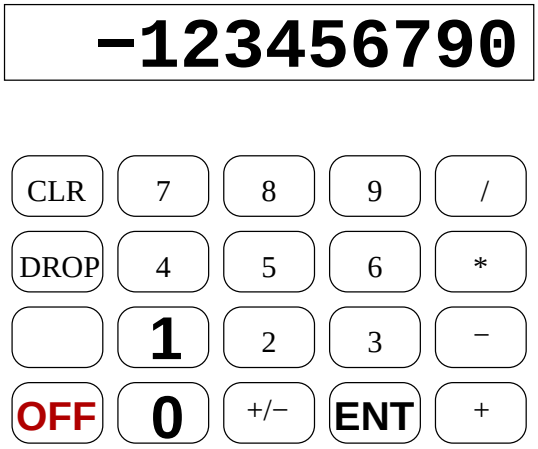
18.1. První varianta

18.1.1. Zadání

Jednoduchá kalkulačka. Pracujeme s celými čísly ve rozlišení dvě slova, tedy 32 bitů. T.j. číselný rozsah je -2147483648 až 2147483647. Kalkulačka je typu RPN.

18.1.2. Poznámky

Obrázek 18-1. Grafický návrh formuláře



18.1.3. Realizace

Aplikace je pojmenována „Kalkulačka“

Createor: Kalk, Type: rsrc, ResourceDB

18.1.3.1. Formulář

Formulář `tFRM 1000` obsahuje jedno pole displeje `Field 1010` a řadu tlačítek `Button ????`

Field 1010

top: 10, left: 10, width: 140, height: 16, Usable, Underline, SingleLine, Alignment: right, Font: LED, MaxChars: 16

Button 1100

Title: "0", top: 137, left: 39, width: 24, height: 20, Usable, Enabled, Style: button, Frame: standard, Font: LED

Button 1110

Title: "ENT", top: 137, left: 97, width: 24, height: 20, Usable, Enabled, Style: button, Frame: standard, Font: Large

Button 1111

Title: "+/-", top: 137, left: 68, width: 24, height: 20, Usable, Enabled, Style: button, Frame: standard, Font: Large

Button 1112

Title: "+", top: 137, left: 126, width: 24, height: 20, Usable, Enabled, Style: button, Frame: standard, Font: Large

Button 1113

Title: "-", top: 112, left: 126, width: 24, height: 20, Usable, Enabled, Style: button, Frame: standard, Font: Large

Button 1114

Title: "*", top: 87, left: 126, width: 24, height: 20, Usable, Enabled, Style: button, Frame: standard, Font: Large

Button 1115

Title: "/", top: 62, left: 126, width: 24, height: 20, Usable, Enabled, Style: button, Frame: standard, Font: Large

Button 1116

Title: "CLR", top: 62, left: 10, width: 24, height: 20, Usable, Enabled, Style: button, Frame: standard, Font: Large

Button 1117

Title: "DROP", top: 87, left: 10, width: 24, height: 20, Usable, Enabled, Style: button, Frame: standard, Font: Standard

Button 1118

Title: "OFF", top: 137, left: 10, width: 24, height: 20, Usable, Enabled, Style: button, Frame: standard, Font: Large

18.1.3.2. Program

Program je z důvodů jednoduchosti napsán v jedom souboru (jedné poznámce)

```
\ Kalkulačka
needs ids       ❶
needs Resources ❷
needs OnDo      ❸
needs Events    ❹
needs ezUI      ❺
```

```
needs tester

decimal
```

❶ Potřebujeme slovo (ID)

❷ Budeme pracovat se zdroji

❸ Používáme konstrukci `on: ... do: ...` z modulu OnDo.

❹ Potřebujeme symbolická jména událostí pro srozumitelnost kodu.

❺ Pro práci s polem zobrazovače **display** použijeme pro jednoduchost modul ezUI.

Teď si otevřeme zdrojovou databázi aplikace s formulářem

```
(ID) Kalk (ID) rsrs use-resources
```

A nadefinujeme si symbolické konstanty pro formulář a jednotlivá tlačítka.

```
1000 constant MainFormID
1010 dblField display
1116 constant btnClrID
1117 constant btnDropID
1118 constant btnOffID
```

Vytvoření proměnných.

```
create stack 20 cells allot  ❶
variable stackp  stack stackp !
```

❶ Vyhrazení 20 buněk paměnti pro zásobník naší kalkulačky. Takto veliký zásobník vystačí na 10 čísel.

Následující slova uvedu v obráceném pořadí, než jsou v souboru. Je to pro souvislost myšenky. Ve forthu totiž musí být slova při definici použitá definována předem.

Slovo **go** je spouštěcím slovem programu.

```
: go ( ⟶ )
       MainFormID ShowForm  ❶
       handle-events ;      ❷
```

❶ Otevření hlavního a jediného formuláře aplikace.

❷ Spuštění hlavní smyčky zpracování událostí.

Smyčka zpracování událostí. Kód je jednoduchý. V nekonečné smyčce `begin ... again` si v každém průchodu vyzvedneme slovem **ekey** událost z fronty, a tu pak procedurou **dispatch-event** zpracujeme. Nakonec je třeba událost ze zásobníku zahodit (**drop**) aby se nám na něm nehromadily.

```
: handle-events ( ⟶ )
       begin
               ekey dispatch-event drop
       again ;
```

Přidělování „dispatch“ události dalším častem je v našem případě velmi jednoduché. Zajímají nás jen událost stisku tlačítka, tedy ctlSelectEvent.

```
: dispatch-event ( ekey ⟶ ekey )
        on: ctlSelectEvent      do: cltSelect
        ;
```

Rozhodování které tlčítko bylo zmáčknuto.

```
: ctlSelect ( ekey ⟶ ekey )
        event >abs ItemID
        on: btnOffID    ddo: (bye)
        on: btnClrID    ddo: btnClr
        on: btnDropID   ddo: btnDrop
        ;
```

Zmáčknutí tlačítka CLR vymaže zobrazovač.

```
: btnClr ( ⟶ )
        0. SET display ;
```

Program je z důvodů organizace a snadné manipulace napsán do jediné poznámky (memo). První řádek, který je současně názvem poznámky

```
\ Kalkulačka
```

Import zdrojů z jiných souborů

```
needs ids
needs Resources
needs OnDo
needs Events
needs ezUI
needs tester
```

Pro jistotu se přepneme do dekadické číselné soustavy.

```
decimal
```

Otevření zdrojové databáze programu.

```
(ID) Kalk (ID) resr use-resource
```

Nadefinujeme si konstanty a prvky grafického rozhraní.

```
1000 constant MainFormID
1010 dblField display
1116 constant btnClrID
1117 constant btnDropID
1118 constant btnOffID
```

18.1.3.2.1. Zásobník

Modul zásobník sluší pro práci se zásobníkem dlouhých čísel. Je použit jako zásobník čísel naší kalkulačky. Bylo by vhodné vydělit tento modul do samostatného souboru, ale pro jednoduchost jej nechávám zde.

```
\ === Zásobník ===
.( Zásobník) cr
```

Datová struktura zásobníku se skládá ze dvou objektů. Pole dlouhých čísel

```
create stack 20 cells allot
```

A ukazatele do tohoto pole

```
variable stackp
```

Ukazatel hned inicializujem. Necháme ho ukazovat na dno našeho zásobníku

```
stack stackp !
```

Uložení čísla na zásobník.

```
: >stack ( n. ⟶ ) ( ⟶ n. )
        stackp @ 2!             \ ulož n. na zásbník
        stackp @ 4 + stackp ! ; \ posuň ukazatel dopředu
```

Vyjmutí čísla ze zásobníku

```
: stack> ( ⟶ n. ) ( n. ⟶ )
        stackp @ 4 - stackp !   \ vrat' ukazatel zpět
        stackp @ 2@ ;           \ vyjmi n. ze zásobníku
```

Právě nadefinovaná slova si hned otestujeme.

```
TESTING >stack stack>
{ 34567. >stack -> }
{ stack> -> 34567. }
{ 134167. 2. >stack >stack -> }
{ stack> stack> -> 134167. 2. }
```

Protože budeme potřebovat vybírat a ukládat čísla na zásobník v párech, nedefinujeme si pro to slova **2>stack** a 2stack>

```
: 2>stack ( n1. n2. ⟶ ) ( ⟶ n1. n2. )
        >stack >stack ;
: 2stack> ( ⟶ n1. n2. ) ( n1. n2. ⟶ )
        stack> stack> ;
```

A otestujeme si jestli jsou v pořádku

```
TESTING 2>stack 2stack>
{ 34567. 3. 2>stack -> }
{ 2stack> -> 34567. 3.}
```

Nyní již můžeme přistoupit k operacím nad naším zásobníkem. Začneme sčítáním.

```
: stack+ ( ⟶ ) ( n1. n2. ⟶ n. )
        2stack> D+ >stack;
```

A opět si je hned odzkoušíme

```
TESTING stack+
{ 1. 56. 2>stack stack+ stack> -> 57. }
```

18.1.3.2.2. Obsluha událostí

Obsluha událostí v četně reakcí na jednotlivá stalčená tlačítka.

```
: btnClr ( ⟶ ) ( ⟶ )
        0. SET display ;

: btnDrop ( ⟶ ) ( n. ⟶ )
        ;

: ctlSelect ( ekey ⟶ ekey )
        event >abs ItemID
        on: btnOffID    ddo: (bye)
        on: btnClrID    ddo: btnClr
        on: btnDropID   ddo: btnDrop
```

Přidělování události je v našem případě velmi jednoduché, protože nás zajímají jen události stisku tlačítek.

```
: dispatch-event ( ekey ⟶ ekey )
        on: ctlSelectEvent      do: ctlSelect
```

Obsluha událostí je nekonečná smyčka jenž vybírá (**ekey**) události a provádí jejich přidělení (**dispatch-event**)

```
: handle-events ( ⟶ )
        begin
                ekey dispatch-event drop
        again ;
```

18.1.3.2.3. Spuštění programu

Spuštění programu, tedy slovo **go** provede zobrazení hlavního formuláře a předá řízení obsluze událostí.

```
: go ( ⟶ )
        MainFormID ShowForm
        handle-events ;
```

A to je z našeho malého programu všechno.

Kapitola 19. KeyMaster

* *rcsinfo="$Header: /home/radek/cvs/forth-book/ch-KeyMaster.xml,v 1.3 2005/10/20 05:33:42 radek Exp $"*

ftipný apygraf

Text kapitoly

19.1. Zadání

Nejdříve si popíšeme co má aplikace dělat, abychom věděli kdy jsme s prací hotovi.

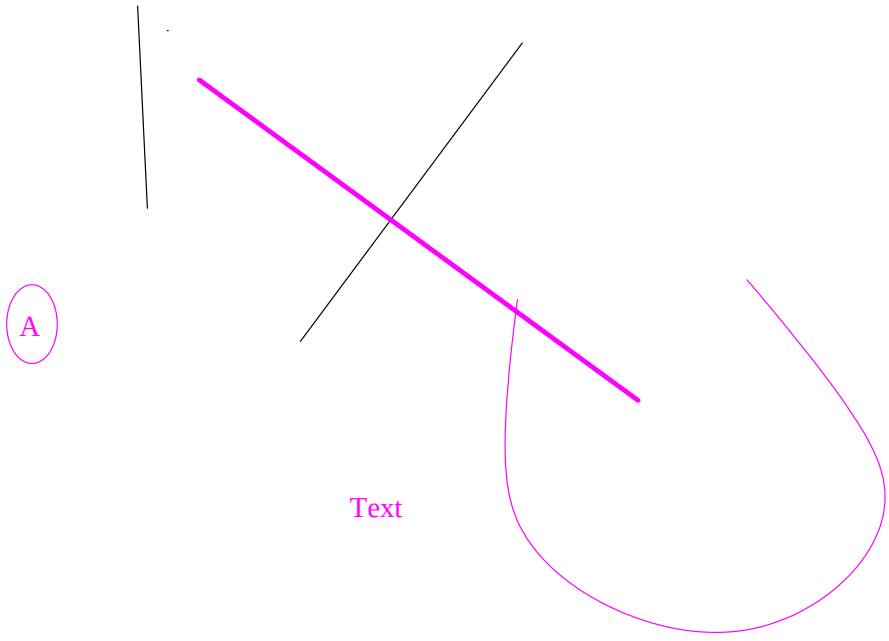
Aplikace bude sloužita jako kartotéka účtů. Ke každému účtu budeme mít poznamenán jeho název, datum změny hesla, heslo, ...

Tabulka 19-1. Struktura databáze programu KeyMaster

název	typ	popis
TITLE	C32	titulek, název kartičky s účtem
MDATE	Date	datum poslední změny hesla
ACCOUNT	C Var	název účtu
PASSWD	C Var	heslo k účtu
NOTE	Memo	poznámka

Formulář bude jen jeden, a to z pohledu karty.

Obrázek 19-1. Grafický návrh formuláře



V. PP Forth

V této části je popsána implementace jazyka Forth určená pro počítače Palm.

Kapitola 20. PPForth

* *rcsinfo="$Header: /home/radek/cvs/forth-book/ch-ppforth.xml,v 1.3 2005/10/20 05:33:42 radek Exp $"*

epigram

Text kapitoly

Seznam souborů

`HWRsrcA.prc`

?

`HWRsrcB.prc`

?

`asm68k.f.pdb`

?

`kernel.f.pdb`

?

`metaforth.f.pdb`

?

`palmeta.f.pdb`

?

`ppforth.prc`

?

`ppforthrsrc.prc`

?

`rsave.f.pdb`

?

`ui.f.pdb`

?

`view.f.pdb`

?

`vocs.f.pdb`

?

20.1. Sekce

.

20.2. ppforthrsc

.

name

ppforthrsc

Records

6

Size

652

Creator

PaiG

Type

rsrc

Copy Protect

NO

Backup

NO

ResourceDB

YES

Read only

NO

ppforthrsc Resource List

- MBAR 2000

Tabulka 20-1. Přehled zdrojů souboru `ppforthrsc.prc` [1:1:4]

MBAR	2000	hlavní menu
Talt	3000	
tAIB	1000	
tAIB	1001	
tFRM	1000	
tSTR	3001	

20.3. Struktura slovníku a slov

Příklad 20-1. Záznam slova

```
2B  view
2B  link
    counted string = 1B counter, "string"
    code
```

Kapitola 21. PPForth zevnitř

* *rcsinfo="$Header: /home/radek/cvs/forth-book/ch-ppf-internals.xml,v 1.4 2005/10/20 05:33:42 radek Exp $"*

epigram

Text kapitoly

21.1. Analýza binárního kódu

.

21.1.1. Začátek zdroje code 1

Příklad 21-1. První část code 1

```
; PPForth code 1
;        1111111112222222222333333333344444444445555555555666666666677777777778
;2345678901234567890123456789012345678901234567890123456789012345678901234567890
0000= ....: 4E56 FFF4 .... ....  START:     LINK    A6, -12
0004=       486E FFF4                       PEA     -12(A6)
0008=       486E FFF8                       PEA     -8(A6)
0004=       486E FFFC                       PEA     -4(A6)
0010=       4E4F A08F                       SYSTRAP SysAppStartup
0014=       4FEF 000C                       LEA     12(A7), A7
0018=       4A40                            TST.W   D0
001A=       670E                          .-BEQ.S   OK
                                          |
                                          | ; V případě neúspěchu pípneme a ukončíme program
001C=       1F3C 0003                     | MOVE.B  #3, -(A7)
0020=       4E4F A234                     | SYSTRAP SndPlaySystemSound
0024=       548F                          | ADDQ.L  #2, A7
0026=       70FF                          | MOVEQ   #-1, D0
0028=       602A                        .-|-BRA.S   END  ;=$8056    ; Konec
                                        | |
                                        | | ; Příprava argumentů a volání procedury SBR_0058
002A=       206E FFFC             OK:   | '>MOVE.L  -4(A6), A0
002E=       3F28 0006                   |   MOVE    6(A0), -(A7)
0032=       2F28 0002                   |   MOVE.L  2(A0), -(A7)
0036=       3F10                        |   MOVE    (A0), -(A7)
0038=       4EBA 001E                   |  =JSR     SBR_0058        ; CALL ()
                                        |
003C= 803E: 508F                        |   ADDQ.L  #8, RP
003E= 8040: 2F2E FFF4                   |   MOVE.L  -12(FP), -(RP)
0042= 8044: 2F2E FFF8                   |   MOVE.L  -8(FP), -(RP)
0046= 8048: 2F2E FFFC                   |   MOVE.L  -4(FP), -(RP)
004A= 804C: 4E4F A090                   |   SYSTRAP SysAppExit
004E= 8050: 4FEF 000C                   |   LEA     12(RP), RP
0052= 8054: 7000                        |   MOVEQ   #0, D0
                                        |
0054= 8056: 4E5E                  END:  '-->UNLK    A6              ; Return to PalmOS
0056= 8058: 4E75                            RTS                     ; %
```

; Subroutine $0058

```
0058=       4E56 0000            SBR_0058:  LINK    A6, #0          ;Rámec bez lokálních promě
005C=       4A6E 0008                       TST.W   8(A6)           ;ARG.
0060=       6704                          .-BEQ     0066            ; $+4
0062=       4E5E                          | UNLK    A6
0064=       4E75                          | RTS
0066=       6100 001E                   .='>BSR     0086
                                        |
006A=       0004 0001 0028 00D1         |   DW      4, 1, $28, $D1
0072=       000A                        |   DW      $0A
0074=       0000 0000                   |   DW      0, 0
0078=       00D1 7374 6172 7474         |   DB      00, $D1, "startt"
0080=       75B5                        |
0082=       F66C                        |
0084=       A14E                        |
0086=       588F                        '-->ADDQ.L  #0, A7
0088=       47FA FF76                       LEA     $FF76(PC), A3
008C=       2F3C 0000                       MOVE.L  #0, -(A7)
0090=       FFE0
0092=       4E4F A013                       SYSTRAP $A013
0096=       588F                            ADDQ.L  #0, A7
0098=       203C 0000                       MOVE.L  #0, D0
009C=       8000
009E=       5980
00A0=       21B3
00A2=       0800
00A4=       0080

011C=       01B4                            DW      $01B4       ; view
011E=       0000                            DW      0           ; link
0120=       046E 6F6F 7004                  DB      4, "noop", 4
0126=       3A1A                            MOVE.W  (A2)+, D5
0128=       4EF3 5800                       JMP     ???

012C=       01B9
012E=       0000
0130=       4564 6F6C 6974 0500             DB      $40+5, "dolit", 5, 0
0138=       3F1A                            MOVE.W  (A2)+, -(A7)
013A=       3A1A                            MOVE.W  (A2)+, D5
013C=       4EF3                            JMP     ???
013E=       5800

0140=       01C0 0000
0144=       4764 6F76 616C 7565 0700        DB      $40+7, "dovalue", 7, 0
014F=       3F1A                            MOVE.W  (A2)+, -(A7)
0150=       3A1A                            MOVE.W  (A2)+, D5
0152=       4EF3 5800                       JMP     ???


0156=       01C7 0000
015A=       4664 6F6C 6973 7406             DB      $40+6, "dolist", 6
0162=       220A                            MOVE.L  A2, D1
0164=       928B

0170=       01D5 0000
0174=       4664 6F64 6F65 7306             DB      $40+6, "dodoes", 6
017C=       220A 928B
```

```
0190=       01E5 012E
0194=       4662 726E 6578 7406             DB      $40+6, "brnext", 6
019C=       5354 4A54

01BF=       01F6
01C0=       0000
01C2=       473F 6272 616E 6368 0700        DB      $40+7, "?branch", 7



C5C8=       C02C C0A2
C5CC=       0666 6F72 6D68 6906             DB      6, "formhi", 5
C5D4=       3A3C 0162
C5D8=       4EB3
C5DA=       5800
```

VI. Jiné jazyky inspirované Forthem

Kusé informace i jazycích podobných či inspirovaných forthem.

Zkrácený přehled:

- Joy
- Factor (http://factorcode.org/)

Kapitola 22. Factor

Factor

VII. Mikroprocesory

Kapitola 23. Motorola MC68000 CPU

* *rcsinfo="$Header: /home/radek/cvs/forth-book/ch-mc68k.xml,v 1.7 2005/10/20 05:33:42 radek Exp $"*

epigram

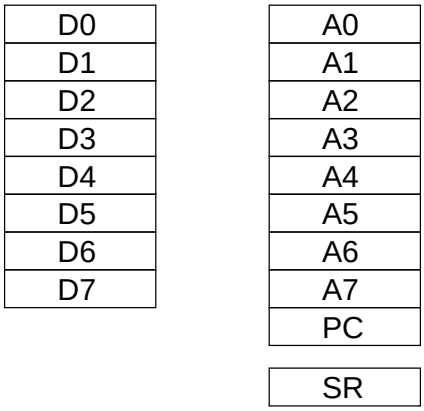
* *Umístněním této kapitoly si nejsem jist. Je možno z ní udělat přílohu (appendix).*

Text kapitoly

23.1. Registry

odstavec

Obrázek 23-1. Registry mikroprocessoru MC68000



23.2. Způsoby adresování

Popis jednotlivých způsobů adresování.

Tabulka 23-1. Adresovací módy MC68k [2:1:1]

mód	**popis**	**zápis**
Data Register Direct	EA = Dn	Dn
Address Register Direct	EA = An	An
Adress Register Indirect	EA = (An)	(An)
Address Register Indirect with Postincrement	EA = (An) + SIZE	(An)+
Address Register Indirect with Predecrement	EA = (An) - SIZE	-(An)
Address Register Indirect with Displacement	EA = (An) + d_{16}	(d_{16}, An)
Address Register Indirect with Index (8-bit Displacement)	EA = (An) + (Xn) + d_8	(d_8, An, Xn.SIZE*SCALE)
Address Register Indirect with Index (Base Displacement)	EA = (An) + (Xn) + bd	(bd, An, Xn.SIZE*SCALE
Memory Indirect Postindexed	EA = (An + bd) + Xn.SIZECALE + od	([bd,An], Xn.SIZE*SCALE, od)

mód	**popis**	**zápis**
Memory Indirect Preindexed	EA = (bd + An) + Xn-SIZE*SCALE + od	([bd, An, Xn.SIZE*SCALE], od)
Program Counter Indirect with Displacement	EA = (PC) + d_{16}	(d_{16},PC)
Program Counter Indirect with Index (8-bit Displacement)	EA = (PC) + (Xn) + d_8	(d_8, PC, Xn.SIZE*SCALE)
Program Counter Indirect with Index (Base Displacement)	EA = (PC) + (Xn) + bd	(bd,PC,Xn.SIZE*SCALE)
Program Counter Memory Indirect Postindexed	EA = (bd + PC) + X.SIZE*SCALE + od	([bd, PC], Xn.SIZE*SCALE,od)
Program Counter Memory Indirect Preindexed	EA = (bd + PC) + Xn.SIZE*SCALE + od	([bd, PC, Xn.SIZE*SCALE], od)
Absolute Short Addressing	EA GIVEN	(xxx).W
Absolute Long Addressing	EA GIVEN	(xxx).L
Immediate Data	OPERAND GIVEN	#<xxx>

Tabulka 23-2. Effective Addressing Modes and Categories [3:2:1:1:1:1:1:1]

Addressing Modes	**Syntax**	**Mode Field**	**Reg. Field**	**Data**	**Memory**	**Control**	**Alterable**
Register Direct							
Data	Dn	000	reg. no.	X	-	-	X
Address	An	001	reg. no.	-	-	-	X
Register Indirect							
Address	(An)	010	reg. no.	X	X	X	X
Address with Postincrement	(An)+	011	reg. no.	X	X	-	X
Address with Predecrement	-(An)	100	reg. no.	X	X	-	X
Address with Displacement	(d_{16},An)	101	reg. no.	X	X	X	X
Address Register Indirect with Index							
8-bit Dispalcement	(d_8,An,Xn)	110	reg. no.	X	X	X	X
Base Displacement	(bd,An,Xn)	110	reg. no.	X	X	X	X
Memory Indirect							
Postindexed	([bd,An],Xn,od)	110	reg. no.	X	X	X	X
Preindexed	([bd,An,Xn],od)	110	reg. no.	X	X	X	X
Program Counter Indirect							

Addressing Modes	Syntax	Mode Field	Reg. Field	Data	Memory	Control	Alterable
with Displacement	(d_{16},PC)	111	010	X	X	X	-
Program Counter Indirect with Index							
8-Bit Displacement	(d_8,PC,Xn)	111	011	X	X	X	-
Base Dispalcement	(bd,PC,Xn)	111	011	X	X	X	-
Program Counter Memory Indirect							
Postindexed	([bd,PC],Xn,od)	111	011	X	X	X	X
Preindexed	([bd,PC,Xn],od)	111	011	X	X	X	X
Absolute data Addressing							
Short	(xxx).W	111	000	X	X	X	-
Long	(xxx).L	111	000	X	X	X	-
Immediate	#<xxx>	111	100	X	X	-	-

Tabulka 23-3. Effective Addressing Modes and Categories

Addressing Modes	Syntax	Mode Field	Reg. Field	Data	Memory	Control	Alterable
Register Direct							
Data	Dn	000	reg. no.	X	-	-	X
Address	An	001	reg. no.	-	-	-	X
Register Indirect							
Address	(An)	010	reg. no.	X	X	X	X
Address with Postincre-ment	(An)+	011	reg. no.	X	X	-	X
Address with Pre-decrement	-(An)	100	reg. no.	X	X	-	X
Address with Dis-placement	(d_{16},An)	101	reg. no.	X	X	X	X
Address Register Indirect with Index							
8-bit Dis-palcement	(d_8,An,Xn)	110	reg. no.	X	X	X	X

Addressing Modes	Syntax	Mode Field	Reg. Field	Data	Memory	Control	Alterable
Base Displacement	(bd,An,Xn)	110	reg. no.	X	X	X	X
Memory Indirect							
Postindexed	([bd,An],Xn,od)	110	reg. no.	X	X	X	X
Preindexed	([bd,An,Xn],od)	110	reg. no.	X	X	X	X
Program Counter Indirect							
with Displacement	(d_{16},PC)	111	010	X	X	X	-
Program Counter Indirect with Index							
8-Bit Displacement	(d_8,PC,Xn)	111	011	X	X	X	-
Base Dispalcement	(bd,PC,Xn)	111	011	X	X	X	-
Program Counter Memory Indirect							
Postindexed	([bd,PC],Xn,od)	111	011	X	X	X	X
Preindexed	([bd,PC,Xn],od)	111	011	X	X	X	X
Absolute data Addressing							
Short	(xxx).W	111	000	X	X	X	-
Long	(xxx).L	111	000	X	X	X	-
Immediate	#<xxx>	111	100	X	X	-	-

VIII. Přílohy

Příloha A. Různé zatím nezařazené sekce

A.1. Assembler procesoru 6502

* *$Header: /home/radek/cvs/forth-book/sec-6502asm.xml,v 1.1 2003/12/28 18:21:56 radek Exp $*

FIXME: obsah

```
( FORTH-65 ASSEMBLER )
HEX
VOCABULARY ASSEMBLER IMMEDIATE  ASSEMBLER DEFINITIONS
( REGISTER ASSIGNMENT SPECIFIC TO IMPLEMENTATION )
E0  CONSTANT  XSAVE
DC  CONSTANT  W
DE  CONSTANT  UP
D9  CONSTANT  IP
D1  CONSTANT  N

( NUCLEUS LOCATIONS ARE IMPLEMENTATION SPECIFIC )
'  (DO)  0E  +  CONSTANT  POP
'  (DO)  OC  +  CONSTANT  POPTWO
'  LIT 13 + CONSTANT PUT
'  LIT 11 + CONSTANT PUSH
'  LIT 18 + CONSTANT NEXT
'  EXECUTE NFA 11 - CONSTANT SETUP

( FORTH-65 ASSEMBLER )
0 VARIABLE INDEX -2 ALLOT
0900 , 1505 , 0115 , 8011 , 8009 , 1D0D , 8019 , 8080 ,
0080 , 1404 , 8014 , 8080 , 8080 , 1C0C , 801C , 2C08 ,

2 VARIABLE MODE
: .A  0 MODE ! ;   : #  1 MODE ! ;   : MEM 2 MODE ! ;
: ,X  3 MODE ! ;   : ,Y 4 MODE ! ;   : X)  5 MODE ! ;
: )Y  6 MODE ! ;   : )  F MODE ! ;

: BOT  ,X   0 ;  ( ADDRESS THE BOTTOM OF THE STACK *)
: SEC  ,X   2 ;  ( ADDRESS SECOND ITEM ON STACK    *)
: RP)  ,X 101 ;  ( ADDRESS BOTTOM OF RETURN STACK  *)
```

A.2. Forth Objects

* *section id="forth-objects"*
* *rcsinfo="$Header: /home/radek/cvs/forth-book/sec-oop_ve_forthu.xml,v 1.3 2005/10/20 05:33:42 radek Exp $"*
* *Podle http://c2.com/cgi/wiki?ForthObjects.*

Kompletní implemetace HYPE, jednoduchého OOF s použitím ANSI standard Forth

```
: LIT,  ( x )  POSTPONE LITERAL ;
: >SIZE  ( ta - n )  CELL+ @ ;
0 VALUE SELF
: SELF+  ( n - a )  SELF + ;
```

```
: SEND  ( a xt )  SELF >R   SWAP TO SELF EXECUTE  R> TO SELF ;
VARIABLE CLS  ( contains ta )
: SIZE^  ( - aa )  CLS @ ?DUP 0= ABORT" scope?" CELL+ ;
: MFIND  ( ta ca u - xt n )  2>R BEGIN DUP WHILE DUP @ 2R@ ROT
    SEARCH-WORDLIST ?DUP IF ROT DROP 2R> 2DROP EXIT THEN
    CELL+ CELL+ @ REPEAT -1 ABORT" can't?" ;
: SEND'  ( a ta "m " )  BL WORD COUNT MFIND 0< STATE @ AND
    IF SWAP LIT, LIT, POSTPONE SEND ELSE SEND THEN ;
: SUPER  ( "m " )  SIZE^ CELL+ @ BL WORD COUNT MFIND 0>
    IF EXECUTE ELSE COMPILE, THEN ; IMMEDIATE
: DEFS  ( n "f " )  CREATE SIZE^ @ , SIZE^ +! IMMEDIATE
    DOES> @ STATE @ IF LIT, POSTPONE SELF+ ELSE SELF+ THEN ;
: METHODS  ( ta )  DUP CLS ! @ DUP SET-CURRENT
    >R GET-ORDER R> SWAP 1+ SET-ORDER ; ( ALSO CONTEXT ! )
: CLASS  ( "c " )  CREATE HERE 0 , 0 , 0 ,
    WORDLIST OVER ! METHODS ;
: SUBCLASS  ( ta "c " )  CLASS SIZE^ OVER >SIZE OVER ! CELL+ ! ;
: END  ( )  SIZE^ DROP PREVIOUS DEFINITIONS 0 CLS ! ;
: NEW  ( ta "name " )  CREATE DUP , >SIZE ALLOT IMMEDIATE
    DOES> DUP CELL+ SWAP @ SEND' ;
```

Poznámky

- Tento kód implementuje třídy a podtřídy s veřejnými metodami a proměnnými instancí.
- Kód je jedním z nejjednoduších objektově orientovaných rozšíření Forthu který je přiměřeně použitelný. Kód není nejpěknější, ale je standardní, kompaktní a přiměřeně dobře faktorizovaný.
- HYPE je celé napsáno jen velikými písmeny. Toto je vyžadováno přísnou kompatibilitou z ANSI Forth.
- HYPE se přeloží do méně než 2K kódu na většině Forthů. Na některých platformách to bude méně než 500 bajtů.

S použitím uvedeného kódu můžeme psát:

```
: VAR 1 CELLS DEFS ;            \ Helper for creating instance vars

CLASS BUTTON
    VAR TEXT
    VAR LEN
    VAR X
    VAR Y
: DRAW  ( )
    X @ Y @ AT-XY               \ Get X and Y, and position cursor on screen
    TEXT @ LEN @ TYPE ;         \ Get TEXT and LENgth, and type it
: INIT  ( ca u )  0 X ! 0 Y ! LEN ! TEXT ! ;
END

: BOLD   27 EMIT ." [1m" ;      \ Emit code to turn on BOLD TEXT
: NORMAL 27 EMIT ." [0m" ;      \ Emit code to return to normal text

BUTTON SUBCLASS BOLD-BUTTON
: DRAW  ( )  BOLD SUPER DRAW NORMAL ;
END
```

Definované třídy můžeme použít

```
BUTTON NEW FOO                  \ creates new button "FOO"
S" thin foo" FOO INIT           \ calls init method on FOO with string arg.
```

```
PAGE                            \ clear the screen
FOO DRAW                        \ draw FOO
BOLD-BUTTON NEW BAR             \ create new bold-button
S" fat bar" BAR INIT            \ initialize
1 BAR Y !                       \ change the Y instance variable
BAR DRAW                        \ draw the BAR button
```

A.3. Editor Forth INC.

* *$Header: /home/radek/cvs/forth-book/sec-forth_inc_editor.xml,v 1.1 2003/12/28 18:21:56 radek Exp $*

FIXME: obsah

SCR # 69

```
      ( FORTH INC.'S EDITOR )

      ( This editor was written by S.H. Daniel, in FORTH DIMENSIONS,
      ( Volume III, number 3.
    5
      ( The only changes was to make the cursor a "block" for higher
      ( visibility.  P. Mullarky 9/29/81

      -->
```

SCR # 70

```
      ( FORTH INC.'S EDITOR )

      BASE @ FORTH DEFINITIONS HEX

    5 : TEXT HERE C/L 1+ BLANKS WORD HERE PAD C/L 1+ CMOVE ;
      : LINE DUP FFF0 AND 17 ?ERROR SCR @ (LINE) DROP ;
      VOCABULARY EDITOR IMMEDIATE
      : WHERE DUP B/SCR / DUP SCR ! ." SCR # " DECIMAL . SWAP
      C/L /MOD C/L * ROT BLOCK + CR C/L TYPE [COMPILE] EDITOR QUIT ;
   10 EDITOR DEFINITIONS
      : #LOCATE R# @ C/L /MOD ;
      : #LEAD #LOCATE LINE SWAP ;
      : #LAG #LEAD DUP >R + C/L R> - ;
      : -MOVE LINE C/L CMOVE UPDATE ;
   15 : BUF-MOVE PAD 1+ C@ IF PAD SWAP C/L 1+ CMOVE ELSE DROP THEN ;
      : >LINE# #LOCATE SWAP DROP ;  -->
```

SCR # 71

```
      ( FORTH INC.'S EDITOR )

      : FIND-BUF PAD 50 + ;
      : INSERT-BUF FIND-BUF 50 + ;
```

```
    5 : (HOLD) LINE INSERT-BUF 1+ C/L DUP INSERT-BUF C! CMOVE ;
      : (KILL) LINE C/L BLANKS UPDATE ;
      : (SPREAD) >LINE# DUP 1 - E DO I LINE I 1+ - MOVE -1
      +LOOP (KILL) ;
      : X >LINE# DUP (HOLD) F DUP ROT DO I 1+ LINE I -MOVE
   10 LOOP (KILL) ;
      : DISPLAY-CURSOR CR SPACE #LEAD TYPE A0 EMIT #LAG TYPE
      #LOCATE . DROP ;
      : T C/L * R# ! 0 DISPLAY-CURSOR ;
      : L SCR @ LIST ;
   15 : N 1 SCR +! ;
      : B -1 SCR +! ;      -->
```

SCR # 72

```
      ( FORTH INC.'S EDITOR )

      : (TOP) 0 R# ! ;
      : SEEK-ERROR (TOP) FIND-BUF HERE C/L 1+ CMOVE HERE COUNT TYPE
    5 ." None" QUIT ;
      : (R) >LINE# INSERT-BUF 1+ SWAP -MOVE ;
      : P 5E TEXT INSERT-BUF BUF-MOVE (R) ;
      : WIPE 10 0 DO I (KILL) LOOP ;
      : COPY B/SCR * OFFSET @ + SWAP B/SCR * B/SCR OVER + SWAP DO DUP
   10 FORTH I BLOCK 2 - ! 1+ UPDATE LOOP DROP FLUSH ;
      : 1LINE #LAG FIND-BUF COUNT MATCH R# +! ;
      : (SEEK) BEGIN 3FF R# @ < IF SEEK-ERROR THEN 1LINE UNTIL ;
      : (DELETE) >R #LAG + R - #LAG R MINUS R# +! #LEAD + SWAP
      CMOVE R> BLANKS UPDATE ;
   15 : (F) 5E TEXT FIND-BUF BUF-MOVE (SEEK) ;
      : F (F) DISPLAY-CURSOR ;   -->
```

SCR # 73

```
      ( FORTH INC.'S EDITOR )

      : E (E) DISPLAY-CURSOR ;
      : D (F) E ;
    5 : TILL #LEAD + 5E TEXT FIND-BUF BUF-MOVE 1LINE 0= IF
      SEEK-ERROR THEN #LEAD + SWAP - (DELETE) DISPLAY-CURSOR ;
      0 VARIABLE COUNTER
      : BUMP 1 COUNTER 1+ COUNTER @ 38 > IF 0 COUNTER ! CR CR
      F MESSAGE C EMIT THEN ;
   10 : S C EMIT 5E TEXT 0 COUNTER ! FIND-BUF BUF-MOVE SCR @ DUP
      >R DO I SCR ! (TOP) BEGIN 1LINE IF DISPLAY-CURSOR SCR ? BUMP
      THEN 3FF R# @ < UNTIL LOOP R> SCR ! ;
      : I 5E TEXT INSERT-BUF BUF-MOVE INSERT-BUF COUNT #LAG ROT
      OVER MIN >R R R# +! R - >R DUP HERE R CMOVE HERE #LEAD + R>
   15 CMOVE R> CMOVE UPDATE
      DISPLAY-CURSOR ;     -->
```

SCR # 74

```
   ( FORTH INC.'S EDITOR )

   : U C/L R# +! (SPREAD) P ;
   : R (E) I ;
 5 : M SCR @ >R R# @ >R >LINE# (HOLD) SWAP SCR ! 1+ C/L * R#
   (SPREAD) (R) R> C/L + R# R> SCR ! ;


   DECIMAL
10 LATEST 12 +ORIGIN !
   HERE 28 +ORIGIN !
   HERE 30 +ORIGIN !
   ' EDITOR 6 + 32 +ORIGIN !
   HERE FENCE !
15 FORTH DEFINITIONS  BASE !  FORTH ;S
```

IX. Slovníky

abstract

Zde uvádím někdy ne zcela úplné slovníky různých implementací forthu.

Předem se omlouvám za organizaci této části. Je ve stavu do jakého se pomalu vyvýjela. Nikdy jsem s ní nebyl spokojen a nejsem ani teď. Ovšem nenapadá mne lepší způsob organizace. Vždy se vyskytnou nějaká ale a zcela zásední problémy.

Abych uvedl čtenáře do obrazu, každé slovo má svůj identifikátor, který mi dovoluje se na něj odkazovat. Potom ukázky programů kdekoliv v této knize jsou doslova „prošpikovány“ odkazy na použitá slova. Čtenář tak může ihned přeskočit k definici slova kterému nerozumí, nebo by je potřeboval osvětlit, jak je vidět na ukázce.

```
&session.dict-plus-store;
```

Zásadním problémem je, že v knize zmiňuji nekolik implementací forthu které se liší přidanými slovy. Vyvztává otázka jak slova v této části organizovat, neb některá slova se zcela přirozeně vyskytují ve více nebo téměř všech implementacích. Rozdělení na samostatné slovníky přinese jednak navýšení objmu knihy z důvodu duplicitních informací. Druhým závažnějším problémem je, která varianta slova, v kterém slovníku je ta správná na kterou se odkazovat.

Nejpřirozenější mi přijde vytvořit jeden veliký slovník pojímající slova všech implementací. Tím se vyřeší problém s duplicitami i odkazy. Ovšem tento jeden veliký slovník se pak z určitého pohledu stává méně přehledným, neb obsahuje velmi rozsáhlou koleci slov. Připustil jsem proto existenic samostatných slovníků tam, kde jsou rozsáhlé specifické množiny slov dané implementace, jako je například slovník [xref linkend="PalmOS-API"/].

Ovšem zůstává problém historický, kdy mnoho slov je zapsáno některým z dříve použitých způsobů a tak potrvá delší čas než překlopím celou slovníkovou přílohu do nové organizace.

I. (Veliký) Slovník Forthu

* *reference id="forth-dictionary"*
* *rcsinfo=$Id: dbheader.xml,v 1.1 2005/01/19 21:52:04 radek Exp $*

Tento slovník je hlavním slovníkem knihy. Sem umístním všechna slova, které chci popsat. Ostatní slovníky jsou jen pro implementačně velmi specifická slova které jsem z důvodů obdáhlosti nechtěl umístňovat do tohoto, hlavního slovníku.

Odkazy na

- **FIXME:** ()

!CSP

* *$Header: /home/radek/cvs/forth-book/dictionary/ store-csp,v new 2003/12/31 00:08:29 radek Exp $*

Jméno

!CSP — „sound“ popis

CORE

Přehled

(⟶)

Přeložit: Save the stack position in CSP. Used as part of the compiler security.

Popis

FIXME:

Definice

FIXME:

```
FIXME: definice
```

II. Slovník ANSI forthu

Slova definovaná v ANSI

Některá slova, jenž jsem považoval za vhodná zde uvést.

Odkazy na

- dpANS Forth (http://forth.sourceforge.net/standard/dpans/) na comp.lang.forth.repository (http://forth.sourceforge.net/)
- dpANS Forth (http://pfe.sourceforge.net/dpans/dpans6.htm) jako součást PFE na SourceForge
- dpANS Froth 94 (http://www.taygeta.com/forth/dpans.htm) na http://www.taygeta.com
- dpANS Forth (http://kristopherjohnson.net/dpans/) na Kristopher Johnson (http://kristopherjohnson.net/)

!

* *$Header: /home/radek/cvs/forth-book/dictionary/ansi/0010.store,v 1.1 2003/12/28 18:21:57 radek Exp $*

Jméno

! — „store“, uložení hodnoty na adresu

Přehled

```
: !   ( x a-addr ⟶ ) ;
```

Popis

Definováno v: dpANS Forth 6.1.0010 CORE (http://forth.sourceforge.net/standard/dpans/dpans6.htm#6.1.0010)

Uloží hodnotu *x* do buňky na adrese *a-addr*

Příklad použití

```
# $Id: dict-store.ses,v 1.1 2003/02/02 12:39:15 radek Exp $
Gforth 0.6.2, Copyright (C) 1995-2003 Free Software Foundation, Inc.
Gforth comes with ABSOLUTELY NO WARRANTY; for details type ‘license’
VARIABLE v  ok
7 v !  ok
v @ . 7  ok
BYE
```

Kód slova v Qurtus Forthu

Příklad 1. *:[90]

number

* *$Header: /home/radek/cvs/forth-book/dictionary/ansi/0030.number-sign,v 1.1 2003/12/28 18:21:57 radek Exp $*

Jméno

`number` — „number-sign“, dělení čísle v proměnné base

Přehled

```
: #   ( ud1 ⟶ ud2 ) ;
```

Popis

Definováno v: dpANS Forth 6.1.0030 CORE (http://forth.sourceforge.net/standard/dpans/dpans6.htm#6.1.0030)

Dělí *ud1* hodnotou proměnné base. v *ud2* je výsledek dělení zbytek po dělení, tedy nejméně významná číslice čísla v soustave určené base se uloží jako znak do výstupního řetězce. Používá se mezi {xref linkend="less-number-sign"/> a {xref linkend="number-sign-greater"/>.

Příklad použití

```
*FIXME: dplnit příklad
```

Kód slova v Qurtus Forthu

Příklad 1. #> ***[80]***

number >

* *$Header: /home/radek/cvs/forth-book/dictionary/ansi/0040.number-sign-greater,v 1.1 2003/12/28 18:21:57 radek Exp $*

Jméno

`number >` — „number-sign-greater“, ukončení formátování

Přehled

```
: #>   ( xd ⟶ c-addr u ) ;
```

Popis

Definováno v: dpANS Forth 6.1.0040 CORE (http://forth.sourceforge.net/standard/dpans/dpans6.htm#6.1.0040)

Příbuzná slova

#>, #S, <#

Příklad použití

```
*FIXME:
```

Kód slova v Qurtus Forthu

Příklad 1. *:[90]

#S

* *$Header: /home/radek/cvs/forth-book/dictionary/ansi/0050.number-sign-s,v 1.1 2003/12/28 18:21:57 radek Exp $*

Jméno

#S — FIXME: jednořádkový popis

Přehled

```
: #S   ( ud1 ⟶ ud2 ) ;
```

Popis

Definováno v: dpANS Forth 6.1.0050 CORE (http://forth.sourceforge.net/standard/dpans/dpans6.htm#6.1.0050)

Převede jednu číslici z *ud1* podle previdel pr #. Převod pokračuje dokud podíl není nula. *ud2* není nula.

Příbuzná slova

#, #>, <#

Příklad použití

```
*FIXME:
```

Kód slova v Qurtus Forthu

Příklad 1. *:[90]

number TIB

* *$Header: /home/radek/cvs/forth-book/dictionary/ansi/0060.number-t-i-b,v 1.1 2003/12/28 18:21:57 radek Exp $*

Jméno

`number TIB` — „t-i-b“ adresa buňky obsahující velikost tib

Přehled

(⟶ a-addr)

Adresa *a-addr* uložená na zásobník je adresa buňky jenž obsahuje počet znaků ve vstupním terminálovém bufferu (TIB).

Popis

*FIXME:

Description

a-addr is the address of a cell containing the number of characters in the terminal input buffer.

Poznámka: This word is obsolescent and is included as a concession to existing implementations.

The function of #TIB has been superseded by {xref linkend="source"/>.

Příklad použití

```
# $Id: dict-plus-store.ses,v 1.2 2003/12/28 18:21:58 radek Exp $
Gforth 0.6.2, Copyright (C) 1995-2003 Free Software Foundation, Inc.
Gforth comes with ABSOLUTELY NO WARRANTY; for details type ‘license’
VARIABLE v  ok
8 v !  ok
4 v +!  ok
v @ . 12  ok
BYE
```

Kód slova v Qurtus Forthu

Příklad 1. *:[90]

```
,
```

* *$Header: /home/radek/cvs/forth-book/dictionary/ansi/0070.tick,v 1.1 2003/12/28 18:21:57 radek Exp $*

Jméno

' — „tick“

Přehled

```
: '   ( "<spaces>name" ⟶ xt) ;
```

Popis

Definováno v: dpANS Forth 6.1.0070 CORE (http://forth.sourceforge.net/standard/dpans/dpans6.htm#6.1.0070)

Description

Skip leading space delimiters. Parse name delimited by a space. Find name and return xt, the execution token for name. An ambiguous condition exists if name is not found.

When interpreting, ' xyz EXECUTE is equivalent to xyz.

Return the execution token of the following name. This word is _not_ immediate and may not do what you expect in compile-mode. See ['] and '> - note that in FIG-forth the word of the same name had returned the PFA (not the CFA) and was immediate/smart, so beware when porting forth-code from FIG-forth to ANSI-forth.

Příklad použití

```
*FIXME:
```

Kód slova v Qurtus Forthu

Příklad 1. *:[90]

(

* *$Header: /home/radek/cvs/forth-book/dictionary/ansi/0080.paren,v 1.1 2003/12/28 18:21:57 radek Exp $*

Jméno

(— „paren“, zahájení kometáře

Přehled

("ccc<paren>" ⟶) ; IMMEDIATE

Definováno v: dpANS Forth 6.1.0080 CORE (http://forth.sourceforge.net/standard/dpans/dpans6.htm#6.1.0080)

Definováno v: dpANS Forth 11.6.1.0080 FILE (http://forth.sourceforge.net/standard/dpans/dpans11.htm#11.6.1.0080)

Popis

Ignoruje vše až do „)“

Description

Parse ccc delimited by) (right parenthesis). (is an immediate word. The number of characters in ccc may be zero to the number of characters in the parse area.

Extend the semantics of 6.1.0080 (to include: When parsing from a text file, if the end of the parse area is reached before a right parenthesis is found, refill the input buffer from the next line of the file, set >IN to zero, and resume parsing, repeating this process until either a right parenthesis is found or the end of the file is reached.

Příklad použití

```
*FIXME:
```

Kód slova v Qurtus Forthu

Příklad 1. *:[90]

(LOCAL)

* *$Header: /home/radek/cvs/forth-book/dictionary/ansi/0086.paren-local-paren,v 1.1 2003/12/28 18:21:57 radek Exp $*

Jméno

`(LOCAL)` — popis

Přehled

`Interpretace:` *`význam není definován`*

`Běh:` **`( c-addr u ⟶ )`**

`Běh:` **`( strptr strcnt ⟶ )`**

Definováno v: dpANS Forth 13.6.1.0086 LOCAL (http://forth.sourceforge.net/standard/dpans/dpans13.htm#13.6.1.0086)

FIXME:

Popis

Ukončí práci v prostředí Forthu, a provede návrat do systému odkud jsme Forth spustili.

Description

When executed during compilation, (LOCAL) passes a message to the system that has one of two meanings. If u is non-zero, the message identifies a new local whose definition name is given by the string of characters identified by c-addr u. If u is zero, the message is last local and c-addr has no significance.

The result of executing (LOCAL) during compilation of a definition is to create a set of named local identifiers, each of which is a definition name, that only have execution semantics within the scope of that definition's source.

```
local Execution: ( -- x )
```

Push the local's value, x, onto the stack. The local's value is initialized as described in 13.3.3 Processing locals and may be changed by preceding the local's name with TO. An ambiguous condition exists when local is executed while in interpretation state.

Poznámka: This word does not have special compilation semantics in the usual sense because it provides access to a system capability for use by other user-defined words that do have them. However, the locals facil-

ity as a whole and the sequence of messages passed defines specific usage rules with semantic implications that are described in detail in section 13.3.3 Processing locals.

Poznámka: This word is not intended for direct use in a definition to declare that definition's locals. It is instead used by system or user compiling words. These compiling words in turn define their own syntax, and may be used directly in definitions to declare locals. In this context, the syntax for (LOCAL) is defined in terms of a sequence of compile-time messages and is described in detail in section 13.3.3 Processing locals.

Poznámka: The Locals word set modifies the syntax and semantics of 6.2.2295 TO as defined in the Core Extensions word set.

this word is used to create compiling words that can declare LOCALS| - it shall not be used directly to declare a local, the pfe provides LVALUE for that a purpose beyond LOCALS|

*****

* *$Header: /home/radek/cvs/forth-book/dictionary/ansi/0090.star,v 1.1 2003/12/28 18:21:57 radek Exp $*

Jméno

`*` — „star“, násobení

Přehled

`( n1|u1 n2|u2 ⟶ n3|u3)`

Definováno v: dpANS Forth 6.1.0090 CORE (http://forth.sourceforge.net/standard/dpans/dpans6.htm#6.1.0090)

Popis

FIXME:

Kód slova v Qurtus Forthu

Příklad 1. *:[90]

*/

* *$Header: /home/radek/cvs/forth-book/dictionary/ansi/0100.star-slash,v 1.1 2003/12/28 18:21:57 radek Exp $*

Jméno

`*/` — FIXME: jednořádkový popis

CORE

Přehled

`( n1 n2 n3 ⟶ n4 )`

Definováno v: dpANS Forth 6.1.0100 CORE (http://forth.sourceforge.net/standard/dpans/dpans6.htm#6.1.0100)

Popis

vynásobí n1 n2 a výsledek dělí n3. Mezivýsledky jsou v dvojnásobné velikosti.

Výsledek by měl být stejný jako u posloupností příkazů

```
>R M* R> FM/MOD SWAP DROP
```

nebo

```
>R M* R> SM/REM SWAP DROP
```

Description

Multiply n1 by n2 producing the intermediate double-cell result d. Divide d by n3 giving the single-cell quotient n4. An ambiguous condition exists if n3 is zero or if the quotient n4 lies outside the range of a signed number. If d and n3 differ in sign, the implementation-defined result returned will be the same as that returned by either the phrase:

```
>R M* R> FM/MOD SWAP DROP
```

or the phrase

```
>R M* R> SM/REM SWAP DROP
```

Příklad použití

```
FIXME:
```

Kód slova v Qurtus Forthu

Příklad 1. *:[90]

*/MOD

* *$Header: /home/radek/cvs/forth-book/dictionary/ansi/0110.star-slash-mod,v 1.1 2003/12/28 18:21:57 radek Exp $*

Jméno

`*/MOD` — FIXME: jednořádkový popis

CORE

Přehled

`( n1 n2 n3 ⟶ n4 n5 )`

Definováno v: dpANS Forth 6.1.0110 CORE (http://forth.sourceforge.net/standard/dpans/dpans6.htm#6.1.0110)

Popis

FIXME:vynásobí n1 n2 a výsledek dělí n3. Mezivýsledky jsou v dvojnásobné velikosti.

Výsledek by měl být stejný jako u posloupností příkazů

```
>R M* R> FM/MOD SWAP DROP
```

nebo

```
>R M* R> SM/REM SWAP DROP
```

Description

Multiply n1 by n2 producing the intermediate double-cell result d. Divide d by n3 producing the single-cell remainder n4 and the single-cell quotient n5. An ambiguous condition exists if n3 is zero, or if the quotient n5 lies outside the range of a single-cell signed integer. If d and n3 differ in sign, the implementation-defined result returned will be the same as that returned by either the phrase >R M* R> FM/MOD or the phrase >R M* R> SM/REM

Příklad použití

```
FIXME:
```

Kód slova v Qurtus Forthu

Příklad 1. *:[90]

+

* *$Header: /home/radek/cvs/forth-book/dictionary/ansi/0120.plus,v 1.1 2003/12/28 18:21:57 radek Exp $*

Jméno

+ — „plus“, sečte dva prvky na zásobníku

CORE

Přehled

($n_1|u_1$ $n_2|u_2$ ⟶ $n_3|u_3$)

Definováno v: dpANS Forth 6.1.0120 CORE (http://forth.sourceforge.net/standard/dpans/dpans6.htm#6.1.0120)

FIXME: Toto je odstavec v tagu `refsynopsisdiv`.

Příklad použití

```
# $Id: dict-plus.ses,v 1.1 2003/02/02 12:39:15 radek Exp $
Gforth 0.6.2, Copyright (C) 1995-2003 Free Software Foundation, Inc.
Gforth comes with ABSOLUTELY NO WARRANTY; for details type ‘license’
1 2 + . 3  ok
589 32 + . 621  ok
BYE
```

+!

* *$Header: /home/radek/cvs/forth-book/dictionary/ansi/0130.plus-store,v 1.1 2003/12/28 18:21:57 radek Exp $*

Jméno

+! — „plus-store“, přičte hodnotu k buňce na adrese

Přehled

(n|u a-addr ⟶)

Popis

*FIXME:

Description

Add n|u to the single-cell number at a-addr.

Příklad použití

```
# $Id: dict-plus-store.ses,v 1.2 2003/12/28 18:21:58 radek Exp $
Gforth 0.6.2, Copyright (C) 1995-2003 Free Software Foundation, Inc.
Gforth comes with ABSOLUTELY NO WARRANTY; for details type ‘license’
VARIABLE v  ok
8 v !  ok
4 v +!  ok
v @ . 12  ok
BYE
```

Kód slova v Qurtus Forthu

Příklad 1. *:[90]

+LOOP

* *$Header: /home/radek/cvs/forth-book/dictionary/ansi/0140.plus-loop,v 1.1 2003/12/28 18:21:57 radek Exp $*

Jméno

+LOOP — FIXME: jednořádkový popis

CORE

Přehled

Překlad: **(do-sys ⟶)**

Běh: **(n ⟶) (R: loop-sys1 ⟶ loop-sys2)**

Definováno v: dpANS Forth 6.1.0140 CORE (http://forth.sourceforge.net/standard/dpans/dpans6.htm#6.1.0140)

V době překladu vyhradí místo pro budoucí výskyt LEAVE.

Při běhu přičte *n* k indexové proměnné smyčky.

Description

Append the run-time semantics given below to the current definition. Resolve the destination of all unresolved occurrences of LEAVE between the location given by do-sys and the next location for a transfer of control, to execute the words following +LOOP.

An ambiguous condition exists if the loop control parameters are unavailable. Add n to the loop index. If the loop index did not cross the boundary between the loop limit minus one and the loop limit, continue execution at the beginning of the loop. Otherwise, discard the current loop control parameters and continue execution immediately following the loop.

Příbuzná slova

DO, I, LEAVE

Příklad použití

```
# $Id: dict-plus-store.ses,v 1.2 2003/12/28 18:21:58 radek Exp $
Gforth 0.6.2, Copyright (C) 1995-2003 Free Software Foundation, Inc.
Gforth comes with ABSOLUTELY NO WARRANTY; for details type ‘license’
VARIABLE v  ok
8 v !  ok
4 v +!  ok
v @ . 12  ok
BYE
```

Kód slova v Qurtus Forthu

Příklad 1. *:[90]

```
,
```

* *$Header: /home/radek/cvs/forth-book/dictionary/ansi/0150.comma,v 1.1 2003/12/28 18:21:57 radek Exp $*

Jméno

`,` — „comma“

Přehled

`( x ⟶ )`

Definováno v: dpANS Forth 6.1.0150 CORE (http://forth.sourceforge.net/standard/dpans/dpans6.htm#6.1.0150)

Rezervuje jednu buňku v prostoru na nějž ukazuje `data-space` ukazovatel a uloží do ní hodnotu *x*.

Popis

FIXME:

Description

Reserve one cell of data space and store x in the cell. If the data-space pointer is aligned when , begins execution, it will remain aligned when , finishes execution. An ambiguous condition exists if the data-space pointer is not aligned prior to execution of **,**.

-

* *$Header: /home/radek/cvs/forth-book/dictionary/ansi/0160.minus,v 1.1 2003/12/28 18:21:57 radek Exp $*

Jméno

`-` — „minus“ odčítání

CORE

Přehled

`( n1|u1 n2|u2 ⟶ n3|u3 )`

Definováno v: dpANS Forth 6.1.0160 CORE (http://forth.sourceforge.net/standard/dpans/dpans6.htm#6.1.0160)

Popis

*FIXME:

Příklad použití

```
*FIXME:
```

Kód slova v Qurtus Forthu

Příklad 1. *:[90]

-TRAILING

* *$Header: /home/radek/cvs/forth-book/dictionary/ansi/0170.dash-trailing,v 1.1 2003/12/28 18:21:57 radek Exp $*

Jméno

`-TRAILING` — „dash-trailing“ **FIXME:**

STRING

Přehled

(c-addr u_1 ⟶ c-addr u_2)

Definováno v: dpANS Forth 6.1.0170 STRING (http://forth.sourceforge.net/standard/dpans/dpans6.htm#6.1.0170)

Popis

FIXME:

Description

If u1 is greater than zero, u2 is equal to u1 less the number of spaces at the end of the character string specified by c-addr u1. If u1 is zero or the entire string consists of spaces, u2 is zero.

.

* *$Header: /home/radek/cvs/forth-book/dictionary/ansi/0180.dot,v 1.1 2003/12/28 18:21:57 radek Exp $*

Jméno

. — „dot“ zobrazí/vytiskne číslo na vrcholu zásobníku

CORE

Přehled

(n ⟶)

Definováno v: dpANS Forth 6.1.0180 CORE (http://forth.sourceforge.net/standard/dpans/dpans6.htm#6.1.0180)

Popis

*FIXME:

Příklad použití

```
*FIXME:
```

Kód slova v Qurtus Forthu

Příklad 1. *:[90]

."

* *$Header: /home/radek/cvs/forth-book/dictionary/ansi/0190.dot-quote,v 1.1 2003/12/28 18:21:57 radek Exp $*
* *refentry id="dot-quote" xreflabel="."*

Jméno

." — „dot-quote“ zobrazí/vytiskne řetězec znaků až do znaku „"“

CORE

Přehled

Překlad: **("ccc<quote>" ⟶)**

Běh: **(⟶)**

Definováno v: dpANS Forth 6.1.0190 CORE (http://forth.sourceforge.net/standard/dpans/dpans6.htm#6.1.0190)

Viz.: .(

Popis

FIXME:

Příklad použití

```
FIXME:
```

Kód slova v Qurtus Forthu

Příklad 1. ." *:[90]

.(

* *$Header: /home/radek/cvs/forth-book/dictionary/ansi/0200.dot-paren,v 1.1 2003/12/28 18:21:57 radek Exp $*

Jméno

. (— „dot-paren“ tisk textu až do znaku „)“

CORE EXT

Přehled

Běh: **("ccc<paren>" ⟶) IMMEDIATE**

Vytiskne text za slovem až do znaku).

Definováno v: dpANS Forth 6.2.0200 CORE EXT (http://forth.sourceforge.net/standard/dpans/dpans6.htm#6.2.0200)

Viz.: ."

Popis

Tento příkaz je obdobou příkazu .". Zatímco ." se používá v definici slov, tedy kompiluje se do nově definovaných slov, tak .(je určen k tisku přímo.

.R

* *$Header: /home/radek/cvs/forth-book/dictionary/ansi/0210.dot-r,v 1.1 2003/12/31 00:08:29 radek Exp $*

Jméno

`.R` — „dot-r“ zobrazení čísla v poli zadané šířky

CORE EXT

Přehled

(n_1 n_2 ⟶)

Zobrazí číslo n_1 zarovnané doprava v poli širokém n_2 znaků. Je-li počet číslic v čísle větší než n_2, jsou vytištěny všechny číslice čísla a nejsou tištěny žádné úvodní mezery.

Definováno v: dpANS Forth 6.2.0210 CORE (http://forth.sourceforge.net/standard/dpans/dpans6.htm#6.2.0210)

Definováno v: dpANS Forth A.6.2.0210 CORE (http://forth.sourceforge.net/standard/dpans/dpans.htm#A.6.2.0210)

Popis

FIXME:

Definice

FIXME:

```
: ;
```

Ukázka použití

```
# $Id: DICT.dot-r.ses,v 1.1 2003/12/31 00:08:29 radek Exp $
Gforth 0.6.2, Copyright (C) 1995-2003 Free Software Foundation, Inc.
Gforth comes with ABSOLUTELY NO WARRANTY; for details type ‘license’
34 4 .R   34 ok
987654321 3 .R 987654321 ok
3245 8 .R     3245 ok
BYE
```

.S

* *$Header: /home/radek/cvs/forth-book/dictionary/ansi/0220.dot-s,v 1.2 2003/12/31 00:08:29 radek Exp $*

Jméno

`.S` — „dot-s“ vytiskne obsah celého datového zásobníku bez změny zásobníku

TOOLS

Přehled

(⟶)

Definováno v: dpANS Forth 15.6.1.0220 TOOLS (http://forth.sourceforge.net/standard/dpans/dpans15.htm#15.6.1.0220)

Popis

Vytiskne obsah zásodníku, aniž by byl zásobník jakkoliv modifikován. Slouží hlavně při interaktivní práci a ladění, kdy se potřebujeme podívat co všechno se v zásobníku nachází.

Description

Copy and display the values currently on the data stack. The format of the display is implementation-dependent. .S may be implemented using pictured numeric output words. Consequently, its use may corrupt the transient region identified by #>.

Příklad použití

```
4 5 6 .S

*FIXME:
```

Kód slova v Qurtus Forthu

Příklad 1. Kód slova .s v Qurtus Forthu *:[90]

```
;*FIXME:
    = BE30: ....
    = BE32: 022E 5300                       DB      2, ".S"
....= BE36: 4EAA 82C0           .S:         JSR     $-7D40(CS)     ;= DEPTH
....= BE3A: 4A47                            TST.W   TOS
....= BE3C:                                 BNE.S   $BE4C
    = BE3E:                                 MOVE.W  (A4)+, TOS  ;= DROP
    = BE40:                                 LEA     $40(PC), A0
    = BE44:                                 MOVEQ.L #12, D0
      BE46:                                 JSR     $-77D2(CS)
      BE4A:                                 BRA.S   $BE7C
      BE4C:                                 BGE.S   $BE58
      BE4E:                                 MOVE.W  TOS, -(SP)  ;= DUP
```

```
BE50:                                   MOVE.W  #-4, TOS
BE54:                                   JSR     $-49CC(CS)  ;= THROW
BE58:                                   MOVE.W  TOS, -(SP)  ;= DUP
BE5A:                                   MOVEQ.L #0, TOS
BE5C:                                   MOVE.W  TOS, -(SP)  ;= DUP
BE5E:                                   SWAP    TOS
BE60:                                   MOVE.W  TOS, -(SP)  ;= DUP
BE62:                                   MOVEQ.L #$3C, TOS
BE64:                                   JSR     $-7748(CS)  ;= EMIT
BE68:                                   JSR     $-6236(CS)  ;= D.R
BE6C:                                   MOVE.W  TOS, -(SP)  ;= DUP
BE6E:                                   MOVEQ.L #$3E, TOS
BE70:                                   JSR     $-7748(CS)  ;= EMIT
BE74:                                   JSR     $-7730(CS)  ;= SPACE
BE78:                                   JSR     $-41EE(CS)  ;
BE7C:                                   JSR     $-78DC(CS)  ;= CR
BE80:                                   BRA.S   $BE8E
BE82:                                   MOVE.L  $61(A3, D7.W), A0
BE86:                                   BLS.S   $BEF3
BE88:                                   MOVE.L  -(A5), A0
BE8A:                                   BLT.S   $BEFC
BE8C:                                   MOVEQ.L #$79, D2
BE8E:                                   RTS                 ;= EXIT
```

/

* *$Header: /home/radek/cvs/forth-book/dictionary/ansi/0230.slash,v 1.1 2003/12/28 18:21:57 radek Exp $*

Jméno

/ — „slash“, dělení

CORE

Přehled

(n_1 n_2 ⟶ n_3)

Definováno v: dpANS Forth 6.1.0230 CORE (http://forth.sourceforge.net/standard/dpans/dpans6.htm#6.1.0230)

Description

Divide n1 by n2, giving the single-cell quotient n3. An ambiguous condition exists if n2 is zero. If n1 and n2 differ in sign, the implementation-defined result returned will be the same as that returned by either the phrase >R S>D R> FM/MOD SWAP DROP or the phrase >R S>D R> SM/REM SWAP DROP .

Příklad použití

```
# $Id: dict-plus.ses,v 1.1 2003/02/02 12:39:15 radek Exp $
Gforth 0.6.2, Copyright (C) 1995-2003 Free Software Foundation, Inc.
Gforth comes with ABSOLUTELY NO WARRANTY; for details type ‘license’
1 2 + . 3  ok
```

```
589 32 + . 621  ok
BYE
```

/MOD

* *$Header: /home/radek/cvs/forth-book/dictionary/ansi/0240.slash-mod,v 1.1 2003/12/28 18:21:57 radek Exp $*

Jméno

`/MOD` — „slash-mod“, dělení se zbytkem

CORE

Přehled

(n_1 n_2 ⟶ n_3 n_4)

Definováno v: dpANS Forth 6.1.0240 CORE (http://forth.sourceforge.net/standard/dpans/dpans6.htm#6.1.0240)

Description

Divide n1 by n2, giving the single-cell remainder n3 and the single-cell quotient n4. An ambiguous condition exists if n2 is zero. If n1 and n2 differ in sign, the implementation-defined result returned will be the same as that returned by either the phrase `>R S>D R> FM/MOD` or the phrase `>R S>D R> SM/REM`.

Příklad použití

```
# $Id: dict-plus.ses,v 1.1 2003/02/02 12:39:15 radek Exp $
Gforth 0.6.2, Copyright (C) 1995-2003 Free Software Foundation, Inc.
Gforth comes with ABSOLUTELY NO WARRANTY; for details type ‘license’
1 2 + . 3  ok
589 32 + . 621  ok
BYE
```

/STRING

* *$Header: /home/radek/cvs/forth-book/dictionary/ansi/0245.slash-string,v 1.1 2003/12/31 00:08:29 radek Exp $*

Jméno

`/STRING` — „slash-string“ **FIXME:**popis

STRING

Přehled

(c-addr$_1$ u$_1$ n ⟶ c-addr$_2$ u$_2$)

Definováno v: dpANS Forth 17.6.1.0245 CORE (http://forth.sourceforge.net/standard/dpans/dpans17.htm#17.6.1.0245)

Definováno v: dpANS Forth A.17.6.1.0245 CORE (http://forth.sourceforge.net/standard/dpans/dpans.htm#A.17.6.1.0245)

Popis

FIXME:

Definice

FIXME:

```
: ;
```

0<

* *$Revision: 1.1 $ $Date: 2003/12/28 18:21:57 $*

Jméno

`0<` — „zero-less“ vrátí true, je-li v TOS záporné číslo

CORE

Přehled

(n ⟶ false | true)

Definováno v: dpANS Forth 6.1.0250 CORE (http://forth.sourceforge.net/standard/dpans/dpans6.htm#6.1.0250)

Popis

*FIXME:

Příklad použití

```
*FIXME:
```

Kód slova v Qurtus Forthu

Příklad 1. *:[90]

```
....= 92F8:        TST.W   TOS
    = 92FA:        SLT     TOS
    = 92FC:        EXT.W   TOS
    = 92FE:        RTS         ;= EXIT
```

0<>

* *$Header: /home/radek/cvs/forth-book/dictionary/ansi/0260.zero-not-equals,v 1.1 2003/12/31 00:08:29 radek Exp $*

Jméno

`0<>` — „zero-not-equals“ test nenulovosti

CORE EXT

Přehled

`( x ⟶ flag )`

Je-li *x* nenulové, vrátí `true`, je-li nulové vrátí `false`.

Definováno v: dpANS Forth 6.1.0260 CORE (http://forth.sourceforge.net/standard/dpans/dpans6.htm#6.1.0260)

Popis

FIXME:

Definice

FIXME:

```
:  ;
```

0=

* *$Header: /home/radek/cvs/forth-book/dictionary/ansi/0270.zero-equals,v 1.1 2003/12/28 18:21:57 radek Exp $*

Jméno

`0=` — „zero-equals“ vrátí true, je-li v TOS nula

CORE

Přehled

`( n ⟶ false | true)`

Definováno v: dpANS Forth 6.1.0270 CORE (http://forth.sourceforge.net/standard/dpans/dpans6.htm#6.1.0270)

Popis

*FIXME:

Příklad použití

```
*FIXME:
```

Kód slova v Qurtus Forthu

Příklad 1. 0= *:[90]

```
....= 9322:        TST.W   TOS
    = 9324:        SEQ     TOS
    = 9326:        EXT.W   TOS
    = 9328:        RTS         ;= EXIT
```

0>

* *$Header: /home/radek/cvs/forth-book/dictionary/ansi/0280.zero-greater,v 1.1 2003/12/28 18:21:57 radek Exp $*

Jméno

`0>` — „zero-grater“ vrátí true, je-li v TOS kladné nenulové číslo

CORE

EXT

Přehled

`( n ⟶ false | true)`

Definováno v: dpANS Forth 6.2.0280 CORE (http://forth.sourceforge.net/standard/dpans/dpans6.htm#6.2.0280)

Popis

*FIXME:

Příklad použití

```
*FIXME:
```

Kód slova v Qurtus Forthu

Příklad 1. 0> *:[90]

```
....= 9314:        TST.W   TOS
    = 9316:        SGT     TOS
    = 9318:        EXT.W   TOS
    = 931A:        RTS         ;= EXIT
```

1+

* *$Header: /home/radek/cvs/forth-book/dictionary/ansi/0290.one-plus,v 1.1 2003/12/31 00:08:29 radek Exp $*

Jméno

`1+` — „one-plus“ zvětší číslo na vrcholu zásobníku o jedničku

CORE

Přehled

`( n`$_1$`|u`$_1$ ⟶ **`n`$_2$`|u`$_2$ **`)`

Přičte jedničku (1) k $n_1|u_1$ a vrátí jako $n_2|u_2$.

Definováno v: dpANS Forth 6.1.0290 CORE (http://forth.sourceforge.net/standard/dpans/dpans6.htm#6.1.0290)

Popis

FIXME:

Definice

FIXME:

```
: ;
```

1-

* *$Header: /home/radek/cvs/forth-book/dictionary/ansi/0300.one-minus,v 1.1 2003/12/31 00:08:29 radek Exp $*

Jméno

`1-` — „one-minus“ zmenší číslo na vrcholu zásobníku o jedničku

CORE

Přehled

```
( n1|u1 ⟶ n2|u2 )
```

Odečte jedničku (1) od $n_1|u_1$ a vrátí jako $n_2|u_2$.

Definováno v: dpANS Forth 6.1.0300 CORE (http://forth.sourceforge.net/standard/dpans/dpans6.htm#6.1.0300)

Popis

FIXME:

Definice

FIXME:

```
: ;
```

2CONSTANT

* *$Header: /home/radek/cvs/forth-book/dictionary/ansi/0360.two-constant,v 1.1 2003/12/28 18:21:57 radek Exp $*

Jméno

`2CONSTANT` — „two-constant“ definice konstanty velké dvě buňky

DOUBLE

Přehled

`( n. "`*`jméno`*`" ⟶ )`

`jméno` **`( ⟶ n. )`**

Definováno v: dpANS Forth 8.6.1.0360 CORE (http://forth.sourceforge.net/standard/dpans/dpans8.htm#8.6.1.0360)

Definováno v: dpANS Forth A.8.6.1.0360 CORE (http://forth.sourceforge.net/standard/dpans/dpans.htm#A.8.6.1.0360)

Popis

*FIXME:

Příklad použití

```
*FIXME:
```

Description

Skip leading space delimiters. Parse name delimited by a space. Create a definition for name with the execution semantics defined below. name is referred to as a two-constant. name Execution: `( -- x1 x2 )` Place cell pair x1 x2 on the stack.

2DROP

* *$Header: /home/radek/cvs/forth-book/dictionary/ansi/0370-two-drop,v 1.1 2003/12/31 00:08:29 radek Exp $*

Jméno

2DROP — „two-drop“ odstranění dvou buněk ze zásobníku

CORE

Přehled

(x_1 x_2 ⟶)

Definováno v: dpANS Forth 6.1.0370 CORE (http://forth.sourceforge.net/standard/dpans/dpans6.htm#6.1.0370)

Popis

FIXME:

Definice

```
: 2drop DROP DROP ;
```

Příklad použití

```
*FIXME:
```

2DUP

* *$Header: /home/radek/cvs/forth-book/dictionary/ansi/0380.two-dupe,v 1.1 2003/12/28 18:21:57 radek Exp $*

Jméno

2DUP — Zdvojení dvoubuňky *Duplicate cell pair*

CORE

Přehled

(n1 n2 ⟶ n1 n2 n1 n2)

Definováno v: dpANS Forth 6.1.0380 CORE (http://forth.sourceforge.net/standard/dpans/dpans6.htm#6.1.0380)

Popis

FIXME:

Definice

```
: 2dup OVER OVER ;
```

Příklad použití

```
*FIXME:
```

2OVER

* *$Header: /home/radek/cvs/forth-book/dictionary/ansi/0400.two-over,v 1.1 2003/12/31 00:08:29 radek Exp $*

Jméno

2OVER — „two-over“ zkopíruje druhý pár buněk pod vrcholem zásobníku na vrchol

CORE

Přehled

(x_1 x_2 x_3 x_4 ⟶ x_1 x_2 x_3 x_4 x_1 x_2)

Zkopíruje druhý pár buňek pod vrcholem zásobníku na vrchol. Je to obdoba slova OVER která pracuje s páry buňek, dvojtými buňkami.

Definováno v: dpANS Forth 6.1.0400 CORE (http://forth.sourceforge.net/standard/dpans/dpans6.htm#6.1.0400)

Popis

FIXME:

Definice

```
: 2dup OVER OVER ;
```

Příklad použití

```
*FIXME:
```

2SWAP

* *$Header: /home/radek/cvs/forth-book/dictionary/ansi/0430.two-swap,v 1.1 2003/12/28 18:21:57 radek Exp $*

Jméno

2SWAP — **FIXME:**Exchange the top two cell pairs.

Přehled

(a,a b,b ⟶ b,b a,a,)

Definováno v: dpANS Forth 6.1.0420 CORE (http://forth.sourceforge.net/standard/dpans/dpans6.htm#6.1.0420)

Popis

FIXME:

Description

double-cell swap, see SWAP and 2DUP simulate:

```
: 2SWAP LOCALS| B1 B2 A1 A2 | B2 B1 A2 A1 ;
```

Příklad použití

.

Kód slova v Qurtus Forthu

Příklad 1. Kód slova 2SWAP *:[80]

```
02DE= 82E0: 82D0                            DW      $82D0           ; link to SWAP
02E0= 82E2: 0532 5357 4150                  DB    5, "2SWAP"
02E6= 82E8: 202C 0002           2SWAP:      MOVE.L  2(SP), D0       ;=
02EA= 82EC: 4847                            SWAP    TOS             ;
02EC= 82EE: 3E14                            MOVE.W  (SP), TOS
02EE= 82F0: 3880                            MOVE.W  D0, (SP)
```

```
02F0= 82F2: 2947 0002                   MOVE.L  TOS, 2(SP)
02F4= 82F6: 4840                        SWAP    D0              ;
02F6= 82F8: 3E00                        MOVE.W  D0, TOS
02F8= 82FA: 4E75                        RTS                     ;= EXIT
```

2VARIABLE

* *$Header: /home/radek/cvs/forth-book/dictionary/ansi/0440.two-variable,v 1.1 2003/12/28 18:21:57 radek Exp $*

Jméno

2VARIABLE — „two-variable“ Vytvoření proměnné veliké dvě buňky.

DOUBLE

Přehled

("*jméno*" ⟶)
***jméno* (⟶ addr)**

Popis

*FIXME:

Description

Skip leading space delimiters. Parse name delimited by a space. Create a definition for name with the execution semantics defined below. Reserve two consecutive cells of data space. name is referred to as a two-variable. name Execution: (-- a-addr) a-addr is the address of the first (lowest address) cell of two consecutive cells in data space reserved by 2VARIABLE when it defined name. A program is responsible for initializing the contents.

See: VARIABLE

Příklad použití

```
*FIXME:
```

:

* *$Header: /home/radek/cvs/forth-book/dictionary/ansi/0450.colon,v 1.1 2003/12/28 18:21:57 radek Exp $*

Jméno

: — „colon“, překladač, zahájení definice nového slova — DOCON

Přehled

("název slova" ⟶)

Definováno v: dpANS Forth 6.1.0450 CORE (http://forth.sourceforge.net/standard/dpans/dpans6.htm#6.1.0450)

Popis

Slovo : otevírá definici nového slova. Jako první následující je jméno tohoto nového slova a za ním definice ukončená slovem ;. V assembleru bývá pojmenováno často docon. Je to tak proto, že znak : nemůže sloužit jako jméno, neboť má obvykle jiný význam (v assembleru).

Příklad použití

```
: double DUP + ;
```

```
# $Id: dict-docon.ses,v 1.1 2002/12/18 23:25:07 radek Exp $
Gforth 0.6.2, Copyright (C) 1995-2003 Free Software Foundation, Inc.
Gforth comes with ABSOLUTELY NO WARRANTY; for details type ‘license’
: double DUP 2 + ;    ok
2 double . 4  ok
BYE
```

```
# $Id: dict-docon2.ses,v 1.1 2002/12/18 23:25:07 radek Exp $
Gforth 0.6.2, Copyright (C) 1995-2003 Free Software Foundation, Inc.
Gforth comes with ABSOLUTELY NO WARRANTY; for details type ‘license’
: double DUP 2 + ;  ok
2 double . 4  ok
BYE
```

Kód slova v Qurtus Forthu

Příklad 1. Kód slova : v Qurtus Forthu *:[90]*

```
;ADDR  WORDS           LABEL        MNEMO   ARGS
;----  --------------  -----------  ------- --------------------
969E: (8000)                          DW
96A0: 013A                            DB      1, ":"
96A2: 4EAA 9622            DOCON:     JSR     $-69DE(CS)
96A6: 4EAA 957E                       JSR     $-6A82(CS)      ;= (header)
96AA: 4E75                            RTS                     ;= EXIT
```

Kód slova v FIG6502

```
;                                       :
;                                       SCREEN 33 LINE 2
;
L832      .BYTE $C1,$BA
          .WORD L813     ; link to C!
COLON     .WORD DOCOL
          .WORD QEXEC
          .WORD SCSP
          .WORD CURR
          .WORD AT
          .WORD CON
          .WORD STORE
          .WORD CREAT
          .WORD RBRAC
          .WORD PSCOD
;
DOCOL     LDA IP+1
          PHA
          LDA IP
          PHA
          JSR TCOLON     ; mark the start of a traced : def.
          CLC
          LDA W
          ADC #2
          STA IP
          TYA
          ADC W+1
          STA IP+1
          JMP NEXT
```

;

* *$Header: /home/radek/cvs/forth-book/dictionary/ansi/0460.semicolon,v 1.1 2003/12/28 18:21:57 radek Exp $*

Jméno

`;` — ukončení/uzavření definice slova

Přehled

`: ;  ( ⟶ ) ;`

Popis

*FIXME:

Příklad použití

```
# $Id: dict-plus-store.ses,v 1.2 2003/12/28 18:21:58 radek Exp $
Gforth 0.6.2, Copyright (C) 1995-2003 Free Software Foundation, Inc.
Gforth comes with ABSOLUTELY NO WARRANTY; for details type ‘license’
VARIABLE v  ok
8 v !  ok
4 v +!  ok
v @ . 12  ok
BYE
```

Kód slova v Qurtus Forthu

Příklad 1. *:[90]

< number

* *Header: /home/radek/cvs/forth-book/db-dict/<#,v 1.1 2003/02/02 12:39:12 radek Exp*
* *$Revision: 1.1 $ $Date: 2003/12/28 18:21:57 $*

Jméno

`< number` — FIXME: jednořádkový popis

Přehled

```
: <#   ( ⟶ ) ;
```

Popis

Definováno v: dpANS Forth 6.1.0490 CORE (http://forth.sourceforge.net/standard/dpans/dpans6.htm#6.1.0490)

Description

Initialize the pictured numeric output conversion process.

See also HOLD for old-style forth-formatting words and PRINTF of the C-style formatting - this word does initialize the pictured numeric output space.

Příklad použití

```
*FIXME:
```

Kód slova v Qurtus Forthu

Příklad 1. *:[90]

Kód slova v FIG6502

```
;                                       <#
;                                       SCREEN 75 LINE 3
;
L3460     .BYTE $82,'<',$A3
          .WORD L3442    ; link to SPACES
BDIGS     .WORD DOCOL
          .WORD PAD
          .WORD HLD
          .WORD STORE
          .WORD SEMIS
```

Příklad 2. *:[90]

>IN

* *$Header: /home/radek/cvs/forth-book/dictionary/ansi/0560.to-in,v 1.1 2003/12/28 18:21:57 radek Exp $*

Jméno

`>IN` — offset/posunutí ve vstupním bufferu TIBtib

Přehled

`( ⟶ a-addr )`

`a-addr` je adresa buňky obsahující offset od začátku vstupního bufferu. Ukazuje na začátek *parse area*.

Popis

Definováno v: dpANS Forth 6.1.0560 CORE (http://forth.sourceforge.net/standard/dpans/dpans6.htm#6.1.0560)

Příklad použití

FIXME:

Kód slova v Qurtus Forthu

Příklad 1. Kód slova >IN **:[90]*

```
;ADDR  WORDS           LABEL        MNEMO   ARGS
;----  --------------  -----------  ------- --------------------
81CA: 81B2 831C                        DW      $81B2, $831C          ; link to SOURCE
81CC: 833E 494E                        DB      $80+3, ">IN"
81D0: 3907                 to-in:      MOVE    D7, -(A4)          ;= DUP
81D2: 3E3C 0196                        MOVE    #406, D7           ;
81D6: 4E75                             RTS                        ;= EXIT
```

>R

* *$Header: /home/radek/cvs/forth-book/dictionary/ansi/0580.to-r,v 1.1 2003/12/28 18:21:57 radek Exp $*

Jméno

>R — popis

CORE

Přehled

Interpretace: *význam není definován*

Běh: **(x ⟶) (R: ⟶ x)**

Definováno v: dpANS Forth 6.1.0580 CORE (http://forth.sourceforge.net/standard/dpans/dpans6.htm#6.1.0580)

Přesune vrchol zásobníku na vrchol zásobníku návratových adres.

Popis

FIXME:

Description

save the value onto the return stack. The return stack must be returned back to clean state before an exit and you should note that the return-stack is also touched by the DO ... WHILE loop. Use R> to clean the stack and R@ to get the last value put by >R

?DUP

* *$Header: /home/radek/cvs/forth-book/dictionary/ansi/0630.question-dupe,v 1.1 2003/12/31 00:08:29 radek Exp $*

Jméno

`?DUP` — „question-dupe“ podmíněné zdvojení

CORE

Přehled

```
( x ⟶ 0 | x x )
```

Je-li prvek na zásobníku nenulový, bude zduplikován.

Definováno v: dpANS Forth 6.1.0630 CORE (http://forth.sourceforge.net/standard/dpans/dpans6.htm#6.1.0630)

Popis

FIXME:

@

* *$Header: /home/radek/cvs/forth-book/dictionary/ansi/0650.fetch,v 1.1 2003/12/28 18:21:57 radek Exp $*

Jméno

@ — „fetch“, uloží na vrchol zásobníku hodnotu určenou adresou (FETCH)

Přehled

```
: @   ( addr ⟶ u ) ;
```

Popis

Definováno v: dpANS Forth 6.1.0650 CORE (http://forth.sourceforge.net/standard/dpans/dpans6.htm#6.1.0650)

Vyzvedne číslo ze zásobníku (TOS) použije jako adresu a obsah buňky na této adrese uloží na zásobník (TOS).

Příklad použití

```
# $Id: dict-fetch.ses,v 1.1 2002/12/18 23:25:08 radek Exp $
Gforth 0.6.2, Copyright (C) 1995-2003 Free Software Foundation, Inc.
Gforth comes with ABSOLUTELY NO WARRANTY; for details type ‘license’
VARIABLE A  ok
45 A !  ok
A @ . 45  ok
BYE
```

Kód slova v Qurtus Forthu

Příklad 1. *:[72]

```
;ADDR  WORDS           LABEL        MNEMO   ARGS
;----  --------------  -----------  ------- --------------------
92C4
8140
3E35 7000      FETCH:    MOVE.W $0(A5,D7.w),D7     ;= @
4E75                     RTS                       ;= EXIT
```

AGAIN

* *$Header: /home/radek/cvs/forth-book/dictionary/ansi/0700.again,v 1.1 2003/12/28 18:21:57 radek Exp $*

Jméno

`AGAIN` — převede řízení na jiné místo

CORE EXT

Přehled

Překlad: **(dest ⟶)**

Běh: **(⟶)**

Definováno v: dpANS Forth 6.2.0700 CORE (http://forth.sourceforge.net/standard/dpans/dpans6.htm#6.2.0700)

Popis

Převede řízení/vykonávání kódu na uvedené místo.

Toto slovo je možno použít jen ve stavu/fázi kompilace. T.j. při definování nového slova.

Description

Append the run-time semantics given below to the current definition, resolving the backward reference dest. Run-time: (--) Continue execution at the location specified by dest. If no other control flow words are used, any program code after AGAIN will not be executed.

Kód slova

Příklad 1. Kód slova again

```
BC16: 8000                            DW      $8000
BC18: 4541 4741 494E ....  'again':   DB      $40+5, "again"
BC1E: 4EAA BB92                       JSR     $BB92(CS)
BC22: 4EAA BBCE                       JSR     $BBCE(CS)
BC26: 4E75                            RTS                   ;= EXIT
```

Příklad použití

```
*FIXME:
```

ALSO

* *$Header: /home/radek/cvs/forth-book/dictionary/ansi/0715.also,v 1.1 2003/12/28 18:21:57 radek Exp $*

Jméno

ALSO — jednořádkový popis

SEARCH EXT

Přehled

(⟶)

Definováno v: dpANS Forth 16.6.2.0715 CORE (http://forth.sourceforge.net/standard/dpans/dpans16.htm#16.6.2.0715)

Popis

*FIXME:

Description

Transform the search order consisting of widn, ... wid2, wid1 (where wid1 is searched first) into widn, ... wid2, wid1, wid1. An ambiguous condition exists if there are too many word lists in the search order.

Příklad použití

```
*FIXME:
```

Kód slova v Qurtus Forthu

Příklad 1. Kód slova currentx *:[90]

```
;
;021E= 8220: 8210                              DW     $8210              ; link to BASE
;0220= 8222: 8863 7572 7265 6E74               DB     $80+8, "currentx", 0
;0228=       7800
....= B4EE: ....                 also:        JSR     $-4B7E(CS)    ;= GET-ORDER
....= B4F2:                                   MOVE.W  (A4), -(A4)
....= B4F4:                                   ADDQ.W   #$1, TOS          ;= 1+
....= B4F6:                                   JSR      $-4BBA(CS)        ;= SET-ORDER
B4FA:                                 RTS                         ;= EXIT
```

BASE

* *$Header: /home/radek/cvs/forth-book/dictionary/ansi/0750.base,v 1.1 2003/12/28 18:21:57 radek Exp $*

Jméno

BASE — jednořádkový popis

Přehled

slovo (zásobníkový efekt ⟶)

Definováno v: dpANS Forth 6.1.0750 CORE (http://forth.sourceforge.net/standard/dpans/dpans6.htm#6.1.0750)

Popis

FIXME:

Příklad použití

FIXME:

Kód slova v Qurtus Forthu

Příklad 1. Kód slova BASE *:[90]

```
;
020E= 8210: 8200                            DW     $8200             ; link to {xref linkend="QF.event"/}
0210= 8212: 8442 4153 4500                  DB     $80+4, "BASE", 0
0216= 8218: 3907                BASE:       MOVE.W  TOS, -(SP)       ;= 376
0218= 821A: 3E3C 0178                       MOVE.W  #376, TOS        ;+
021C= 821E: 4E75                            RTS                      ;= EXIT
```

BYE

* *$Header: /home/radek/cvs/forth-book/dictionary/ansi/0830.bye,v 1.1 2003/12/28 18:21:57 radek Exp $*

Jméno

BYE — ukončení práce v prostředí forthu a návrat do systému

Přehled

(⟶)

Protože slovo ukončuje práci prostředí forthu, nedojde nikdy k návratu z něj.

Popis

Definováno v: dpANS Forth 15.6.2.0830 EXT (http://forth.sourceforge.net/standard/dpans/dpans15.htm#15.6.2.0830)

Ukončí práci v prostředí Forthu, a provede návrat do systému odkud jsme Forth spustili.

Kód slova

Příklad 1. Kód slova bye

```
;
8112: 80EA                                  DW      $80EA
8114:                                       DB      3, "BYE"
8118:                         "BYE":        PEA     16(PC)
811C:                                       SYSTRAP EvtAddEventToQue...
8120:                                       ADDQ.L  #4, RP
```

```
8122:                                   JSR     ...(CS)
8126:                                   MOVE    (SP)+, TOS
8128:                                   BRA     $+22  ;=8140
812A:                                   DW      4,0,0,0, $108,0,8,0, 0,0,0
8140:                                   RTS
```

Příklad použití

```
256 true HwrBacklight
```

Strojový kód slova/procedury v Qurtus Forthu

Příklad 2. BYE

```
0110= 8112: 80EA                            DW      $80EA           ; link to {xref linkend=
0112= 8114:                                 DB      3, "BYE"
0116= 8118:                       BYE:      PEA     16(PC)
011A=                                       SYSTRAP EvtAddEventToQue...
011E=                                       ADDQ.L  #4, RP          ;= UNLOOP
0120=                                       JSR     -29302(CS)      ;= EKEY
0124=                                       MOVE    (SP)+, TOS      ;= DROP
0126=                                     .--BRA     $+22 = 013E
                                          |  ; Popis události přidané do fronty událostí
0128=                                     |  DW      $0004, $0000, $0000, $0000
0130=                                     |  DW      $0108, $0000, $0008, $0000
0138=                                     |  DW      $0000, $0000, $0000
013E=                                     '->RTS                                          ;=
```

CELL+

* *$Header: /home/radek/cvs/forth-book/dictionary/ansi/0880.cell-plus,v 1.1 2003/12/28 18:21:57 radek Exp $*

Jméno

`CELL+` — „cell-plus“ FIXME: jednořádkový popis

Přehled

`( a-addr1 ⟶ a-addr2 ) ;`

Definováno v: dpANS Forth 6.1.0880 CORE (http://forth.sourceforge.net/standard/dpans/dpans6.htm#6.1.0880)

Přičte velikost v buňkách k adrese na zásobníku.

Popis

*FIXME:

Příklad použití

```
# $Id: dict-plus-store.ses,v 1.2 2003/12/28 18:21:58 radek Exp $
Gforth 0.6.2, Copyright (C) 1995-2003 Free Software Foundation, Inc.
Gforth comes with ABSOLUTELY NO WARRANTY; for details type ‘license’
VARIABLE v  ok
8 v !  ok
4 v +!  ok
v @ . 12  ok
BYE
```

Kód slova v Qurtus Forthu

Příklad 1. *:[90]

CELLS

* *$Header: /home/radek/cvs/forth-book/dictionary/ansi/0890.cells,v 1.1 2003/12/28 18:21:57 radek Exp $*

Jméno

`CELLS` — FIXME: jednořádkový popis

Přehled

(n1 ⟶ n2) ;

Definováno v: dpANS Forth 6.1.0890 CORE (http://forth.sourceforge.net/standard/dpans/dpans6.htm#6.1.0890)

Definováno v: dpANS Forth A.6.1.0890 CORE (http://forth.sourceforge.net/standard/dpans/dpans.htm#A.6.1.0890)

Popis

*FIXME:

Příklad použití

```
# $Id: dict-plus-store.ses,v 1.2 2003/12/28 18:21:58 radek Exp $
Gforth 0.6.2, Copyright (C) 1995-2003 Free Software Foundation, Inc.
Gforth comes with ABSOLUTELY NO WARRANTY; for details type ‘license’
VARIABLE v  ok
8 v !  ok
4 v +!  ok
v @ . 12  ok
BYE
```

Kód slova v Qurtus Forthu

Příklad 1. *:[90]

CREATE

* *$Header: /home/radek/cvs/forth-book/dictionary/ansi/1000.create,v 1.1 2003/12/28 18:21:57 radek Exp $*

Jméno

CREATE — vytvoří hlavičku slova na slovníku

CORE

Přehled

CREATE ("název" ⟶)

Definováno v: dpANS Forth 6.1.1000 CORE (http://forth.sourceforge.net/standard/dpans/dpans6.htm#6.1.1000)

Popis

Přečte ze vstupního bufferu název nového slova a vytvoří jeho hlavičku ve slovníku.

Strojový kód slova/procedury v Qurtus Forthu

Příklad 1. CREATE

```
                                                        ;*FIXME:
18D6= 98D8: 98AE (____)                     DW      $98AE ($____)   ; link to
18D8= 98DA: 0643 5245 4154 4500             DB      $6 , "CREATE", 0
18E0= 98E2: 4EAA 96A2           CREATE:     JSR     $-695E(A2)                      ;=
```

```
18E4= 98E6: 4EAA 81E0                    JSR     $-7E20(A2)        ;= HERE
18E8= 98EA: 4EAA 97B8                    JSR     $-6848(A2)
18EC= 98EE: 3907                         MOVE.W  TOP, -(A4)
18EE= 98F0: 3E3C 4EAA                    MOVE.W  #$4EAA, TOP
18F2= 98F4: 4EAA 96C8                    JSR     -26963(A2)
18F6= 98F8: 3907                         MOVE.W  TOP, -(A4)
18F8= 98FA: 3E3C 80E8                    MOVE.W  #-32536, TOP
18FC= 98FE: 3B6D 01A6 018E               MOVE.W  422(A5), 398(A5)
1902= 9904: 4EAA 9138                    JSR     -28360(A2)
1906= 9908: 4EAA 971A                    JSR     -26854(A2)
190A= 990C: 4E75                         RTS                       ;= EXIT
```

Příklad použití

```
*FIXME:
```

D+

* *$Header: /home/radek/cvs/forth-book/dictionary/ansi/1040.d-plus,v 1.1 2003/12/28 18:21:57 radek Exp $*

Jméno

D+ — sčítání v dvojnásobé aritmetice

DOUBLE

Přehled

($d_1|ud_1$ $d_2|ud_2$ ⟶ $d_3|ud_3$)

Protože slovo ukončuje práci prostředí forthu, nedojde nikdy k návratu z něj.

Definováno v: dpANS Forth 8.6.1.1040 DOUBLE (http://forth.sourceforge.net/standard/dpans/dpans8.htm#8.6.1.1040)

Popis

FIXME:

DEPTH

* *$Header: /home/radek/cvs/forth-book/dictionary/ansi/1200.depth,v 1.1 2003/12/28 18:21:57 radek Exp $*

Jméno

DEPTH — hloubka zásobníku, počet buněk uložených na zásobník

Přehled

(⟶ n)

Definováno v: dpANS Forth 6.1.1200 CORE (http://forth.sourceforge.net/standard/dpans/dpans6.htm#6.1.1200)

Popis

Zjistí aktuální počet buněk v zásobníku a toto číslo uloží do zásodníku.

Příklad použití

```
1 2 3 DEPTH .s Enter<4> 1 2 3 3
 4 8 9 DEPTH .s Enter<8> 1 2 3 3 4 8 9 7
```

Kód slova v Qurtus Forthu

Příklad 1. Kód slova DEPTH *:[80]

```
02B6= 82B8: 82A2                            DW      $82A2           ; link to {xref linkend="QF.MainFormID
02B8= 82BA: 0544 4550 5448                  DB      5, "DEPTH"
02BE= 82C0: 200C                DEPTH:      MOVE.L  A4, D0          ;=
02C0= 82C2: 222D 0012                       MOVE.L  18(A5), D1      ;+
02C4= 82C6: 9280                            SUB.L   D0, D1
02C6= 82C8: E289                            LSR.L   #1, D1
02C8= 82CA: 3907                            MOVE.W  TOS, -(A4)
02CA= 82CC: 3E01                            MOVE.W  D1, D7
02CC= 82CE: 4E75                            RTS                     ;= EXIT
```

DO

* *$Header: /home/radek/cvs/forth-book/dictionary/ansi/1240.do,v 1.1 2003/12/28 18:21:57 radek Exp $*

Jméno

DO — FIXME: jednořádkový popis

Přehled

(⟶)

Definováno v: dpANS Forth 6.1.1240 CORE (http://forth.sourceforge.net/standard/dpans/dpans6.htm#6.1.1240)

Popis

*FIXME:

Příklad použití

```
# $Id: dict-plus-store.ses,v 1.2 2003/12/28 18:21:58 radek Exp $
Gforth 0.6.2, Copyright (C) 1995-2003 Free Software Foundation, Inc.
Gforth comes with ABSOLUTELY NO WARRANTY; for details type ‘license’
VARIABLE v  ok
8 v !  ok
4 v +!  ok
v @ . 12  ok
BYE
```

Kód slova v Qurtus Forthu

Příklad 1. *:[90]

DROP

* *$Header: /home/radek/cvs/forth-book/dictionary/ansi/1260.drop,v 1.1 2003/12/28 18:21:57 radek Exp $*

Jméno

DROP — Odstraní prvek z vrcholu zásobníku

Přehled

(x ⟶)

Definováno v: dpANS Forth 6.1.1260 CORE (http://forth.sourceforge.net/standard/dpans/dpans6.htm#6.1.1260)

Popis

FIXME:

Description

Just drop the word on the top of stack, see DUP

DUP

* *$Header: /home/radek/cvs/forth-book/dictionary/ansi/1290.dup,v 1.1 2003/12/28 18:21:57 radek Exp $*

Jméno

DUP — „dupe“ duplikuj tos

CORE

Přehled

(x ⟶ x x)

Zduplikuje hodnotu na vrcholu zásobníku.

Popis

Definováno v: dpANS Forth 6.1.1290 CORE (http://forth.sourceforge.net/standard/dpans/dpans6.htm#6.1.1290)

Ukončí práci v prostředí Forthu, a provede návrat do systému odkud jsme Forth spustili.

Description

duplicate the cell on top of the stack - so the two topmost cells have the same value (they are equal w.r.t =) , see DROP for the inverse

ELSE

* *$Header: /home/radek/cvs/forth-book/dictionary/ansi/1310.else,v 1.1 2003/12/28 18:21:57 radek Exp $*

Jméno

ELSE — else část větvení IF ... THEN

CORE

Přehled

Překlad: **(orig1 ⟶ orig2)**

Běh: **(⟶)**

Uvádí else část větvení jenž se vykoná při logické hodnotě `false`.

Definováno v: dpANS Forth 6.1.1310 CORE (http://forth.sourceforge.net/standard/dpans/dpans6.htm#6.1.1310)

Definováno v: dpANS Forth A.6.1.1310 CORE (http://forth.sourceforge.net/standard/dpans/dpans.htm#A.6.1.1310)

Viz. IF, THEN

Popis

Put the location of a new unresolved forward reference orig2 onto the control flow stack. Append the run-time semantics given below to the current definition. The semantics will be incomplete until orig2 is resolved (e.g., by THEN). Resolve the forward reference orig1 using the location following the appended run-time semantics.

Run-time: (--)

Continue execution at the location given by the resolution of orig2.

EXECUTE

* *$Header: /home/radek/cvs/forth-book/dictionary/ansi/1370.execute,v 1.1 2003/12/28 18:21:57 radek Exp $*

Jméno

EXECUTE — Vykoná slovo jehož CFA najde na TOS

Přehled

(CFA ⟶ ???)

(i*x xt ⟶ j*x)

Definováno v: dpANS Forth 6.1.1370 CORE (http://forth.sourceforge.net/standard/dpans/dpans6.htm#6.1.1370)

Popis

Vykoná slovo, jehož adresa je na zásobníku.

Description

Remove xt from the stack and perform the semantics identified by it. Other stack effects are due to the word EXECUTEd.

Run the execution-token on stack - this will usually trap if it was null for some reason, see >EXECUTE simulate:

```
: EXECUTE >R exit ;
```

See: ', [']

Příklad použití

```
*FIXME:
```

Strojový kód slova v Quartus Forthu

Příklad 1. Kód slova EXECUTE [80]

```
                                ; code of word: execute
018C= 818E: 8112 83C8                       DW      $8112 $83C8     ; link to BYE
018E= 8190: 0745 5845 4355 5445             DB      7, "EXECUTE"
0196= 8198: 3207                EXECUTE:    MOVE    TOS, D1           ; odložíme si TOS
0198= 819A: 3E1C                            MOVE    (SP)+, TOS        ;= DROP
019A= 819C: 4EF2 1000                       JMP     0(CS,D1.W)        ; skok na odložený TOS ❶
019E= 81A0: 4E75                            RTS                       ;= EXIT
```

❶ Slovo voláme přes JMP, při RTS daného slova tedy nedojde k návratu do EXECUTE ale o jednu úroveň více. Tedy do slova jenž volalo EXECUTE.

FALSE

* *$Header: /home/radek/cvs/forth-book/dictionary/ansi/1485.false,v 1.1 2003/12/28 18:21:57 radek Exp $*

Jméno

FALSE — umístní do zásobníku příznak/logickou hodnotu `false`

CORE EXT

Přehled

```
( ⟶ false ) ;
```

Definováno v: dpANS Forth 6.2.1485 CORE EXT (http://forth.sourceforge.net/standard/dpans/dpans6.htm#6.2.1485)

Popis

Umístní na vrchol zásobníku logickou hodnotu nepravda, `false`

Příklad použití

* *FIXME:*

```
*FIXME:
```

HERE

* *$Header: /home/radek/cvs/forth-book/dictionary/ansi/1650.here,v 1.1 2003/12/28 18:21:57 radek Exp $*

Jméno

`HERE` — proměnná obsahující adresu první volné buňky v datovém segmentu

Přehled

```
HERE   ( zásobníkový efekt ⟶ )
```

Popis

Definováno v: dpANS Forth 6.1.1650 CORE (http://forth.sourceforge.net/standard/dpans/dpans6.htm#6.1.1650)

Příklad použití

```
*FIXME:
```

Kód slova v Qurtus Forthu

Příklad 1. Kód slova HERE *:[90]

```
; : HERE ( ⟶ 420 ) 420 ;
```

```
01D6= 81D8: 81CA 8D82                       DW      $81CA $8D82     ; link to >IN
01D8= 81DA: 8448 4552 4500                  DB      $80+4, "HERE", 0
01DE= 81E0: 3907                HERE:       MOVE.W  TOS, -(SP)      ;= 420
01E0= 81E2: 3E2D 01A4                       MOVE.W  420(DS), TOS    ;+
01E4= 81E6: 4E75                            RTS                     ;= EXIT
```

I

* *$Header: /home/radek/cvs/forth-book/dictionary/ansi/1680.i,v 1.1 2003/12/28 18:21:57 radek Exp $*

Jméno

`I` — FIXME: jednořádkový popis

Přehled

Běh: **(⟶ n|u) (R: loop-sys ⟶ loop-sys)**

Definováno v: dpANS Forth 6.1.1680 CORE (http://forth.sourceforge.net/standard/dpans/dpans6.htm#6.1.1680)

Popis

*FIXME:

Description

n|u is a copy of the current (innermost) loop index. An ambiguous condition exists if the loop control parameters are unavailable.

Příklad použití

```
# $Id: dict-plus-store.ses,v 1.2 2003/12/28 18:21:58 radek Exp $
Gforth 0.6.2, Copyright (C) 1995-2003 Free Software Foundation, Inc.
Gforth comes with ABSOLUTELY NO WARRANTY; for details type ‘license’
VARIABLE v  ok
8 v !  ok
4 v +!  ok
v @ . 12  ok
BYE
```

Kód slova v Qurtus Forthu

Příklad 1. *:[90]

IF

* *$Header: /home/radek/cvs/forth-book/dictionary/ansi/1700.if,v 1.1 2003/12/28 18:21:57 radek Exp $*

Jméno

`IF` — větvení programu podle podmínky

CORE

Přehled

(cond ⟶)

Definováno v: dpANS Forth 6.1.1700 CORE (http://forth.sourceforge.net/standard/dpans/dpans6.htm#6.1.1700)

Viz.: ELSE, THEN

Description

Put the location of a new unresolved forward reference orig onto the control flow stack. Append the run-time semantics given below to the current definition. The semantics are incomplete until orig is resolved, e.g., by THEN or ELSE.

```
Run-time: ( x -- )
```

If all bits of x are zero, continue execution at the location specified by the resolution of orig.

Popis

Slovo IF je součástí konstrukce {xref linkend="if...then"/} a {xref linkend="if...else...then"/}.

Kód slova v Quartus Forth

```
BC4A:    JSR     $-4442(A2)
BC4E:     JSR      $-43C6(A2)
BC52:     RTS           ;= EXIT
```

LEAVE

* *$Header: /home/radek/cvs/forth-book/dictionary/ansi/1760.leave,v 1.1 2003/12/28 18:21:57 radek Exp $*

Jméno

LEAVE — FIXME: jednořádkový popis

Přehled

(⟶)

Definováno v: dpANS Forth 6.1.1760 CORE (http://forth.sourceforge.net/standard/dpans/dpans6.htm#6.1.1760)

Popis

*FIXME:

Příklad použití

```
# $Id: dict-plus-store.ses,v 1.2 2003/12/28 18:21:58 radek Exp $
Gforth 0.6.2, Copyright (C) 1995-2003 Free Software Foundation, Inc.
Gforth comes with ABSOLUTELY NO WARRANTY; for details type ‘license’
VARIABLE v  ok
8 v !  ok
4 v +!  ok
v @ . 12  ok
BYE
```

Kód slova v Qurtus Forthu

Příklad 1. *:[90]

LOOP

* *$Header: /home/radek/cvs/forth-book/dictionary/ansi/1800.loop,v 1.1 2003/12/28 18:21:57 radek Exp $*

Jméno

LOOP — FIXME: jednořádkový popis

CORE

Přehled

(⟶) (R: loop-sys1 ⟶ loop-sys2)

Definováno v: dpANS Forth 6.1.1800 CORE (http://forth.sourceforge.net/standard/dpans/dpans6.htm#6.1.1800)

Popis

Append the run-time semantics given below to the current definition. Resolve the destination of all unresolved occurrences of LEAVE between the location given by do-sys and the next location for a transfer of control, to execute the words following the LOOP. Run-time: (--) (R: loop-sys1 -- | loop-sys2) An ambiguous condition exists if the loop control parameters are unavailable. Add one to the loop index. If the loop index is then equal to the loop limit, discard the loop parameters and continue execution immediately following the loop. Otherwise continue execution at the beginning of the loop.

See: DO, I.

Příklad použití

```
# $Id: dict-plus-store.ses,v 1.2 2003/12/28 18:21:58 radek Exp $
Gforth 0.6.2, Copyright (C) 1995-2003 Free Software Foundation, Inc.
Gforth comes with ABSOLUTELY NO WARRANTY; for details type ‘license’
VARIABLE v  ok
8 v !  ok
4 v +!  ok
v @ . 12  ok
BYE
```

Kód slova v Qurtus Forthu

Příklad 1. *:[90]

M*

* *$Header: /home/radek/cvs/forth-book/dictionary/ansi/1810.m-star,v 1.1 2004/03/06 00:33:20 radek Exp $*

Jméno

`M*` — „m-star“ **FIXME:**

CORE

Přehled

(n_1 n_2 ⟶ d)

d je výsledek násobení n_1 krát n_2 se znaménkem.

Definováno v: dpANS Forth 6.1.1810 CORE (http://forth.sourceforge.net/standard/dpans/dpans6.htm#6.1.1810)

Definováno v: dpANS Forth A.6.1.1810 CORE (http://forth.sourceforge.net/standard/dpans/dpans.htm#A.6.1.1810)

Popis

FIXME:

Definice

FIXME:

```
: ;
```

Quartus Forth

```
hex see m*
85C0 MOVE.W D7,D0
85C2 MOVE.W (A4)+,D7  ;= DROP
85C4 MOVE.W D7,D1
85C6 MOVE.W (A4)+,D7  ;= DROP
85C8 MULS.W D1,D0
85CA MOVE.W D7,-(A4)  ;= DUP
85CC MOVE.L D0,D7
85CE MOVE.W D7,-(A4)  ;= DUP
FIXME:
```

M+

* *$Header: /home/radek/cvs/forth-book/dictionary/ansi/1830.m-plus,v 1.1 2004/03/06 00:33:20 radek Exp $*

Jméno

M+ — „m-plus“ **FIXME:**popis

DOUBLE

Přehled

($d_1|ud_1$ n ⟶ $d_2|ud_2$)

Přičte *n* k $d_1|ud_1$.

Definováno v: dpANS Forth 8.6.1.1830 CORE (http://forth.sourceforge.net/standard/dpans/dpans8.htm#8.6.1.1830)

Definováno v: dpANS Forth A.8.6.1.1830 CORE (http://forth.sourceforge.net/standard/dpans/dpans.htm#A.8.6.1.1830)

Popis

FIXME:

Definice

FIXME:

```
: ;
```

Forth definice

Slovo je definováno v souboru `Double` takto:

Příklad 1. m+

```
: m+ S>D D+ ;
```

Quartus Forth

```
hex see m+
;23456789012345678901234567890123456789
CBD6 EXT.L D7
CBD8 MOVE.W D7,-(A4)    ;= DUP
CBDA SWAP D7
CBDC JMP $-6152(A2)     ;= D+
```

NIP

* *$Header: /home/radek/cvs/forth-book/dictionary/ansi/1930.nip,v 1.1 2003/12/28 18:21:57 radek Exp $*

Jméno

NIP — Odstraní položku pod vrcholem zásobníku

Přehled

(x1 x2 ⟶ x2)

Definováno v: dpANS Forth 6.2.1930 CORE,EXT (http://forth.sourceforge.net/standard/dpans/dpans6.htm#6.2.1930)

Popis

FIXME:

Description

Drop the first item below the top of stack.

drop the value under the top of stack, inverse of TUCK simulate:

```
: NIP SWAP DROP ;
```

OVER

* *$Header: /home/radek/cvs/forth-book/dictionary/ansi/1990.over,v 1.1 2003/12/28 18:21:57 radek Exp $*

Jméno

OVER — Place a copy of x1 on top of stack

Přehled

(x1 x2 ⟶ x1 x2 x1)

Definováno v: dpANS Forth 6.1.1990 CORE (http://forth.sourceforge.net/standard/dpans/dpans6.htm#6.1.1990)

Popis

Ukončí práci v prostředí Forthu, a provede návrat do systému odkud jsme Forth spustili.

Description

get the value from under the top of stack. The inverse operation would be TUCK

PARSE

* *$Header: /home/radek/cvs/forth-book/dictionary/ansi/2008.parse,v 1.1 2003/12/28 18:21:57 radek Exp $*

Jméno

PARSE — čte/parsuje řetězec ve vstupním bufferu (TIB)

CORE EXT

Přehled

(char"ccc<char>" ⟶ c-addr u) ;

Definováno v: dpANS Forth 6.2.2008 CORE EXT (http://forth.sourceforge.net/standard/dpans/dpans6.htm#6.2.2008)

Čte řetězec ve vstupním bufferu ukončený znakem *char*.

c-addr je adresa (ve vstupním bufferu) a *u* je délka čteného řetězce.

Popis

*FIXME:Parse ccc delimited by the delimiter char. c-addr is the address (within the input buffer) and u is the length of the parsed string. If the parse area was empty, the resulting string has a zero length.

Příklad použití

```
# $Id: dict-plus-store.ses,v 1.2 2003/12/28 18:21:58 radek Exp $
Gforth 0.6.2, Copyright (C) 1995-2003 Free Software Foundation, Inc.
Gforth comes with ABSOLUTELY NO WARRANTY; for details type ‘license’
VARIABLE v  ok
8 v !  ok
4 v +!  ok
v @ . 12  ok
BYE
```

Kód slova v Qurtus Forthu

Příklad 1. *:[90]

PICK

* *$Header: /home/radek/cvs/forth-book/dictionary/ansi/2030.pick,v 1.1 2003/12/31 00:08:29 radek Exp $*

Jméno

`PICK` — duplikování prvku z hlouby zásobník na vrchol

CORE EXT

Přehled

(x_u ... x_1 x_0 u ⟶ x_u ... x_1 x_0 x_u)

Odstraní *u* a zkopíruje u-tý prvek na vrchol zásobníku..

Definováno v: dpANS Forth 6.2.2030 CORE EXT (http://forth.sourceforge.net/standard/dpans/dpans6.htm#6.2.2030)

Definováno v: dpANS Forth A.6.2.2030 CORE EXT (http://forth.sourceforge.net/standard/dpans/dpans.htm#A.6.2.2030)

Popis

FIXME:

POSTPONE

* *$Header: /home/radek/cvs/forth-book/dictionary/ansi/2033.postpone,v 1.1 2003/12/28 18:21:57 radek Exp $*

Jméno

`POSTPONE` — FIXME: jednořádkový popis

Přehled

("<spaces>*name*" ⟶)

Definováno v: dpANS Forth 6.1.2033 CORE (http://forth.sourceforge.net/standard/dpans/dpans6.htm#6.1.2033)

Přeskočí vedoucí mezery a načte/parsuje jméno ukončené mezerou. Najde jméno ve slovníku a přípojí *compilation semantics* jména k aktuální definici.

Popis

*FIXME:

Příklad použití

```
# $Id: dict-plus-store.ses,v 1.2 2003/12/28 18:21:58 radek Exp $
Gforth 0.6.2, Copyright (C) 1995-2003 Free Software Foundation, Inc.
Gforth comes with ABSOLUTELY NO WARRANTY; for details type ‘license’
VARIABLE v  ok
8 v !  ok
4 v +!  ok
v @ . 12  ok
BYE
```

Kód slova v Qurtus Forthu

Příklad 1. *:[90]

ROLL

* *$Header: /home/radek/cvs/forth-book/dictionary/ansi/2150.roll,v 1.1 2003/12/31 00:08:29 radek Exp $*

Jméno

`ROLL` — rolování zásobníku, vyjmutí prvku z hlouby zásobníku a uložení na vrchol

CORE EXT

Přehled

(x_u x_{u-1} ... x_1 x_0 u ⟶ x_u ... x_1 x_0 x_u)

Odstraní *u* a rotuje *u* prvků zásobníku.

Definováno v: dpANS Forth 6.2.2150 CORE EXT (http://forth.sourceforge.net/standard/dpans/dpans6.htm#6.2.2150)

Definováno v: dpANS Forth A.6.2.2150 CORE EXT (http://forth.sourceforge.net/standard/dpans/dpans.htm#A.6.2.2150)

Popis

FIXME:

ROT

* *$Header: /home/radek/cvs/forth-book/dictionary/ansi/2160.rot,v 1.1 2003/12/28 18:21:57 radek Exp $*

Jméno

ROT — *FIXME:

Přehled

(x1 x2 x3 ⟶ x2 x3 x1)

Definováno v: dpANS Forth 6.1.2160 CORE (http://forth.sourceforge.net/standard/dpans/dpans6.htm#6.1.2160)

Popis

FIXME:

Příklad použití

FIXME:

Kód slova v Qurtus Forthu

Příklad 1. Kód slova ROT *:[80]

```
02FA= 82FC: 82E0                            DW      $82E0           ; link to 2SWAP
02FC= 82FE: 0352 4754                       DB      3, "ROT"
0300= 8302: 3007                ROT:        MOVE.W  TOS, D0         ;=
0302= 8304: 2E14                            MOVE.L  (SP), TOS       ;
0304= 8306: 2207                            MOVE.L  TOS, D1
0306= 8308: 3200                            MOVE.W  D0, D1
0308= 830A: 4841                            SWAP    D1
030A= 830C: 2881                            MOVE.L  D1, (SP)        ;
030C= 830E: 4E75                            RTS                     ;= EXIT
```

S>D

* *$Header: /home/radek/cvs/forth-book/dictionary/ansi/2170.stod,v 1.1 2003/12/28 18:21:57 radek Exp $*

Jméno

`S>D` — rozšíření čísla se znaménkem na dlouhé číslo

CORE

Přehled

(n ⟶ d)

Definováno v: dpANS Forth 6.1.2170 CORE (http://forth.sourceforge.net/standard/dpans/dpans6.htm#6.1.2170)

Popis

FIXME:

Kód slova

Příklad 1. Kód slova S>D

```
851C:           EXT.L   TOS                 ; 34074 ext.l d7
851E:           MOVE.W  TOS, -(SP)          ; 34076 move.w d7, -(a4)  = DUP
8520:           SWAP    TOS                 ; 34078 swap d7
8522:           RTS                                         ; 34080 rts = EXIT
```

SOURCE

* *$Header: /home/radek/cvs/forth-book/dictionary/ansi/2216.source,v 1.1 2003/12/28 18:21:57 radek Exp $*

Jméno

`SOURCE` — adresa a velikost vstupního buferu

CORE

Přehled

(⟶ c-addr u)

`c-addr` je adresa vstupního bufferu a *u* počet znaků v něm.

Definováno v: dpANS Forth 6.1.2216 CORE (http://forth.sourceforge.net/standard/dpans/dpans6.htm#6.1.2216)

Popis

FIXME:

Description

The current point of interpret can be gotten through SOURCE. The buffer may flag out TIB or BLK or a FILE and IN gives you the offset therein. Traditionally, if the current SOURCE buffer is used up, REFILL is called that asks for another input-line or input-block. This scheme would have made it impossible to stretch an [IF] ... [THEN] over different blocks, unless [IF] does call REFILL

SOURCE simplifies the process of directly accessing the input buffer by hiding the differences between its location for different input sources. This also gives implementors more flexibility in their implementation of buffering mechanisms for different input sources. The committee moved away from an input buffer specification consisting of a collection of individual variables, declaring TIB and #TIB obsolescent.

SOURCE in this form exists in F83, POLYFORTH, LMI's Forths and others. In conventional systems it is equivalent to the phrase

```
BLK @  IF BLK @ BLOCK 1024  ELSE TIB #TIB @ THEN
```

Příklad použití

```
*FIXME:
```

Strojový kod slova v Qurtus Forthu

Příklad 1. Kód slova SOURCE *:[90]

```
                              ; variable: STATE
01B0= 81B2: 81A2 9D18                        DW      $81a2 ⟶ $9D18    ; link to STATE
01B2= 81B4: 0653 4F55 5243 4500              DB      6, "SOURCE"
01BA= 81BC: 3907              SOURCE:        MOVE    TOS, -(SP)       ;= 94 @
01BC= 81BE: 3E2D 005E                        MOVE    94(DS), TOS      ;+
01C0= 81C2: 3907                             MOVE    TOS, -(SP)       ;= 96 @
01C2= 81C4: 3E29 0060                        MOVE    96(DS), TOS      ;+
01C6= 81C8: 4E75                             RTS                      ;= EXIT
```

SOURCE-ID

* *$Header: /home/radek/cvs/forth-book/dictionary/ansi/2218.source-id,v 1.1 2003/12/28 18:21:57 radek Exp $*

Jméno

`SOURCE-ID` — identifikuje/popisuje vstupní zdroj

Přehled

`slovo ( ⟶ 0 | -1 | fileid )`

- `0` — uživatelské vstupní zařízení
- `-1` — řetězec (přes evaluate)
- `fileid` — testový soubor

`fileid`

Popis

Definováno v: dpANS Forth 6.2.2218 CORE (http://forth.sourceforge.net/standard/dpans/dpans6.htm#6.2.2218)

Definováno v: dpANS Forth 11.6.1.2218 CORE (http://forth.sourceforge.net/standard/dpans/dpans11.htm#11.6.1.2218)

Příklad použití

`FIXME:`

Kód slova v Qurtus Forthu

Příklad 1. Kód slova SOURCE-ID *:[90]

```
;
0246= 8248: 8234                            DW     $8234             ; link to currenty
0248= 824A: 8953 4F55 5243 452D             DB     $80+8, "SOURCE-ID"
0250=       4944
0252= 8254: 3907                SOURCE-ID:  MOVE.W  TOS, -(SP)        ;= 196 @
0254= 8256: 3E2D 00C4                       MOVE.W  196(DS), TOS      ;+
0256= 8258: 4E75                            RTS                       ;= EXIT
```

STATE

* *$Header: /home/radek/cvs/forth-book/dictionary/ansi/2250.state,v 1.1 2003/12/28 18:21:57 radek Exp $*

Jméno

STATE — proměnná obsahující informaci o stavu EXECUTE/COMPILE

CORE

Přehled

STATE (⟶ addr)

Definováno v: dpANS Forth 6.1.2250 CORE (http://forth.sourceforge.net/standard/dpans/dpans6.htm#6.1.2250)

Popis

Proměnná obsahující informaci o režimu/stavu v němž se ystém nacháyí. Hodnota 0 obvykle značí stav EXECUTE a jiná, nenulová hodnota značí COMPILE.

a-addr is the address of a cell containing the compilation-state flag. STATE is true when in compilation state, false otherwise. The true value in STATE is non-zero, but is otherwise implementation-defined. Only the following standard words alter the value in STATE: : (colon), ; (semicolon), ABORT, QUIT, :NONAME, [(left-bracket), and] (right-bracket). Note: A program shall not directly alter the contents of STATE.

Strojový kód slova v Qurtus Forthu

Příklad 1. Kód slova STATE [80]

```
                                ; Uloží do TOS adresu proměnné STATE
01A0= 81A2: 818E 83C8                       DW      $818E $83C8      ; link to EXECUTE
01A2= 8A04: 8553 5441 5445                  DB      $80+5, "STATE"
01A8= 81AA: 3907                STATE:      MOVE.W  TOS, -(SP)        ;= 374
01AA= 81AC: 3E3C 0176                       MOVE    #374, TOS         ;+
01AE= 81B0: 4E75                            RTS                       ;= EXIT
```

Příklad použití

```
*FIXME:
```

SWAP

* *$Header: /home/radek/cvs/forth-book/dictionary/ansi/2260.swap,v 1.1 2003/12/28 18:21:57 radek Exp $*

Jméno

SWAP — prohodí mezi sebou dvě buňky na vrcholu zásobníku, TOS a NOS

Přehled

SWAP (n1 n2 ⟶ n2 n1)

Definováno v: dpANS Forth 6.1.2260 CORE (http://forth.sourceforge.net/standard/dpans/dpans6.htm#6.1.2260)

Popis

FIXME:

Příklad použití

.

Kód slova v Qurtus Forthu

Příklad 1. Kód slova SWAP *:[80]

```
02CE= 82D0: 82B8                            DW      $82B8             ; link to DEPTH
02D0= 82D2: 8453 5741 5000                  DB      $80+4, "SWAP"
02D6= 82D8: 3007                SWAP:       MOVE.W  TOS, D0           ;=
02D8= 82DA: 3E14                            MOVE.W  (SP), TOS         ;
02DA= 82DC: 3880                            MOVE.W  D0, (SP)
02DC= 82DE: 4E75                            RTS                       ;= EXIT
```

THEN

* *$Header: /home/radek/cvs/forth-book/dictionary/ansi/2270.then,v 1.1 2003/12/28 18:21:57 radek Exp $*

Jméno

THEN — ukončení podmíněného větvení IF...THEN nebo IF...ELSE...THEN

CORE

Přehled

Překlad: **(orig ⟶)**

Běh: **(⟶)**

Definováno v: dpANS Forth 6.1.2270 CORE (http://forth.sourceforge.net/standard/dpans/dpans6.htm#6.1.2270)

Popis

Append the run-time semantics given below to the current definition. Resolve the forward reference orig using the location of the appended run-time semantics.

```
Run-time: ( -- )
```

Continue execution.

Viz: IF, ELSE

TIB

* *$Header: /home/radek/cvs/forth-book/dictionary/ansi/2290.t-i-b,v 1.1 2003/12/28 18:21:57 radek Exp $*

Jméno

`TIB` — popis

CORE

EXT

Přehled

`bye` **`( ⟶ )`**

Protože slovo ukončuje práci prostředí forthu, nedojde nikdy k návratu z něj.

Popis

Definováno v: dpANS Forth 15.6.2.0830 EXT (http://forth.sourceforge.net/standard/dpans/dpans15.htm#15.6.2.0830)

Ukončí práci v prostředí Forthu, a provede návrat do systému odkud jsme Forth spustili.

TUCK

* *$Header: /home/radek/cvs/forth-book/dictionary/ansi/2300.tuck,v 1.1 2003/12/28 18:21:57 radek Exp $*

Jméno

`TUCK` — **FIXME:**

CORE

EXT

Přehled

(x1 x2 ⟶ x2 x1 x2)

Definováno v: dpANS Forth 6.2.2300 CORE,EXT (http://forth.sourceforge.net/standard/dpans/dpans6.htm#6.2.2300)

Copy the first (top) stack item below the second stack item.

Popis

FIXME:

Description

shove the top-value under the value beneath. See OVER and NIP simulate:

```
: TUCK  SWAP OVER ;
```

VARIABLE

* *$Header: /home/radek/cvs/forth-book/dictionary/ansi/2410.variable,v 1.1 2003/12/28 18:21:57 radek Exp $*

Jméno

`VARIABLE` — vytvoření proměnné

CORE

Přehled

("*jméno*" ⟶)

`Použití: proměnně:` ***jméno*** **(⟶ addr)**

Popis

Definováno v: dpANS Forth 6.1.2410 CORE (http://forth.sourceforge.net/standard/dpans/dpans6.htm#6.1.2410)

Příklad použití

```
*FIXME:
```

[’]

* *$Header: /home/radek/cvs/forth-book/dictionary/ansi/2510.bracket-tick,v 1.1 2003-12-28 18:21:57 radek Exp $*

Jméno

`[’]` — popis

Přehled

`Překlad:` **`( "<spaces>name" ⟶ )`**

`Běh:` **`( ⟶ xt )`**

Definováno v: dpANS Forth 6.1.2510 CORE (http://forth.sourceforge.net/standard/dpans/dpans6.htm#6.1.2510)

Definováno v: dpANS Forth A.6.1.2510 CORE (http://forth.sourceforge.net/standard/dpans/dpans.htm#A.6.1.2510)

Protože slovo ukončuje práci prostředí forthu, nedojde nikdy k návratu z něj.

Popis

Ukončí práci v prostředí Forthu, a provede návrat do systému odkud jsme Forth spustili.

Description

Skip leading space delimiters. Parse name delimited by a space. Find name. Append the run-time semantics given below to the current definition. An ambiguous condition exists if name is not found. Place name’s execution token xt on the stack. The execution token returned by the compiled phrase [’] X is the same value returned by ’ X outside of compilation state.

See: 3.4.1 Parsing, 6.1.0070 ’ , A.6.1.2033 POSTPONE , A.6.1.2510 [’] , D.6.7 Immediacy.

III. Slovník FAKE

Slova definovaná v ANSI

Slouží k definování cílů odkazů pro které ještě neexistuje záznam ve slovníku.

evaluate

* *Fake entry evaluate*

Jméno

`evaluate` — 7.6.1.1360

Fake

Nothing

exit

* *Fake entry exit*

Jméno

`exit` —

Fake

Nothing

IV. Slovník FIG forthu

Slova definovaná v ANSI

Některá slova, jenž jsem považoval za vhodná zde uvést.

Odkazy na

- **FIXME:** ()

Výplň

* *$Header: /home/radek/cvs/forth-book/dictionary/ansi/template.xml,v 1.2 2003/12/31 00:08:29 radek Exp $*

Jméno

`Výplň` — „sound“ popis

CORE

Přehled

(⟶)

Protože slovo ukončuje práci prostředí forthu, nedojde nikdy k návratu z něj.

Definováno v: dpANS Forth 6.1.0830 CORE (http://forth.sourceforge.net/standard/dpans/dpans6.htm#6.1.0830)

Popis

Toto slovo je jen výplní aby mi nehavaroval překlad dokumentu na prázdný slovník.

FIXME:

Definice

FIXME:

```
: ;
```

V. Slovník ANSI forthu

Slova definovaná v ANSI

Některá slova, jenž jsem považoval za vhodná zde uvést.

BlankFormId

* *$Header: /home/radek/cvs/forth-book/dictionary/qf/dict/BlankFormID,v 1.1 2003/12/28 18:21:58 radek Exp $*

Jméno

`BlankFormId` — jednořádkový popis

Přehled

`slovo ( zásobníkový efekt ⟶ )`

Popis

*FIXME:

Příklad použití

```
*FIXME:
```

Kód slova v Qurtus Forthu

Příklad 1. Kód slova BlankFormID *:[80]

Uloží na vrchol zásobníku ID prázdného formuláře, což je v Quartus Forthu 1001.

```
; BlankFormID ( ⟶ 1001) 1001 ;
0272= 8274: 825C                            DW      $825C            ; link to eventhandler
0274= 8276: 8B42 6261 6E6B 466F
            726D 4944
0280= 8282: 3907                BlankFormID: MOVE.W  TOS, -(SP)       ;= 1001
0282= 8284: 3E3C 03E9                       MOVE.W  #1001, TOS        ;+
0286= 8288: 4E75                            RTS                       ;= EXIT
```

MainFormId

* *$Header: /home/radek/cvs/forth-book/dictionary/qf/dict/MainFormID,v 1.1 2003/12/28 18:21:58 radek Exp $*

Jméno

`MainFormId` — jednořádkový popis

Přehled

slovo (zásobníkový efekt ⟶)

Popis

FIXME:

Příklad použití

FIXME:

Kód slova v Qurtus Forthu

Příklad 1. Kód slova MainFormID *:[80]

Uloží na vrchol zásobníku ID formuláře MainForm.

```
02A0= 82A2: 828A                            DW      $828A             ; link to TitledFormID
02A2= 82A4: 8A4D 6169 6E46 6F72             DB      $40+10, "MainFormID"
            6D49 4400
02AE= 82B0: 3907                MainFormID: MOVE.W  TOS, -(SP)        ;= 1000
02B0= 82B2: 3E3C 03E8                       MOVE.W  #1000, TOS        ;+
02B4= 82B6: 4E75                            RTS                       ;= EXIT
```

TitledFormId

* *$Header: /home/radek/cvs/forth-book/dictionary/qf/dict/TitledFormID,v 1.1 2003/12/28 18:21:58 radek Exp $*

Jméno

`TitledFormId` — jednořádkový popis

Přehled

slovo (zásobníkový efekt ⟶)

Popis

*FIXME:

Příklad použití

```
*FIXME:
```

Kód slova v Qurtus Forthu

Příklad 1. Kód slova TitledFormID *:[80]

Uloží na vrchol zásobníku ID formuláře TitledForm.

```
0288= 828A: 8274                            DW      $8274            ; link to BlankFormID
028A= 828C: 8C54 6974 6C65 6446             DB      $80+12, "TitledFormID"
            6F72 6D49 4400
0298= 829A: 3907                TitledFormID:MOVE.W  TOS, -(SP)       ;= 1002
029A= 829C: 3E3C 03EA                       MOVE.W  #1002, TOS        ;+
029E= 82A0: 4E75                            RTS                       ;= EXIT
```

C@A

* *$Header: /home/radek/cvs/forth-book/dictionary/qf/dict/c-fetch-a,v 1.1 2003/12/28 18:21:58 radek Exp $*

Jméno

C@A — přečte znak z absolutní 32bitové adresy

Přehled

(addr. ⟶ char)

Popis

Přečte znak z absolutní adresy.

Kód slova

Příklad 1. Kód slova cs@

```
36882 swap d7
36884 move.w (a4), d7
```

```
36886 move.l d7, a0
36888 move.l (a4)+, d7 = 2DROP
36890 clr.w d0
36892 move.b (a0), d0
36894 move.w d7, -(a4) = DUP
36896 move.w d0, d7
36898 rts = EXIT
```

currentx

* *$Header: /home/radek/cvs/forth-book/dictionary/qf/dict/currentx,v 1.1 2003/12/28 18:21:58 radek Exp $*

Jméno

`currentx` — jednořádkový popis

Přehled

`slovo ( zásobníkový efekt ⟶ )`

Popis

*FIXME:

Příklad použití

```
*FIXME:
```

Kód slova v Qurtus Forthu

Příklad 1. Kód slova currentx *:[90]

```
;
021E= 8220: 8210                            DW     $8210             ; link to BASE
0220= 8222: 8863 7572 7265 6E74             DB     $80+8, "currentx", 0
0228=       7800
022A= 822C: 3907                currentx:   MOVE.W  TOS, -(SP)        ;= 86
022C= 822E: 3E3C 0056                       MOVE.W  #86, TOS          ;+
0230= 8232: 4E75                            RTS                       ;= EXIT
```

currenty

* *$Header: /home/radek/cvs/forth-book/dictionary/qf/dict/currenty,v 1.1 2003/12/28 18:21:58 radek Exp $*

Jméno

`currenty` — jednořádkový popis

Přehled

`slovo ( zásobníkový efekt ⟶ )`

Popis

*FIXME:

Příklad použití

```
*FIXME:
```

Kód slova v Qurtus Forthu

Příklad 1. *:[90]

Příklad 2. Kód slova currenty *:[90]

```
;
0232= 8234: 8220                            DW     $8220             ; link to currentx
0234= 8236: 8863 7572 7265 6E74             DB     $80+8, "currenty", 0
023C=       7900
023E= 8240: 3907                currenty:   MOVE.W  TOS, -(SP)        ;= 88
0240= 8242: 3E3C 0058                       MOVE.W  #88, TOS          ;+
0244= 8246: 4E75                            RTS                       ;= EXIT
```

event

* *$Header: /home/radek/cvs/forth-book/dictionary/qf/dict/event,v 1.1 2003/12/28 18:21:58 radek Exp $*

Jméno

`event` — adresa „event struktury“ obsahující poslední událost obdrženou od PalmOSu

Přehled

event (⟶ addr)

Popis

Podrobný popis co slovo dělá.

Příklad použití

Ukázkový příklad, nebo několik příkladů použití slova.

```
256 true HwrBacklight
```

Strojový kód slova v Qurtus Forthu

Příklad 1. Kód slova event *:[90]

```
...
;
01FE= 8200: 818E                              DW      $818E             ; link to window-bounds
0200= 8202: 8565 7665 6E74                    DB      $80+5, "event"
0206= 8208: 3907                event:        MOVE.W  TOS, -(SP)        ;= 280
0208= 820A: 3E3C 0118                         MOVE.W  #280, TOS         ;+
020C= 820E: 4E75                              RTS                       ;= EXIT
```

eventhandler

* *$Header: /home/radek/cvs/forth-book/dictionary/qf/dict/eventhandler,v 1.1 2003/12/28 18:21:58 radek Exp $*

Jméno

`eventhandler` — jednořádkový popis

Přehled

`slovo   ( zásobníkový efekt ⟶ )`

Popis

*FIXME:

Příklad použití

```
*FIXME:
```

Kód slova v Qurtus Forthu

Příklad 1. Kód slova eventhandler *:[80]

```
;
025A= 825C: 8248                            DW     $8248             ; link to SOURCE-ID
025C= 825E: 8C65 7665 6E74 6861             DB     $80+12, "eventhandler"
0260=       6E64 6C65 7200
026A= 826C: 3907                eventhandler:MOVE.W  TOS, -(SP)        ;= 190
026C= 826E: 3E3C 00BE                       MOVE.W  #190, TOS          ;+
0270= 8272: 4E75                            RTS                        ;= EXIT
```

needs

* *$Header: /home/radek/cvs/forth-book/dictionary/qf/dict/needs,v 1.1 2003/12/28 18:21:58 radek Exp $*

Jméno

needs — Načtení zdrojového kódu z jiného souboru/memo. Obdoba include.

Přehled

needs *memo-name* (⟶)

Popis

Příkaz načte a vykoná zdrojový program z uvedeného souboru

Příklad použití

Ukázkový příklad, nebo několik příkladů použití slova.

Strojový kód slova v Qurtus Forthu

* *TBD*

Příklad 1. Kód slova needs *:[90]

noop

* *$Header: /home/radek/cvs/forth-book/dictionary/qf/dict/noop,v 1.1 2003/12/28 18:21:58 radek Exp $*

Jméno

noop — jednořádkový popis

Přehled

noop (⟶)

Popis

Procedura nedělá níc.

Příklad použití

FIXME:

Strojový kód slova/procedury v Qurtus Forthu

Příklad 1. noop

```
                                                        ;*FIXME:
00DE= 80E0: 8000 8200                       DW      $8000 $8200     ; ??? end of dictionary
00E0=       446E 6F6F 7000                  DB      $44, "noop", 0
00E6= 80E8: 4E75                noop:       RTS                                             ;=
```

(bye)

* *$Header: /home/radek/cvs/forth-book/dictionary/qf/dict/paren-bye-paren,v 1.1 2003/12/28 18:21:58 radek Exp $*

Jméno

`(bye)` — jednořádkový popis

Přehled

`(bye)  ( ⟶ )`

Popis

Procedura je volána interně když Quartus Forth obdrží appStopEvent. Ukončí běh Qurtus Forthu.

Příklad použití

```
*FIXME:
```

Strojový kód slova/procedury v Qurtus Forthu

Příklad 1. (bye)

```
;ADDR  WORDS           LABEL        MNEMO   ARGS
;----  --------------  -----------  ------- --------------------
00E8= 80EA: 80E0                    DW      $80E0             ; link to noop
00EA= 80EC: 0528 6279 6529          DB      5, "(bye)"
                                    FplFree
00F0= 80F2: 4E4F A0E0  (bye):         = SYSTRAP FplFree
                                    WinSetUnderlineMode(#0.W)
00F4= 80F6: 3F3C 0000                 = MOVE.W  #0, -(RP)
00F8= 80FA: 4E4F A225                 = SYSTRAP WinSetUnderlineMode
00FC= 80FE: 548F                      = ADDQ.L  #2, RP
00FE= 8100: 4A6D 000C               TST.W   12(DS)
0102= 8104: 6706                  .--BEQ.S   $810C           ; $+6 = 010A
0104= 8106: 206D 001E             |  MOVE.L  30(DS), A0
0108= 810A: 4ED0                  |  JMP     (A0)
010A= 810C: 4EEA 8094             '->JMP     -32620(CS)
010E= 8110: 4E75                    RTS                      ;= EXIT
```

(ekey)

* *$Header: /home/radek/cvs/forth-book/dictionary/qf/dict/paren-ekey-paren,v 1.1 2003/12/28 18:21:58 radek Exp $*

Jméno

`(ekey)` — Vyzvednutí události z fronty událostí

Přehled

(time. ⟶ ekey) [time.]

Popis

Vyzvednutí události z fronty událostí PalmOS. Parametr *time* je dve buňky veliký a jedná se o počet setin sekundy. Tedy „1.“ je 10 ms. Není li ve frontě žádná událost, program čeká nastavený čas na vznik událost. Nenastane-li v dané době žádná událost je „vytvořena“ událost nilEvent

Příklad použití

```
need events
: handle-events ( -- )
    begin
        10. (ekey) dispatch-event drop
    again ;
```

(ID)

* *$Header: /home/radek/cvs/forth-book/dictionary/qf/dict/paren-id-paren,v 1.1 2003/12/28 18:21:58 radek Exp $*

Jméno

`(ID)` — převádí čtyřznakový identifikátor zdroje na číslo (32-bitové)

ids

Přehled

`( cccc ⟶ id. )`

Převede čtyřznakoý identifikátor následující slovo na 32-bitové číslo.

Popis

Slouží pro zadávání jmen (identifikátorů) zdrojů.

Příklad použití

V následujícím příkladu otevření zdrojvé databáze je použito slovo **(ID)** hned dvakrát. První použití převede identifikátor `Kalk` na číslo identifikující tvůrce a druhé použití převede `rsrc` na identifikátor typu databáze. Slovo **use-resources** si pak ze tzásobníku vyzvedne obě 32-bitová čísla a použije je k vyhledání databáze jenž otevře.

Příklad 1. Příklad použití slova (ID)

```
(ID) Kalk (ID) rsrc use-resources
```

>ABS

* *$Header: /home/radek/cvs/forth-book/dictionary/qf/dict/to-abs,v 1.1 2003/12/28 18:21:58 radek Exp $*

Jméno

>ABS — převod 16-ti bitové adresy datového prostoru na 32-ti bitovou absolutní adresu

QF builtin: Memory Words

Přehled

(addr ⟶ addr.)

Kód slova

Příklad 1. Kód slova >ABS

```
                          ; code of word: >abs
8F76: 900C                                DW      $900C
8F78: 043E 6162 7300                      DB      4, ">abs", 0
8F7E: 41F5 7000 .... ....  '>abs':        LEA     $0(DS, TOS.W), A0
8F82: 2E08           ....  ________       MOVE.L  A0, TOS
8F84: 4847                                SWAP    TOS
8F86: 3908                                MOVE.W  A0, -(SP)
8F88: 4E75                                RTS                             ;= EXIT
```

>BYTE

* *$Header: /home/radek/cvs/forth-book/dictionary/qf/dict/to-byte,v 1.1 2003/12/28 18:21:58 radek Exp $*

Jméno

>BYTE — refpurpose

Přehled

(n ⟶ n<<)

Popis

Posune spodní slabiku buňky do horní slabiky. C výraz `n << 8`

Příklad použití

```
1 >byte .
256 ok
```

>digit

* *$Header: /home/radek/cvs/forth-book/dictionary/qf/dict/to-digit,v 1.1 2003/12/28 18:21:58 radek Exp $*

Jméno

`>digit` — Převod znaku na číslo

Přehled

>digit (c ⟶ n)

Popis

FIXME: Vykoná slovo, jehož adresa je na zásobníku.

Příklad použití

FIXME:

Definice slova

```
: >digit ( c -- n )
    DUP 64 > 9 and SWAP 15 and + ;

: >digit ( c -- n )
    DUP 64 > 9 and + 15 and ;

: >CHAR ( n -- c ) \ Converts a number to its ASCII representation
    DUP 9 > [ CHAR A  CHAR 9 - 1 - ] LITERAL  AND + ASCII 0 + ;

: digit?  ( c -- f ) \ test whether a char is a digit 0-9
    [CHAR] 0  [ CHAR 9 1+ ] LITERAL  WITHIN ;

: DIGIT  ( u -- char )
    DUP 9 > 7 AND + 48 + ;

: num>char  ( u -- char )
    DUP 9 >
    [ char A char 0 - 10 - ] literal and +
    [char] 0 + ;
```

window-bounds

* *$Header: /home/radek/cvs/forth-book/dictionary/qf/dict/window-bounds,v 1.1 2003/12/28 18:21:58 radek Exp $*

Jméno

`window-bounds` — jednořádkový popis

Přehled

`slovo   ( zásobníkový efekt ⟶ )`

Popis

*FIXME:

Příklad použití

```
*FIXME:
```

Kód slova v Qurtus Forthu

Příklad 1. Kód slova window-bounds *:[90]

```
                                ; : window-bounds ( ⟶ ) 78 ;
01E6= 81E8: 81D8                        DW      $81D8           ; link to HERE
01E8= 81EA: 8D77 696E 646F 772D         DB      $80+13, "window-bounds"
01F0=       626F 75E6 6473
      81F8:                     window-bounds:
01F6= 81F8: 3907                        MOVE.W  TOS, -(SP)      ;= 78
01F8= 81FA: 3E3C 004E                   MOVE.W  #78, TOS        ;+
01FC= 81FE: 4E75                        RTS                     ;= EXIT
```

VI. Slovník 2

Vybraná slova ze slovníku Forthu

Některá slova, jenž jsem považoval za vhodná zde uvést.

Zápis speciálních znaků v atributu `id`

@

`:at:`

>

`:gt:`

<

`:lt:`

.

`:dot:` - ve jménu souboru, pokud se nachází na prvním místě píši jako _

(

`:lpar:`

)

`:rpar:`

=

`:eq:`

+

`:plus:`

číslice

pokud začíná identifikátor číslicí, umístním před ni znak `:`

CloseDB

* *$Header: /home/radek/cvs/forth-book/dictionary/qf/dict-old/CloseDB,v 1.1 2003-12-28 18:21:58 radek Exp $*

Jméno

`CloseDB` — Uzavření databáze určené ovladačem *`dbr`*.

Přehled

```
: CloseDB ( dbr. ⟶ ) ;
```

Popis

FIXME: doplnit

Příklad použití

```
*FIXME:
```

CreateDB

* *$Header: /home/radek/cvs/forth-book/dictionary/qf/dict-old/CreateDB,v 1.1 2003-12-28 18:21:58 radek Exp $*

Jméno

`CreateDB` — vytvoření nové databáze s daným jménem, typem a tvůrcem.

Přehled

```
: CreateDB ( resDB? type. creator. &zname zlen ⟶ ) ;
```

Popis

Otevře databázi. Je li *`resDB?`* true, otevře resource databázi. Řetězec na *`&zname`* musí být ukončen nolovým znakem.

Příklad použití

```
*FIXME:
```

HwrBacklight

* *$Header: /home/radek/cvs/forth-book/dictionary/qf/dict-old/HwrBacklight,v 1.1 2003-12-28 18:21:58 radek Exp $*

Jméno

`HwrBacklight` — refpurpose

Přehled

```
: HwrBacklight ( 0|256 true --  ) ( 0 false -- n ) ;
```

Popis

Zapnutí/Vypnutí podsvětlení zobrazovače, nebo zjištění stavu podsvětlení zobrzovače.

Příklad použití

```
256 true HwrBacklight
```

OpenDB

* *$Header: /home/radek/cvs/forth-book/dictionary/qf/dict-old/OpenDB,v 1.1 2003-12-28 18:21:58 radek Exp $*

Jméno

`OpenDB` — Otevření existující databáze.

Přehled

```
: OpenDB ( mode zaddr len ⟶ dbr. ) ;
```

Popis

Otevře wxistující databázi v daném módu. Databáze je určena svým jménem *zaddr*. Jméno databáze musí být ukončeno nulovým bajtem.

Příklad použití

```
*FIXME:
```

OpenResDB

* *$Header: /home/radek/cvs/forth-book/dictionary/qf/dict-old/OpenResDB,v 1.1 2003-12-28 18:21:58 radek Exp $*

Jméno

`OpenResDB` — otevření databáze zdrojů („resource database“) podle tvůrce a typu

Přehled

```
: OpenResDB ( creator-id. type. ⟶ DbOpenRef. ) ;
```

Popis

Otevře databázi zdrojů a vrátí handler.

Příklad použití

```
*FIXME:
```

UseCard

* *$Header: /home/radek/cvs/forth-book/dictionary/qf/dict-old/UseCard,v 1.1 2003-12-28 18:21:58 radek Exp $*

Jméno

`UseCard` — Nastavení karty se kterou bude pracovat OpenDB.

Přehled

```
: UseCard ( n ⟶ ) ;
```

Popis

Nastaví kartu pro OpenDB. Standardně je 0 a běžne zůstává nezměněna.

Příklad použití

```
*FIXME:
```

docincluded

* *$Header: /home/radek/cvs/forth-book/dictionary/qf/dict-old/docincluded,v 1.1 2003-12-28 18:21:58 radek Exp $*

Jméno

`docincluded` — jednořádkový popis

Modul docinc

Přehled

```
docincluded   ( c-addr u ⟶ )
```

Popis

Toto slovo je definováno v modulu xref linkend="QF.MODULE.docinc"/.

Příklad použití

```
FIXME:
```

Kód slova v Qurtus Forthu

Příklad 1. *:[90]

ekey

ekey

Jméno

`ekey` — Kořenový tag knihy

Přehled

```
: ekey ( -- ekey ) ;
```

Popis

Vyzvednutí události z fronty událostí PalmOS.

Příklad použití

```
need events
: handle-events ( -- )
  begin
    ekey dispatch-event drop
  again ;
```

freeHandle

* *$Header: /home/radek/cvs/forth-book/dictionary/qf/dict-old/freeHandle,v 1.1 2003-12-28 18:21:58 radek Exp $*

Jméno

`freeHandle` — jednořádkový popis

Přehled

```
freeHandle ( handle. ⟶ ) ;
```

Popis

*FIXME:

Příklad použití

```
*FIXME:
```

Definice

Příklad 1. Definice slova freeHandle

```
: freeHandle ( handle. ⟶ )
        2DUP OR IF MemHandleFree DROP
        ELSE 2DROP THEN ;
```

itemID

* *$Header: /home/radek/cvs/forth-book/dictionary/qf/dict-old/itemid,v 1.1 2003-12-28 18:21:58 radek Exp $*

Jméno

`itemID` —

Event

Přehled

(&event. ⟶ menuitem)

Definice

Slovo je definováno v souboru `Event` takto:

Příklad 1. itemID

```
: itemID ( &event. ⟶ menuitem )
    8 M+ @a ;
```

ms

* *$Header: /home/radek/cvs/forth-book/dictionary/qf/dict-old/ms,v 1.1 2003-12-28 18:21:58 radek Exp $*

Jméno

`ms` — čekání

Přehled

```
ms  ( n ⟶ )
```

Popis

Slovo **ms** způsobí pozastavení vykonávání programu na danou dobu. Doba je zadána v milisekundách.

> **Varování**
>
> Protože Quartus Forth počítá čas v setinách sekundy, musí být čekací doba (počet milisekund) násobkem 10.

Příklad použití

```
3000 ms
```

string>Handle

* *$Header: /home/radek/cvs/forth-book/dictionary/qf/dict-old/string>Handle,v 1.1 2003-12-28 18:21:58 radek Exp $*

Jméno

`string>Handle` — jednořádkový popis

string2anyfield

Přehled

```
string>Handle ( str u ⟶ handle. ) ;
```

Popis

Definováno v modulu xref linkend="QF.MODULE.ezUI"/.

Příklad použití

FIXME:

Definice slova

Příklad 1. Definice slova string>Handle

```
: string>Handle (str u ⟶ handle. )
    DUP 1+ S>D MemHandleNew
    2DUP 2>R MemHandleLock 2>R
    DUP 0 SWAP 2R@ ROT M+ C!A
    S>D ROT >ABS 2R> MemMove DROP
    2R> 2DUP MemHandleUnlock DROP ;
```

stringfield

* *$Header: /home/radek/cvs/forth-book/dictionary/qf/dict-old/stringfield,v 1.1 2003-12-28 18:21:58 radek Exp $*

Jméno

`stringfield` —

* *FIXME:*

Přehled

stringfield (GET ⟶ str u) (str u SET ⟶)

Popis

* **FIXME:*

Příklad použití

```
*FIXME:
```

Kód slova v Qurtus Forthu

Příklad 1. *:[90]

Definice slova

Příklad 2.

* ***:[64]***

```
:  stringfield ( GET -- str u ) ( str u SET -- )
  create ,
  does>
    @ swap ( flag fieldid -- fieldid flag )
    case get of
      stringbuf dup (fieldid sb sb )
      rot           ( sb sb fieldid )
      Field>string  ( sb count )
      endof
    set of
      string>anyField endof
    ( else ) commonmethods
    endcase
;
```

VII. Události PalmOS

Reference událostí PalmOS jenž Quartus Forth podporuje.

Popis událostí jenž v PalmOS nastanou a je je možno v prostředí Quartus Forthu ošetřit/odchytit.

V této příloze popisuji události jenž nastanou v PalmOS a Quartus Forth je umí ošetřit. Popisuji postupně všechny informace které se mi o dané události podařilo zjistit a poté se je snažím upravit do dostatečně „husté/hutné“ formy a případně doplnit příklady použití.

* *$Header: /home/radek/cvs/forth-book/dictionary/qf/event/dbheader.xml,v 1.1 2003/12/28 18:21:58 radek Exp $*

Popis událostí jenž v PalmOS nastanou a je je možno v prostředí Quartus Forthu ošetřit/odchytit.

V této příloze popisuji události jenž nastanou v PalmOS a Quartus Forth je umí ošetřit. Popisuji postupně všechny informace které se mi o dané události podařilo zjistit a poté se je snažím upravit do dostatečně „husté/hutné“ formy a případně doplnit příklady použití.

1. Struktura události

Informace o vzniklé události jsou předávány aplikaci v struktuře `EventTyp` ta vypadá takto:

Příklad 1. EventType

```
typedef struct {
 eventsEnum      eType;     ❶
 Boolean         penDown;   ❷
 UInt8           tapCount;  ❸
 Int16           screenX;   ❹
 Int16           screenY;   ❺
        union {
         ...     ❻
 } data;
} EventType;
```

❶ Číslo typu události jenž nastala.

❷ Logický příznak, „flag“ označující, jestli stylus byl v okamžiku události dole `true` nebo nahoře `false`.

❸ Počet ťuknutí v této pozici. Tato hodnota je používána hlavně poli. Když uživatel ťukne do textového pole, dvě ťuknutí vyberou slovo a tři ťuknutí celou řádku.

❹ Relativní pozice pera v pixelech od levé ho okraje okna.

❺ Relativní pozice pera v pixelech od horního okraje okna.

❻ Data specifická pro danou událost, jsou li nějaká. Hodnota v poli *`eType`* určuje která data se zde nacházejí. U jednotlivých událostí dále uvedených tato data popíši.

appStopEvent

* *$Header: /home/radek/cvs/forth-book/dictionary/qf/event/appStopEvent,v 1.1 2003/12/28 18:21:58 radek Exp $*

Jméno

`appStopEvent` — `22` aplikace je informována o tom že bude zastavena

Events

Popis

Chce-li systém/uživatel spustit jinou aplikaci, právě běžící aplikace je o tom informována zprávou **appStopEvent**. Aplikace musí/je povinna v odpověď/reakci na tuto událost ukončit svůj cyklus zpracování událostí, uzavřít otevřené soubory a formuláře a ukončit se.

> **Poznámka:** Jestli se aplikace v odpověď na `appStopEvent` neukončí, systém nemůže spustit jinou aplikaci.

ctlEnterEvent

* *$Header: /home/radek/cvs/forth-book/dictionary/qf/event/ctlEnterEvent,v 1.1 2003/12/28 18:21:58 radek Exp $*

Jméno

`ctlEnterEvent` — `7`

Events

Popis

.

Příklad 1. Data události ctlEnterEvent

```
struct ctlEnter {
        UInt16          controlID;     ❶
        struct ControlType *pControl; ❷
} ctlEnter;
```

❶ Programátorem nadefinovaná identifikace ID řídícího prvku.

❷ Ukazatel na strukturu řídícího prvku.

ctlSelectEvent

* *$Header: /home/radek/cvs/forth-book/dictionary/qf/event/ctlSelectEvent,v 1.1 2003/12/28 18:21:58 radek Exp $*

Jméno

`ctlSelectEvent` — 9 aplikace je informována o tom že bude zastavena

Events

Popis

Příklad 1. Data události `ctlSelectEvent`

```
struct ctlSelect {
        UInt16          controlID;
        struct ControlType* pControl;
        UInt8           reserved1;
        UInt16          value;
} ctlSelect;
```

fldEnterEvent

* *$Header: /home/radek/cvs/forth-book/dictionary/qf/event/fldEnterEvent,v 1.1 2003/12/28 18:21:58 radek Exp $*

Jméno

`fldEnterEvent` — 15

Events

Popis

Příklad 1. Data události fldEnterEvent

```
struct fldEnter {
        UInt16          fieldID;
        struct FieldType *pField;
} fldEnter;
```

nilEvent

* *$Header: /home/radek/cvs/forth-book/dictionary/qf/event/nilEvent,v 1.1 2003/12/28 18:21:58 radek Exp $*

Jméno

`nilEvent` — prázdná událost, je vytvořena vždy když do zadaného limitu nepřijde jiná událost

Popis

Tato událost nastane, nenastala-li do daného časového intervalu událost jiná. Je to taková událost která říká že žádná událost nenastala.

Událost je použitelná pro animace, „polling“ a podobné situace.

penDownEvent

Jméno

`penDownEvent` —

Popis

Tato událost nastane, dotknete-li se perem (stylusem) displaye (obrazovky) palma.

penMoveEvent

Jméno

`penMoveEvent` —

Popis

Tato událost nastane, posunete-li pero (stylus) po displayi (obrazovce) palma.

penUpEvent

Jméno

`penUpEvent` —

Popis

Tato událost nastane, zvednete-li pero (stylus) z displaye (obrazovky) palma.

VIII. PalmOS API

Vybraná volání API PalmOS

Některá slova, jenž jsem považoval za vhodná zde uvést.

DmArchiveRecord

* *$Header: /home/radek/cvs/forth-book/dictionary/qf/palmos-api/DmArchiveRecord,v 1.1 2003/12/28 18:21:58 radek Exp $*

Jméno

`DmArchiveRecord` — FIXME: jednořádkový popis

Přehled

```
: DmArchiveRecord ( index dbP. ⟶ Err ) ;
```

Popis

FIXME: popis

Příklad použití

```
*FIXME:
```

DmAttachRecord

* *$Header: /home/radek/cvs/forth-book/dictionary/qf/palmos-api/DmAttachRecord,v 1.1 2003/12/28 18:21:58 radek Exp $*

Jméno

`DmAttachRecord` — FIXME: jednořádkový popis

Přehled

```
: DmAttachRecord  ( &oldHP. newH. &atP. dbP. ⟶ Err ) ;
```

Popis

FIXME: popis

Příklad použití

```
*FIXME:
```

DmAttachResource

* *$Header: /home/radek/cvs/forth-book/dictionary/qf/palmos-api/DmAttachResource,v 1.1 2003/12/28 18:21:58 radek Exp $*

Jméno

`DmAttachResource` — FIXME: jednořádkový popis

Přehled

```
: DmAttachResource    ( resId resType. newH. dbP. ⟶ Err ) ;
```

Popis

FIXME: popis

Příklad použití

```
*FIXME:
```

DmCloseDatabase

* *$Header: /home/radek/cvs/forth-book/dictionary/qf/palmos-api/DmCloseDatabase,v 1.1 2003/12/28 18:21:58 radek Exp $*

Jméno

`DmCloseDatabase` — FIXME: jednořádkový popis

Přehled

```
: DmCloseDatabase   ( dbP. ⟶ Err ) ;
```

Popis

FIXME: popis

Příklad použití

```
*FIXME:
```

DmCreateDatabase

* *$Header: /home/radek/cvs/forth-book/dictionary/qf/palmos-api/DmCreateDatabase,v 1.1 2003/12/28 18:21:58 radek Exp $*

Jméno

`DmCreateDatabase` — Vytvoří novou databázi na uvedné kartě, s uvedeným názvem, tvůrcem a typem.

Přehled

`DmCreateDatabase ( resDB? type. creator. &nameP. cardNo ⟶ Err )`

```
Err DmCreateDatabase(UInt16 cardNo, const Char *nameP, UInt32 creator, UInt32
type, bool resDB);
```

⟶ *`cardNo`*

Číslo paměťové karty na které chceme databázi vytvořit.

⟶ *`nameP`*

Název databáze. Název může být dlouhý nejvýše 32 znaků v četně ukončujícího znaku null. Název databáze smí obsahovat jen 7-mi bitové ASCII znaky v rozsahu 0x20 až 0x7E.

⟶ *`creator`*

Identifikace tvůrce databáze.

⟶ *`type`*

Typ databáze.

⟶ *`resDB`*

Příznak zdrojové (*resource*) databáze.

Vrací:

errNone

nebyla li žádná chyba

dmErrInvalidDatabaseName

Špatné jméno databáze

dmErrAlreadyExists

Databáze již existuje

Vrátí 0 nenastala li chyba jinak memErrIvalidParam.

Příklad použití

Použití si ukážeme například vytvoření databáze `Cvičná Databáze`. Tuto databázi vytvoříme jako obyčejnou, s identifikací tvůrce `Test` a typem databáze `Data`

```
# $Id: example:DmCreateDatabase.ses,v 1.1 2003/12/28 18:21:58 radek Exp $
needs zstrings
FALSE (ID) Data (ID) Test z" Cvičná databáze" DROP >ABS 0
DmCreateDatabase .
```

FIXME: See Also

DmDeleteDatabase, DmFindDatabase

DmCreateDatabaseFromImage

* *$Header: /home/radek/cvs/forth-book/dictionary/qf/palmos-api/DmCreateDatabaseFromImage,v 1.1 2003/12/28 18:21:58 radek Exp $*

Jméno

`DmCreateDatabaseFromImage` — FIXME: jednořádkový popis

Přehled

```
: DmCreateDatabaseFromImage   ( &bufferP. ⟶ Err ) ;
```

Popis

FIXME: popis

Příklad použití

```
*FIXME:
```

DmDatabaseInfo

* *$Header: /home/radek/cvs/forth-book/dictionary/qf/palmos-api/DmDatabaseInfo,v 1.1 2003/12/28 18:21:58 radek Exp $*

Jméno

`DmDatabaseInfo` — získání atributů databázového souboru

Přehled

```
: DmDatabaseInfo   ( &creatorP. &typeP. &sortInfoIDP. &appInfoIDP. &modNumP.
&bckUpDateP. &modDateP. &crDateP. &versionP. &attributesP. &nameP. dbID. cardNo ⟶ Err ) ;

.
```

Popis

Získání informací o databázi. Přd voláním vložíme na zásobník ukazatele na paměťové oblasti, id databáze získané například voláním DmGetDatabase a číslo paměťové karty. Získáme zpátky chybovou informaci a paměťové oblasti jsou naplněny získanými informacemi o databázi.

Příklad použití

Příklad 1. Netestovaný rozepsaný příklad použiti DmDatabaseInfo

```
\ Example_DmDatabaseInfo

2variable myCreator
2variable myType
2variable mySortInfoId
2variable myAppInfoId
2variable myModNum
2variable myBckUpDate
2variable myModDate
2variable myCrDate
variable myVersion
```

```
variable myAttributes
create myName 32 allot

myCreator >abs
myType >abs
mySortInfoId >abs
myAppInfoId >abs
myModNum >abs
myBckUpDate >abs
myModDate >abs
myCrDate >abs
myVersion >abs
myAttributes >abs
myName >abs

0 0 DmGetDatabase       ( ⟶ LocalID. )
0 DmDatabaseInfo        ( ⟶ err )
.s
.( Version:) myVersion @ .
.( Attributes:) myAttributes @ .
.( Name:) myName 32 type
```

Příklad 2. Výpis jmen všech databází

```
\ Example_ListDatabases

: 5null 0. 0. 0. 0. 0. ;
: 10null 5null 5null ;

create myName 32 allot

: spaceName
    myName
    32 0 do
        1 +
        32 over c!
    loop
    drop
;

: listDatabases
    \ naplnit myName mezerama
    0 DmNumDatabases 0 do
        spaceName
        10null myName >abs
        i 0 DmGetDatabase
        0 DmDatabaseInfo drop
        myName 32 type cr
    loop
;

listDatabases
```

Odkazy:

- DmDatabaseInfo (http://www.geocities.com/amaurycarvalho/Palm.OS.File.System.Simulator.html#robo4)

DmDatabaseProtect

* *$Header: /home/radek/cvs/forth-book/dictionary/qf/palmos-api/DmDatabaseProtect,v 1.1 2003/12/28 18:21:58 radek Exp $*

Jméno

`DmDatabaseProtect` — FIXME: jednořádkový popis

Přehled

```
: DmDatabaseProtect   ( protect? dbID. cardNo ⟶ Err ) ;
```

Popis

FIXME: popis

Příklad použití

```
*FIXME:
```

DmDatabaseSize

* *$Header: /home/radek/cvs/forth-book/dictionary/qf/palmos-api/DmDatabaseSize,v 1.1 2003/12/28 18:21:58 radek Exp $*

Jméno

`DmDatabaseSize` — FIXME: jednořádkový popis

Přehled

```
: DmDatabaseSize   ( &dataBytesP. &totalBytesP. &numRecordsP. dbID. cardNo ⟶ Err ) ;
```

Popis

FIXME: popis

Příklad použití

```
*FIXME:
```

DmDeleteCategory

* *$Header: /home/radek/cvs/forth-book/dictionary/qf/palmos-api/DmDeleteCategory,v 1.1 2003/12/28 18:21:58 radek Exp $*

Jméno

`DmDeleteCategory` — FIXME: jednořádkový popis

Přehled

```
: DmDeleteCategory   ( categoryNum dbR. ⟶ Err ) ;
```

Popis

FIXME: popis

Příklad použití

```
*FIXME:
```

DmDeleteDatabase

* *$Header: /home/radek/cvs/forth-book/dictionary/qf/palmos-api/DmDeleteDatabase,v 1.1 2003/12/28 18:21:58 radek Exp $*

Jméno

`DmDeleteDatabase` — Odstraní databázi se všemi záznamy.

Přehled

```
: DmDeleteDatabase   ( dbID. cardNo ⟶ Err )
```

```
Err DmDeleteDatabase(UInt16 cardNo, LocalID dbID);
```

⟶ *cardNo*

Číslo paměťové karty.

⟶ *dbID*

Identifikátor databáze.

Vrací:

errNone

nebyla li žádná chyba

dmErrInvalidDatabaseName

Špatné jméno databáze

dmErrCantFind

Databáze neexistuje.

Popis

*FIXME:

Příklad použití

```
*FIXME:
```

FIXME: See Also

MemHandleNew
MemHandleLock
MemHandleUnlock
MemHandleSize
MemHandleResize

DmDeleteRecord

* *$Header: /home/radek/cvs/forth-book/dictionary/qf/palmos-api/DmDeleteRecord,v 1.1 2003/12/28 18:21:58 radek Exp $*

Jméno

`DmDeleteRecord` — FIXME: jednořádkový popis

Přehled

```
: DmDeleteRecord   ( index dbP. ⟶ Err ) ;
```

Popis

FIXME: popis

Příklad použití

```
*FIXME:
```

DmDetachRecord

* *$Header: /home/radek/cvs/forth-book/dictionary/qf/palmos-api/DmDetachRecord,v 1.1 2003/12/28 18:21:58 radek Exp $*

Jméno

`DmDetachRecord` — FIXME: jednořádkový popis

Přehled

```
: DmDetachRecord   ( &oldHP. index dbP. ⟶ Err ) ;
```

Popis

FIXME: popis

Příklad použití

```
*FIXME:
```

DmDetachResource

* *$Header: /home/radek/cvs/forth-book/dictionary/qf/palmos-api/DmDetachResource,v 1.1 2003/12/28 18:21:58 radek Exp $*

Jméno

`DmDetachResource` — FIXME: jednořádkový popis

Přehled

```
: DmDetachResource   ( &oldHP. index dbP. ⟶ Err ) ;
```

Popis

FIXME: popis

Příklad použití

```
*FIXME:
```

DmFindDatabase

* *$Header: /home/radek/cvs/forth-book/dictionary/qf/palmos-api/DmFindDatabase,v 1.1 2003/12/28 18:21:58 radek Exp $*

Jméno

`DmFindDatabase` — FIXME: jednořádkový popis

Přehled

```
: DmFindDatabase   ( &nameP. cardNo ⟶ LocalID. ) ;
```

Popis

FIXME: popis

Příklad použití

```
*FIXME:
```

DmFindRecordByID

* *$Header: /home/radek/cvs/forth-book/dictionary/qf/palmos-api/DmFindRecordByID,v 1.1 2003/12/28 18:21:58 radek Exp $*

Jméno

`DmFindRecordByID` — FIXME: jednořádkový popis

Přehled

```
: DmFindRecordByID   ( &indexP. uniqueID. dbP. ⟶ Err ) ;
```

Popis

FIXME: popis

Příklad použití

```
*FIXME:
```

DmFindResource

* *$Header: /home/radek/cvs/forth-book/dictionary/qf/palmos-api/DmFindResource,v 1.1 2003/12/28 18:21:58 radek Exp $*

Jméno

`DmFindResource` — FIXME: jednořádkový popis

Přehled

```
: DmFindResource   ( resH. resID resType. dbP. ⟶ n ) ;
```

Popis

FIXME: popis

Příklad použití

```
*FIXME:
```

DmFindResourceType

* *$Header: /home/radek/cvs/forth-book/dictionary/qf/palmos-api/DmFindResourceType,v 1.1 2003/12/28 18:21:58 radek Exp $*

Jméno

`DmFindResourceType` — FIXME: jednořádkový popis

Přehled

```
: DmFindResourceType   ( typeIndex resType. dbP. ⟶ n ) ;
```

Popis

FIXME: popis

Příklad použití

```
*FIXME:
```

DmFindSortPosition

* *$Header: /home/radek/cvs/forth-book/dictionary/qf/palmos-api/DmFindSortPosition,v 1.1 2003/12/28 18:21:58 radek Exp $*

Jméno

`DmFindSortPosition` — FIXME: jednořádkový popis

Přehled

```
: DmFindSortPosition   ( other &compar. &newRecordInfo. &newRecord. dbP. ⟶ u ) ;
```

Popis

FIXME: popis

Příklad použití

```
*FIXME:
```

DmFindSortPositionV10

* *$Header: /home/radek/cvs/forth-book/dictionary/qf/palmos-api/DmFindSortPositionV10,v 1.1 2003/12/28 18:21:58 radek Exp $*

Jméno

`DmFindSortPositionV10` — FIXME: jednořádkový popis

Přehled

```
: DmFindSortPositionV10   ( other &compar. &newRecord. dbP. ⟶ u ) ;
```

Popis

FIXME: popis

Příklad použití

```
*FIXME:
```

DmGet1Resource

* *$Header: /home/radek/cvs/forth-book/dictionary/qf/palmos-api/DmGet1Resource,v 1.1 2003/12/28 18:21:58 radek Exp $*

Jméno

`DmGet1Resource` — FIXME: jednořádkový popis

Přehled

```
: DmGet1Resource   ( id type. ⟶ VoidHand. ) ;
```

Popis

FIXME: popis

Příklad použití

```
*FIXME:
```

DmGetAppInfoID

* *$Header: /home/radek/cvs/forth-book/dictionary/qf/palmos-api/DmGetAppInfoID,v 1.1 2003/12/28 18:21:58 radek Exp $*

Jméno

`DmGetAppInfoID` — FIXME: jednořádkový popis

Přehled

```
: DmGetAppInfoID   ( dbP. ⟶ LocalID. ) ;
```

Popis

FIXME: popis

Příklad použití

```
*FIXME:
```

DmGetDatabase

* *$Header: /home/radek/cvs/forth-book/dictionary/qf/palmos-api/DmGetDatabase,v 1.1 2003/12/28 18:21:58 radek Exp $*

Jméno

`DmGetDatabase` — FIXME: jednořádkový popis

Přehled

```
: DmGetDatabase   ( index cardNo ⟶ LocalID. ) ;
```

Popis

FIXME: popis

Příklad použití

```
*FIXME:
```

DmGetLastErr

* *$Header: /home/radek/cvs/forth-book/dictionary/qf/palmos-api/DmGetLastErr,v 1.1 2003/12/28 18:21:58 radek Exp $*

Jméno

`DmGetLastErr` — FIXME: jednořádkový popis

Přehled

```
: DmGetLastErr   ( ⟶ Err ) ;
```

Popis

FIXME: popis

Příklad použití

```
*FIXME:
```

DmGetNextDatabaseByTypeCreator

* *$Header: /home/radek/cvs/forth-book/dictionary/qf/palmos-api/DmGetNextDatabaseByTypeCreator,v 1.1 2003/12/28 18:21:58 radek Exp $*

Jméno

`DmGetNextDatabaseByTypeCreator` — FIXME: jednořádkový popis

Přehled

```
: DmGetNextDatabaseByTypeCreator   ( &dbIDP. &cardNoP. onlyLatestVers? creator. type. &stateIn
```

Popis

FIXME: popis

Příklad použití

```
*FIXME:
```

DmGetRecord

* *$Header: /home/radek/cvs/forth-book/dictionary/qf/palmos-api/DmGetRecord,v 1.1 2003/12/28 18:21:58 radek Exp $*

Jméno

`DmGetRecord` — FIXME: jednořádkový popis

Přehled

```
: DmGetRecord   ( index dbP. ⟶ VoidHand. ) ;
```

Popis

FIXME: popis

Příklad použití

```
*FIXME:
```

DmGetResource

* *$Header: /home/radek/cvs/forth-book/dictionary/qf/palmos-api/DmGetResource,v 1.1 2003/12/28 18:21:58 radek Exp $*

Jméno

`DmGetResource` — FIXME: jednořádkový popis

Přehled

```
: DmGetResource   ( id type. ⟶ VoidHand. ) ;
```

Popis

FIXME: popis

Příklad použití

```
*FIXME:
```

DmGetResourceIndex

* *$Header: /home/radek/cvs/forth-book/dictionary/qf/palmos-api/DmGetResourceIndex,v 1.1 2003/12/28 18:21:58 radek Exp $*

Jméno

`DmGetResourceIndex` — FIXME: jednořádkový popis

Přehled

```
: DmGetResourceIndex   ( index dbP. ⟶ VoidHand. ) ;
```

Popis

FIXME: popis

Příklad použití

```
*FIXME:
```

DmInit

* *$Header: /home/radek/cvs/forth-book/dictionary/qf/palmos-api/DmInit,v 1.1 2003/12/28 18:21:58 radek Exp $*

Jméno

`DmInit` — FIXME: jednořádkový popis

Přehled

```
: DmInit   ( ⟶ Err ) ;
```

Popis

FIXME: popis

Příklad použití

```
*FIXME:
```

DmInsertionSort

* *$Header: /home/radek/cvs/forth-book/dictionary/qf/palmos-api/DmInsertionSort,v 1.1 2003/12/28 18:21:58 radek Exp $*

Jméno

`DmInsertionSort` — FIXME: jednořádkový popis

Přehled

```
: DmInsertionSort   ( other &compar. dbR. ⟶ Err ) ;
```

Popis

FIXME: popis

Příklad použití

```
*FIXME:
```

DmMoveCategory

* *$Header: /home/radek/cvs/forth-book/dictionary/qf/palmos-api/DmMoveCategory,v 1.1 2003/12/28 18:21:58 radek Exp $*

Jméno

`DmMoveCategory` — FIXME: jednořádkový popis

Přehled

```
: DmMoveCategory   ( dirty? fromCategory toCategory dbP. ⟶ Err ) ;
```

Popis

FIXME: popis

Příklad použití

```
*FIXME:
```

DmMoveOpenDBContext

* *$Header: /home/radek/cvs/forth-book/dictionary/qf/palmos-api/DmMoveOpenDBContext,v 1.1 2003/12/28 18:21:58 radek Exp $*

Jméno

`DmMoveOpenDBContext` — FIXME: jednořádkový popis

Přehled

```
: DmMoveOpenDBContext   ( &dbP. &listHeadP. ⟶ Err ) ;
```

Popis

FIXME: popis

Příklad použití

```
*FIXME:
```

DmMoveRecord

* *$Header: /home/radek/cvs/forth-book/dictionary/qf/palmos-api/DmMoveRecord,v 1.1 2003/12/28 18:21:58 radek Exp $*

Jméno

`DmMoveRecord` — FIXME: jednořádkový popis

Přehled

```
: DmMoveRecord   ( to from dbP. ⟶ Err ) ;
```

Popis

FIXME: popis

Příklad použití

```
*FIXME:
```

DmNewHandle

* *$Header: /home/radek/cvs/forth-book/dictionary/qf/palmos-api/DmNewHandle,v 1.1 2003/12/28 18:21:58 radek Exp $*

Jméno

`DmNewHandle` — FIXME: jednořádkový popis

Přehled

```
: DmNewHandle   ( size. dbP. ⟶ VoidHand. ) ;
```

Popis

FIXME: popis

Příklad použití

```
*FIXME:
```

DmNewRecord

* *$Header: /home/radek/cvs/forth-book/dictionary/qf/palmos-api/DmNewRecord,v 1.1 2003/12/28 18:21:58 radek Exp $*

Jméno

`DmNewRecord` — vytvoření nového záznamu, vyhrazení místa pro něj

Přehled

```
: DmNewRecord   ( size. &atP. dbP. ⟶ VoidHand. ) ;
```

Vytvoří nový záznam a vyhradí pro něj požadované místo určené parametrem *size.*

Popis

FIXME: popis

Příklad použití

```
*FIXME:
```

DmNewResource

* *$Header: /home/radek/cvs/forth-book/dictionary/qf/palmos-api/DmNewResource,v 1.1 2003/12/28 18:21:58 radek Exp $*

Jméno

`DmNewResource` — FIXME: jednořádkový popis

Přehled

```
: DmNewResource   ( size. resID resType. dbP. ⟶ VoidHand. ) ;
```

Popis

FIXME: popis

Příklad použití

```
*FIXME:
```

DmNextOpenDatabase

* *$Header: /home/radek/cvs/forth-book/dictionary/qf/palmos-api/DmNextOpenDatabase,v 1.1 2003/12/28 18:21:58 radek Exp $*

Jméno

`DmNextOpenDatabase` — FIXME: jednořádkový popis

Přehled

```
: DmNextOpenDatabase   ( currentP. ⟶ DmOpenRef. ) ;
```

Popis

FIXME: popis

Příklad použití

```
*FIXME:
```

DmNextOpenResDatabase

* *$Header: /home/radek/cvs/forth-book/dictionary/qf/palmos-api/DmNextOpenResDatabase,v 1.1 2003/12/28 18:21:58 radek Exp $*

Jméno

`DmNextOpenResDatabase` — FIXME: jednořádkový popis

Přehled

```
: DmNextOpenResDatabase   ( dbP. ⟶ DmOpenRef. ) ;
```

Popis

FIXME: popis

Příklad použití

```
*FIXME:
```

DmNumDatabases

* *$Header: /home/radek/cvs/forth-book/dictionary/qf/palmos-api/DmNumDatabases,v 1.1 2003/12/28 18:21:58 radek Exp $*

Jméno

`DmNumDatabases` — FIXME: jednořádkový popis

Přehled

```
: DmNumDatabases   ( cardNo ⟶ u ) ;
```

Popis

Funkce/Slovo vrací počet databází na paměťové kartě s číslem *`cardNo`*.

You can get the total number of installed databases (including applications, which are just a special type of database) with the DmNumDatabases function.

Příklad použití

```
0 DmNumDatabases .
349 ok.
```

Odkazy:

- Basic Database Management Under the Palm OS (http://www.developer.com/db/article.php/2197021)

DmNumRecords

* *$Header: /home/radek/cvs/forth-book/dictionary/qf/palmos-api/DmNumRecords,v 1.1 2003/12/28 18:21:58 radek Exp $*

Jméno

`DmNumRecords` — FIXME: jednořádkový popis

Přehled

```
: DmNumRecords   ( dbP. ⟶ u ) ;
```

Popis

FIXME: popis

Příklad použití

```
*FIXME:
```

DmNumRecordsInCategory

* *$Header: /home/radek/cvs/forth-book/dictionary/qf/palmos-api/DmNumRecordsInCategory,v 1.1 2003/12/28 18:21:58 radek Exp $*

Jméno

`DmNumRecordsInCategory` — FIXME: jednořádkový popis

Přehled

```
: DmNumRecordsInCategory   ( category dbP. ⟶ u ) ;
```

Popis

FIXME: popis

Příklad použití

```
*FIXME:
```

DmNumResources

* *$Header: /home/radek/cvs/forth-book/dictionary/qf/palmos-api/DmNumResources,v 1.1 2003/12/28 18:21:58 radek Exp $*

Jméno

`DmNumResources` — FIXME: jednořádkový popis

Přehled

```
: DmNumResources   ( dbP. ⟶ u ) ;
```

Popis

FIXME: popis

Příklad použití

```
*FIXME:
```

DmOpenDatabase

* *$Header: /home/radek/cvs/forth-book/dictionary/qf/palmos-api/DmOpenDatabase,v 1.1 2003/12/28 18:21:58 radek Exp $*

Jméno

`DmOpenDatabase` — FIXME: jednořádkový popis

Přehled

```
: DmOpenDatabse   ( mode dbID. cardNo ⟶ DmOpenRef. ) ;
```

Popis

FIXME: popis

Příklad použití

```
*FIXME:
```

DmOpenDatabaseByTypeCreator

* *$Header: /home/radek/cvs/forth-book/dictionary/qf/palmos-api/DmOpenDatabaseByTypeCreator,v 1.1 2003/12/28 18:21:58 radek Exp $*

Jméno

`DmOpenDatabaseByTypeCreator` — FIXME: jednořádkový popis

Přehled

```
: DmOpenDatabseByTypeCreator   ( mode creator. type. ⟶ DmOpenRef. ) ;
```

Popis

FIXME: popis

Příklad použití

```
*FIXME:
```

DmOpenDatabaseInfo

* *$Header: /home/radek/cvs/forth-book/dictionary/qf/palmos-api/DmOpenDatabaseInfo,v 1.1 2003/12/28 18:21:58 radek Exp $*

Jméno

`DmOpenDatabaseInfo` — FIXME: jednořádkový popis

Přehled

```
: DmOpenDatabaseInfo   ( &resDBP. &cardNoP. &modeP. &openCountP. &dbIDP. dbP. ⟶ Err ) ;
```

Popis

FIXME: popis

Příklad použití

```
*FIXME:
```

DmPositionInCategory

* *$Header: /home/radek/cvs/forth-book/dictionary/qf/palmos-api/DmPositionInCategory,v 1.1 2003/12/28 18:21:58 radek Exp $*

Jméno

`DmPositionInCategory` — FIXME: jednořádkový popis

Přehled

```
: DmPositionInCategory   ( category index dbP. ⟶ u ) ;
```

Popis

FIXME: popis

Příklad použití

```
*FIXME:
```

DmQueryNextInCategory

* *$Header: /home/radek/cvs/forth-book/dictionary/qf/palmos-api/DmQueryNextInCategory,v 1.1 2003/12/28 18:21:58 radek Exp $*

Jméno

`DmQueryNextInCategory` — FIXME: jednořádkový popis

Přehled

```
: DmQueryNextInCategory   ( category &indexP. dbP. ⟶ VoidHand. ) ;
```

Popis

FIXME: popis

Příklad použití

```
*FIXME:
```

DmQueryRecord

* *$Header: /home/radek/cvs/forth-book/dictionary/qf/palmos-api/DmQueryRecord,v 1.1 2003/12/28 18:21:58 radek Exp $*

Jméno

`DmQueryRecord` — FIXME: jednořádkový popis

Přehled

```
: DmQueryRecord   ( index dbP. ⟶ VoidHand. ) ;
```

Popis

FIXME: popis

Příklad použití

```
*FIXME:
```

DmQuickSort

* *$Header: /home/radek/cvs/forth-book/dictionary/qf/palmos-api/DmQuickSort,v 1.1 2003/12/28 18:21:58 radek Exp $*

Jméno

`DmQuickSort` — FIXME: jednořádkový popis

Přehled

```
: DmQuickSort   ( index dbP. ⟶ VoidHand. ) ;
```

Popis

FIXME: popis

Příklad použití

```
*FIXME:
```

DmRecordInfo

* *$Header: /home/radek/cvs/forth-book/dictionary/qf/palmos-api/DmRecordInfo,v 1.1 2003/12/28 18:21:58 radek Exp $*

Jméno

`DmRecordInfo` — atributy a informace o záznamu

Přehled

```
: DmRecordInfo   ( &chunkIDP. &uniqueIDP. &attrP. index dbP. ⟶ Err ) ;
```

Získání informací o záznamu. *`dbP.`* je ovladač databáze a *`index`* je číslo záznamu. Získané informace jsou v polích na které ukazují zbývající parametry. Pokud je některý z těchto parametrů `0.` není příslušná informace vyplněna.

Příklad použití

```
*FIXME: doplnit
```

DmReleaseRecord

* *$Header: /home/radek/cvs/forth-book/dictionary/qf/palmos-api/DmReleaseRecord,v 1.1 2003/12/28 18:21:58 radek Exp $*

Jméno

`DmReleaseRecord` — FIXME: jednořádkový popis

Přehled

```
: DmReleaseRecord   ( dirty? index dbP. ⟶ Err ) ;
```

Popis

FIXME: popis

Příklad použití

```
*FIXME:
```

DmReleaseResource

* *$Header: /home/radek/cvs/forth-book/dictionary/qf/palmos-api/DmReleaseResource,v 1.1 2003/12/28 18:21:58 radek Exp $*

Jméno

`DmReleaseResource` — FIXME: jednořádkový popis

Přehled

```
: DmReleaseResource   ( resourceH. ⟶ Err ) ;
```

Popis

FIXME: popis

Příklad použití

```
*FIXME:
```

DmRemoveRecord

* *$Header: /home/radek/cvs/forth-book/dictionary/qf/palmos-api/DmRemoveRecord,v 1.1 2003/12/28 18:21:58 radek Exp $*

Jméno

`DmRemoveRecord` — FIXME: jednořádkový popis

Přehled

```
: DmRemoveRecord   ( index dbP. ⟶ Err ) ;
```

Popis

FIXME: popis

Příklad použití

```
*FIXME:
```

DmRemoveResource

* *$Header: /home/radek/cvs/forth-book/dictionary/qf/palmos-api/DmRemoveResource,v 1.1 2003/12/28 18:21:58 radek Exp $*

Jméno

`DmRemoveResource` — FIXME: jednořádkový popis

Přehled

```
: DmRemoveResource   ( index dbP. ⟶ Err ) ;
```

Popis

FIXME: popis

Příklad použití

```
*FIXME:
```

DmRemoveSecretRecords

* *$Header: /home/radek/cvs/forth-book/dictionary/qf/palmos-api/DmRemoveSecretRecords,v 1.1 2003/12/28 18:21:58 radek Exp $*

Jméno

`DmRemoveSecretRecords` — FIXME: jednořádkový popis

Přehled

```
: DmRemoveSecretRecords   ( dbP. ⟶ Err ) ;
```

Popis

FIXME: popis

Příklad použití

```
*FIXME:
```

DmResetRecordStates

* *$Header: /home/radek/cvs/forth-book/dictionary/qf/palmos-api/DmResetRecordStates,v 1.1 2003/12/28 18:21:58 radek Exp $*

Jméno

`DmResetRecordStates` — FIXME: jednořádkový popis

Přehled

```
: DmResetRecordStates   ( dbP. ⟶ Err ) ;
```

Popis

FIXME: popis

Příklad použití

```
*FIXME:
```

DmResizeRecord

* *$Header: /home/radek/cvs/forth-book/dictionary/qf/palmos-api/DmResizeRecord,v 1.1 2003/12/28 18:21:58 radek Exp $*

Jméno

`DmResizeRecord` — FIXME: jednořádkový popis

Přehled

```
: DmResizeRecord   ( newSize. index dbP. ⟶ VoidHand. ) ;
```

Popis

FIXME: popis

Příklad použití

```
*FIXME:
```

DmResizeResource

* *$Header: /home/radek/cvs/forth-book/dictionary/qf/palmos-api/DmResizeResource,v 1.1 2003/12/28 18:21:58 radek Exp $*

Jméno

`DmResizeResource` — FIXME: jednořádkový popis

Přehled

```
: DmResizeResource   ( newSize. resourceH. ⟶ VoidHand. ) ;
```

Popis

FIXME: popis

Příklad použití

```
*FIXME:
```

DmResourceInfo

* *$Header: /home/radek/cvs/forth-book/dictionary/qf/palmos-api/DmResourceInfo,v 1.1 2003/12/28 18:21:58 radek Exp $*

Jméno

`DmResourceInfo` — FIXME: jednořádkový popis

Přehled

```
: DmResourceInfo   ( &chunkLocalIDP. &resIDP. &resTypeP. index dbP. ⟶ Err ) ;
```

Popis

FIXME: popis

Příklad použití

```
*FIXME:
```

DmSearchRecord

* *$Header: /home/radek/cvs/forth-book/dictionary/qf/palmos-api/DmSearchRecord,v 1.1 2003/12/28 18:21:58 radek Exp $*

Jméno

`DmSearchRecord` — FIXME: jednořádkový popis

Přehled

```
: DmSearchRecord   ( &dbPP. recH. ⟶ n ) ;
```

Popis

FIXME: popis

Příklad použití

```
*FIXME:
```

DmSearchResource

* *$Header: /home/radek/cvs/forth-book/dictionary/qf/palmos-api/DmSearchResource,v 1.1 2003/12/28 18:21:58 radek Exp $*

Jméno

`DmSearchResource` — FIXME: jednořádkový popis

Přehled

```
: DmSearchResource   ( &dbP. resH. resID resType. ⟶ n ) ;
```

Popis

FIXME: popis

Příklad použití

```
*FIXME:
```

DmSeekRecordInCategory

* *$Header: /home/radek/cvs/forth-book/dictionary/qf/palmos-api/DmSeekRecordInCategory,v 1.1 2003/12/28 18:21:58 radek Exp $*

Jméno

`DmSeekRecordInCategory` — FIXME: jednořádkový popis

Přehled

```
: DmSeekRecordInCategory   ( category direction offset &indexP. dbP. ⟶ Err ) ;
```

Popis

FIXME: popis

Příklad použití

```
*FIXME:
```

DmSet

* *$Header: /home/radek/cvs/forth-book/dictionary/qf/palmos-api/DmSet,v 1.1 2003/12/28 18:21:58 radek Exp $*

Jméno

`DmSet` — Zapsání speciální hodnoty do části záznamu.

Přehled

```
: DmSet   ( value[>byte] bytes. offset. &recordP. ⟶ Err )
```

```
Err DmSet(void *recordP, UInt32 offset, UInt32 bytes, UInt8 value);
```

⟶ *recordP*

Ukazatel na uzamčený záznam (chunk pointer).

⟶ *offset*

Posunutí (*offset*) od začátku záznamu.

⟶ bytes

Počet bajtů k zápisu.

⟶ value

Hodnota k zápisu.

Vrací errNone nenastala-li chyba. Může způsobit fatální chybu je li záznam neplatný nebo přepíše-li funkce záznam.

Popis

Nastaví část záznamu na danou hodnotu.

Příklad použití

```
*FIXME:
```

DmSetDatabaseInfo

* *$Header: /home/radek/cvs/forth-book/dictionary/qf/palmos-api/DmSetDatabaseInfo,v 1.1 2003/12/28 18:21:58 radek Exp $*

Jméno

`DmSetDatabaseInfo` — Nastavení informací o databázi

Přehled

```
: DmSetDatabaseInfo   ( &creatorP. &typeP. &sortInfoIDP. &appInfoIDP. &modNumP. &bckUpDateP. &
```

```
Err DmSetDatabaseInfo(UInt16 cardNo, LocalID dbID, const Char *nameP, UInt16*
atributesP, UInt16* versionP, UInt32* crDateP, UInt32* modDateP, UInt32*
heapListOffsetP, UInt32* bckUpDateP, UInt32* modNumP, LocalID* appInfoIDP,
LocalID* sortInfoIDP, UInt32* typeP, UInt32* creatorP);
```

cardNo

Číslo karty na které je databáze.

dbID

Database ID of the database.

nameP

Pointer to 32-byte character array for new name, or NULL.

attributesP

Pointer to new attributes variable, or NULL. See „Database Attribute Constants“ for a list of possible values.

Vrací 0 nenastala-li chyba, nebo memErrCardNotPresent, memErrRAMOnlyCard, memErrInvalidStoreHeader nastala-li chyba.

Popis

Vyzvednutí události z fronty událostí PalmOS.

Příklad použití

```
*FIXME:
```

DmSetRecordInfo

* *$Header: /home/radek/cvs/forth-book/dictionary/qf/palmos-api/DmSetRecordInfo,v 1.1 2003/12/28 18:21:58 radek Exp $*

Jméno

`DmSetRecordInfo` — FIXME: jednořádkový popis

Přehled

```
: DmSetRecordInfo   ( &uniqueIDP. &attrP. index dbP. ⟶ Err ) ;
```

Popis

FIXME: popis

Příklad použití

```
*FIXME:
```

DmSetResourceInfo

* *$Header: /home/radek/cvs/forth-book/dictionary/qf/palmos-api/DmSetResourceInfo,v 1.1 2003/12/28 18:21:58 radek Exp $*

Jméno

`DmSetResourceInfo` — FIXME: jednořádkový popis

Přehled

```
: DmSetResourceInfo   ( &resIDP. &resTypeP. index dbP. ⟶ Err ) ;
```

Popis

FIXME: popis

Příklad použití

```
*FIXME:
```

DmStrCopy

* *$Header: /home/radek/cvs/forth-book/dictionary/qf/palmos-api/DmStrCopy,v 1.1 2003/12/28 18:21:58 radek Exp $*

Jméno

`DmStrCopy` — FIXME: jednořádkový popis

Přehled

```
: DmStrCopy   ( &srcP. offset. &ercordP. ⟶ Err ) ;
```

Popis

FIXME: popis

Příklad použití

```
*FIXME:
```

DmWrite

* *$Header: /home/radek/cvs/forth-book/dictionary/qf/palmos-api/DmWrite,v 1.1 2003/12/28 18:21:58 radek Exp $*

Jméno

`DmWrite` — FIXME: jednořádkový popis

Přehled

```
: DmWrite   ( bytes. &srcP. offset. &recordP. ⟶ Err ) ;
```

Popis

FIXME: popis

Příklad použití

```
*FIXME:
```

DmWriteCheck

* *$Header: /home/radek/cvs/forth-book/dictionary/qf/palmos-api/DmWriteCheck,v 1.1 2003/12/28 18:21:58 radek Exp $*

Jméno

`DmWriteCheck` — FIXME: jednořádkový popis

Přehled

```
: DmWriteCheck   ( bytes. offset. &recordP. ⟶ Err ) ;
```

Popis

FIXME: popis

Příklad použití

```
*FIXME:
```

MemCardInfo

* *$Header: /home/radek/cvs/forth-book/dictionary/qf/palmos-api/MemCardInfo,v 1.1 2003/12/28 18:21:58 radek Exp $*

Jméno

`MemCardInfo` — Vrací informace o paměťové kartě.

MemoryManager

Přehled

```
MemCardInfo   ( &freeBytesP. &ramSizeP. &romSizeP. &crDateP. &versionP.
                &manufNameP. &cardNameP. cardNo ⟶ Err )
```

```
Err MemCardInfo(UInt16 cardNo, Char* cardNameP, Char* manufNameP, UInt16*
versionP, UInt32* crDateP, UInt32* romSizeP, UInt32* ramSizeP, UInt32*
freeBytesP);
```

Vrací 0 nenastala-li chyba.

Popis

Toto volání slouží k získání informací o paměti na kartě. Nezajímají li nás některé hodnoty, pak jako ukaztel předáme 0.

Příklad použití

```
\ Example:MemCardInfo
needs double

create cardName 32 allot
create manufName 32 allot
variable version
```

```
2variable crDate
2variable romSize
2variable ramSize
2variable freeBytes

freeBytes >abs
ramSize >abs
romSize >abs
crDate >abs
version >abs
manufName >abs
cardName >abs

0 MemCardInfo .

.( cardName:) cardName 32 type cr   \*FIXME:
.( manufName:) manufName 32 type cr \*FIXME:
.( version:) version @ . cr
.( crDate:) crDate 2@ d. cr
.( romSize:) romSize 2@ d. cr
.( ramSize:) ramSize 2@ d. cr
.( freeBytes:) freeBytes 2@ d. cr
```

MemCmp

* *$Header: /home/radek/cvs/forth-book/dictionary/qf/palmos-api/MemCmp,v 1.1 2003/12/28 18:21:58 radek Exp $*

Jméno

`MemCmp` — Porovnání dvou bloků paměti.

MemoryManager

Přehled

```
MemCmp   ( numBytes. &s2. &s1. ⟶ n )
```

```
Int16 MemCmp(const void* s1, const void* s2, Int32 numBytes);
```

Vrací 0 mají-li bloky shodný obsah, kladné číslo když s1 > s2 a záporné číslo, když s1 < s2.

Popis

*FIXME:

Příklad použití

```
*FIXME:
```

MemHandleFree

* *$Header: /home/radek/cvs/forth-book/dictionary/qf/palmos-api/MemHandleFree,v 1.1 2003/12/28 18:21:58 radek Exp $*

Jméno

`MemHandleFree` — Uvolní/rozpustí (dispose) posouvatelný kousek paměti (movable chunk).

Přehled

```
MemHandleFree ( handle. ⟶ 0|err )
```

```
Err MemHandleFree(MemHandle h);
```

⟶ *h*

Chunk handle, ovladač kousku paměti.

Vrátí 0 nenastala li chyba jinak memErrIvalidParam.

Popis

*FIXME:

Příklad použití

```
*FIXME:
```

FIXME: See Also

MemHandleNew
MemHandleLock
MemHandleUnlock
MemHandleSize
MemHandleResize

MemHandleLock

* *$Header: /home/radek/cvs/forth-book/dictionary/qf/palmos-api/MemHandleLock,v 1.1 2003/12/28 18:21:58 radek Exp $*

Jméno

`MemHandleLock` — Allocate a new movable chunk in the dynamic heap and returns a handle to it.

Přehled

```
MemHandleLock ( handle. ⟶ addr. )
```

```
MemPtr MemHandleLock(MemHandle h);
```

⟶ *h*

Chunk handle, ovladač kousku paměti.

Vrací ukazatel na kousek paměti.

Popis

*FIXME:

Příklad použití

```
*FIXME:
```

FIXME: See Also

MemHandleNew
MemHandleUnlock

MemHandleNew

* *$Header: /home/radek/cvs/forth-book/dictionary/qf/palmos-api/MemHandleNew,v 1.1 2003/12/28 18:21:58 radek Exp $*

Jméno

`MemHandleNew` — Allocate a new movable chunk in the dynamic heap and returns a handle to it.

Přehled

```
MemHandleNew ( size. ⟶ VoidHand. )
```

```
MemHandle MemHandleNew(UInt32 size);
```

cardNo

Číslo karty.

vardNameP

Ukazatel na pole znaků (32 bytes), nebo 0.

Vrací hadler/ovladač nového chunku/kousku paměti, nebo 0 při neuspěchu.

Popis

*FIXME:

Příklad použití

```
*FIXME:
```

MemHandleResize

* *$Header: /home/radek/cvs/forth-book/dictionary/qf/palmos-api/MemHandleResize,v 1.1 2003/12/28 18:21:58 radek Exp $*

Jméno

`MemHandleResize` — Změní velikost kousku paměti (chunk).

Přehled

```
MemHandleResize ( newSize. handle. ⟶ 0 | err )
```

```
Err MemHandleResize(MemHandle h, UInt32 newSize);
```

⟶ *h*

Chunk handle, ovladač kousku paměti.

⟶ *newSize*

Nově požadovaná velikost kousku paměti.

Vrací

0

Bez chyby

memErrInvalidParam

Neplatný parametr.

memErrNotEnoughSpace

Není sdostatek paměti na haldě pro zvetšení kousku paměti.

memErrChunkLocked

Není možno zvětšit kousek paměti, protože je uzamčen.

Popis

*FIXME:

Příklad použití

```
*FIXME:
```

FIXME: See Also

MemHandleNew
MemHandleFree
MemHandleLock
MemHandleUnlock
MemHandleSize

MemHandleSize

* *$Header: /home/radek/cvs/forth-book/dictionary/qf/palmos-api/MemHandleSize,v 1.1 2003/12/28 18:21:58 radek Exp $*

Jméno

`MemHandleSize` — Zjistí velikost kousku paměti (chunk).

Přehled

```
MemHandleSize ( handle. ⟶ size. )
```

```
UInt32 MemHandleSize(MemHandle h);
```

⟶ *h*

Chunk handle, ovladač kousku paměti.

Vrátí velikost kousku paměti.

Popis

*FIXME:

Příklad použití

```
*FIXME:
```

FIXME: See Also

MemHandleNew
MemHandleFree
MemHandleLock
MemHandleUnlock
MemHandleResize

MemHandleUnlock

* *$Header: /home/radek/cvs/forth-book/dictionary/qf/palmos-api/MemHandleUnlock,v 1.1 2003/12/28 18:21:58 radek Exp $*

Jméno

`MemHandleUnlock` — Uvolnění/odemčení kousku paměti (chunk).

Přehled

```
MemHandleUnlock ( handle. ⟶ 0 | err )
```

```
Err MemHandleUnlock(MemHandle h);
```

⟶ *h*

Chunk handle, ovladač kousku paměti.

Vrací 0 nevyskytla-li se chaba, jinak vrací číslo chyby.

Popis

*FIXME:

Příklad použití

```
*FIXME:
```

FIXME: See Also

MemHandleNew
MemHandleLock

MemPtrNew

* *$Header: /home/radek/cvs/forth-book/dictionary/qf/palmos-api/MemPtrNew,v 1.1 2003/12/28 18:21:58 radek Exp $*

Jméno

`MemPtrNew` — Vyhrazení (alokování) nového neposouvatelného kousku paměti na dynamické haldě.

Přehled

```
MemPtrNew ( size. ⟶ addr. )
```

```
MemPtr MemPtrNew(UInt32 size);
```

size

Požadovaná velikost kousku paměti.

Vrací ukazatel na nový kousek paměti, nebo 0 při neúspěchu.

Popis

*FIXME:

Příklad použití

```
*FIXME:
```

MemStoreInfo

* *$Header: /home/radek/cvs/forth-book/dictionary/qf/palmos-api/MemStoreInfo,v 1.1 2003/12/28 18:21:58 radek Exp $*

Jméno

`MemStoreInfo` — Return information on either RAM store or the ROM store for memory card.

MemoryManager

Přehled

```
MemStoreInfo   ( &datbaseDirIDP. &initCodeOffset2P. &initCodeOffset1P.
                 &heapListOffsetP. &bckUpDateP. &crDateP. &nameP. &flagsP.
                 &versionP. storeNumber cardNo ⟶ Err )
```

```
Err MemStoreInfo(UInt16 cardNo, UInt16 storeNumber, UInt16* versionP,
UInt16* flagsP, Char* nameP, UInt32* crDateP, UInt32* bckUpDateP, UInt32*
heapListOffsetP, UInt32* intCodeOffset1P, UInt32* intCodeOffset2P, LocalID*
databaseDirIDP);
```

Vrací 0 nenastala-li chyba, nebo memErrCardNotPresent, memErrRAMOnlyCard, memErrInvalidStoreHeader nastala-li chyba.

Popis

Toto volání slouží k získání informací o paměti na kartě. Nezajímají li nás některé hodnoty, pak jako ukaztel předáme 0.

Příklad použití

```
*FIXME:
```

WinDrawPixel

* *$Header: /home/radek/cvs/forth-book/dictionary/qf/palmos-api/WinDrawPixel,v 1.1 2003/12/28 18:21:58 radek Exp $*

Jméno

`WinDrawPixel` — jednořádkový popis Systrap 0xA383

Přehled

```
: slovo ( zásobníkový efekt ) ;
```

```
WinDrawPixel(UInt16 x, UInt16 y);
```

Popis

Vyzvednutí události z fronty událostí PalmOS.

Příklad použití

```
*FIXME:
```

Definice slova

Příklad 1. Definice slova WinDrawPixel

***** ***[64]***

```
: WinDrawPixel ( y x ⟶ )
  (hex) a383 systrap 2drop ; inline
```

WinErasePixel

* *$Header: /home/radek/cvs/forth-book/dictionary/qf/palmos-api/WinErasePixel,v 1.1 2003/12/28 18:21:58 radek Exp $*

Jméno

`WinErasePixel` — jednořádkový popis Systrap 0xA384

Přehled

```
: slovo ( zásobníkový efekt ) ;
```

```
WinErasePixel(UInt16 x, UInt16 y);
```

Popis

Vyzvednutí události z fronty událostí PalmOS.

Příklad použití

```
*FIXME:
```

Definice slova

Příklad 1. Definice slova WinErasePixel

***** ***[64]***

```
: WinErasePixel ( y x ⟶ )
  (hex) a384 systrap 2drop ; inline
```

IX. Instrukce rodiny procesor<65533> Motorola MC68000

* *Tato <65533><65533>st by m<65533>la j<65533>t p<65533>em<65533>stnit. Nejlep<65533><65533>m m<65533>stem by byla <65533><65533>st o procesorech (../electronics/processors.html), v knize Elektronika (../electronics/index.html)*

V t<65533>to <65533><65533>sti uv<65533>d<65533>m popis n<65533>kter<65533> instrukc<65533> rodiny procesor<65533> Motorola MC68000. Tato jsou zde uvedena jen kv<65533>li hypertextov<65533>m odkaz<65533>m.

Popis n<65533>kter<65533>ch/vybran<65533>ch instrukc<65533> rodiny procesor<65533> Motorola 68k. Instrukce jsou <65533>azeny abecedn<65533>.

ADDQ

* *$Header: /home/radek/cvs/forth-book/db-mc68k-inst/ADDQ,v 1.4 2003/12/28 18:21:56 radek Exp $*

Jméno

ADDQ — Add Quick

Integer Arithmetic Operation

Přehled

```
ADDQ #data ea
```

Popis

P<65533>i<65533>te p<65533><65533>mou hodnotu (1 a<65533> 8) k operandu specifikovan<65533>mu efektivn<65533> adresou *ea*.

P<65533><65533>klad

```
ADDQ.L  #4, A7
```

P<65533><65533>buzn<65533> instrukce

ADD, ADDA, ADDI, ADDX, CLR, CMP, CMPA, CMPI, CMPM, CMP2, DIVS/DIVU, DIVSL/DIVUL, EXT, EXTB, MULS/MULU, NEG, NEGX, SUB, SUBA, SUBI, SUBQ, SUBX

BEQ

* *$Header: /home/radek/cvs/forth-book/db-mc68k-inst/BEQ,v 1.3 2003/12/28 18:21:56 radek Exp $*

Jméno

BEQ — Branch Equal viz. Bcc

Program Control Operation

P<65533><65533>buzn<65533> instrukce

JMP, BRA, JSR, BSR, RTS

BRA

* *$Header: /home/radek/cvs/forth-book/db-mc68k-inst/BRA,v 1.3 2003/12/28 18:21:56 radek Exp $*

Jméno

BRA — Branch Always

skupina instrukc<65533>

Přehled

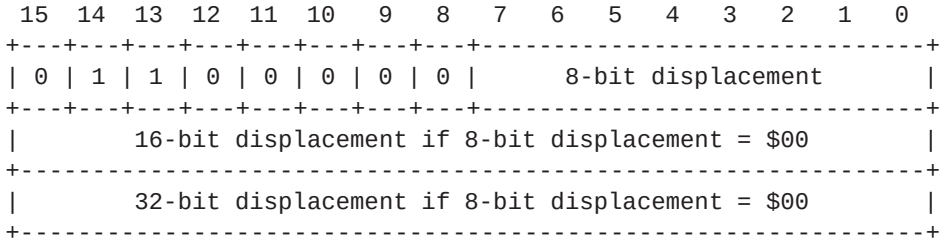
BRA *label*

Popis

Instrukce vykon<65533> skok / p<65533>evede <65533><65533>zen<65533> programu na adresu danou aktu<65533>ln<65533> adresou a uveden<65533>m posunut<65533>m.

$PC + d_n \longrightarrow PC$

Form<65533>t instrukce



P<65533><65533>klad

*FIXME:uk<65533>zka pou<65533>it<65533>

P<65533><65533>buzn<65533> instrukce

JMP, Bcc

BSR

* *$Header: /home/radek/cvs/forth-book/db-mc68k-inst/BSR,v 1.3 2003/12/28 18:21:56 radek Exp $*

Jméno

BSR — Branch to Subroutine

skupina instrukc<65533>

Přehled

BSR *label*

Popis

Instrukce vykon<65533> skok / p<65533>evede <65533><65533>zen<65533> do podprogramu na adrese dan<65533> aktu<65533>ln<65533> adresou a uveden<65533>m posunut<65533>m.

SP - 4 ⟶ SP; PC ⟶ (SP); PC + d_n ⟶ PC

Form<65533>t instrukce

```
 15  14  13  12  11  10   9   8   7   6   5   4   3   2   1   0
+---+---+---+---+---+---+---+---+-------------------------------+
| 0 | 1 | 1 | 0 | 0 | 0 | 0 | 1 |      8-bit displacement       |
+---+---+---+---+---+---+---+---+-------------------------------+
|        16-bit displacement if 8-bit displacement = $00        |
+---------------------------------------------------------------+
|        32-bit displacement if 8-bit displacement = $00        |
+---------------------------------------------------------------+
```

P<65533><65533>klad

```
*FIXME:uk<65533>zka pou<65533>it<65533>
```

P<65533><65533>buzn<65533> instrukce

JSR, RTS, JMP, BRA, Bcc

Bcc

* *$Header: /home/radek/cvs/forth-book/db-mc68k-inst/Bcc,v 1.5 2003/12/28 18:21:56 radek Exp $*

Jméno

`Bcc` — Branch Conditionally

Program Control Operation

Přehled

```
Bcc label

BCC label
BCS label
BEQ label
BGE label
BGT label
BHI label
BLE label
BLS label
BLT label
BMI label
BNE label
BPL label
BVC label
BVS label
```

Popis

Podle podm<65533>nky instrukce pokra<65533>uje dal<65533><65533> instrukc<65533>, nebo vykon<65533> skok / p<65533>evede <65533><65533>zen<65533> programu na adresu danou aktu<65533>ln<65533> adresou a uveden<65533>m posunut<65533>m.

```
If Condition True
        PC + dn ⟶ PC
```

Tabulka 1. Podm<65533>nkov<65533> k<65533>dy *:[2:3:1:7]

Mnemonic	Podm<65533>nka	K<65533>d	Test
T* / BRA	True	0000	1

Mnemonic	Podm<65533>nka	K<65533>d	Test
F* / BSR	False	0001	0
HI	High	0010	(not C) and (not Z)
LS	Low or Same	0011	C or Z
CC(HI)	Carry Clear	0100	C
CS(LO)	Carry Set	0101	C
NE	Not Equal	0110	Z
EQ	Equal	0111	Z
VC	Overflow Clear	1000	V
VS	Overflow Set	1001	V
PL	Plus	1010	N
MI	Minus	1011	N
GE	Grerat or Equal	1100	(N and V) or ((not N) and (not V))
LT	Less Than	1101	(N and (not V)) or ((not N) and V)
GT	Greater Than	1110	(N and V and (not Z)) or ((not N) and (not V) and (not Z))
LE	Less or Equal	1111	Z or N and (not V) or (not N) and V

Form<65533>t instrukce

```
 15  14  13  12  11  10   9   8   7   6   5   4   3   2   1   0
+---+---+---+---+---------------+-------------------------------+
| 0 | 1 | 1 | 0 |   condition   |      8-bit displacement       |
+---+---+---+---+---------------+-------------------------------+
|        16-bit displacement if 8-bit displacement = $00        |
+---------------------------------------------------------------+
|        32-bit displacement if 8-bit displacement = $00        |
+---------------------------------------------------------------+
```

P<65533><65533>klad

```
*FIXME:uk<65533>zka pou<65533>it<65533>
```

P<65533><65533>buzn<65533> instrukce

JMP, BRA, JSR, BSR, RTS

DB

* *$Header: /home/radek/cvs/forth-book/db-mc68k-inst/DB,v 1.1 2003/12/28 18:21:56 radek Exp $*

Jméno

`DB` — pseudonistrukce pro definov<65533>n<65533> bytu nebo bytov<65533> posloupnosti

Pseudoinstrukce

Přehled

```
DB arg[,arg]
```

Popis

*FIXME:Popis instrukce

P<65533><65533>klad

```
*FIXME:uk<65533>zka pou<65533>it<65533>
```

P<65533><65533>buzn<65533> instrukce

DW

DW

* *$Header: /home/radek/cvs/forth-book/db-mc68k-inst/DW,v 1.1 2003/12/28 18:21:56 radek Exp $*

Jméno

`DW` — pseudonistrukce pro definov<65533>n<65533> slova nebo posloupnosti slov

Pseudoinstrukce

Přehled

```
DW arg[,arg]
```

Popis

*FIXME:Popis instrukce

P<65533><65533>klad

```
*FIXME:uk<65533>zka pou<65533>it<65533>
```

P<65533><65533>buzn<65533> instrukce

DB

EXT

* *$Header: /home/radek/cvs/forth-book/db-mc68k-inst/EXT,v 1.2 2004/04/27 20:54:34 radek Exp $*

Jméno

EXT — roz<65533><65533><65533>en<65533> znam<65533>nka

FIXME: skupina instrukc<65533>

Přehled

EXT FIXME:*arg*

Popis

FIXME: Popis instrukce

P<65533><65533>klad

FIXME:uk<65533>zka pou<65533>it<65533>

P<65533><65533>buzn<65533> instrukce

FIXME:

JMP

* *$Header: /home/radek/cvs/forth-book/db-mc68k-inst/JMP,v 1.4 2003/12/28 18:21:56 radek Exp $*

Jméno

JMP — Jump

skupina instrukc<65533>

Přehled

```
JMP ea
```

Popis

*FIXME:Popis instrukce

```
ea ⟶ PC
```

P<65533><65533>klad

```
*FIXME:uk<65533>zka pou<65533>it<65533>
```

P<65533><65533>buzn<65533> instrukce

Bcc, BRA, BSR, JSR

JSR

* *$Header: /home/radek/cvs/forth-book/db-mc68k-inst/JSR,v 1.4 2003/12/28 18:21:56 radek Exp $*

Jméno

JSR — Jump to Subroutine

skupina instrukc<65533>

Přehled

```
JSR ea
```

Popis

*FIXME:

```
SP - 4 ⟶ SP; PC ⟶ (SP); ea ⟶ PC
```

P<65533><65533>klad

```
*FIXME:uk<65533>zka pou<65533>it<65533>
```

P<65533><65533>buzn<65533> instrukce

Bcc, BRA, BSR, JMP, JSR, RTS.

LEA

* *$Header: /home/radek/cvs/forth-book/db-mc68k-inst/LEA,v 1.4 2003/12/28 18:21:57 radek Exp $*

Jméno

LEA — Load Effective Address

Data Movement Operation

Přehled

LEA *ea, An*

Popis

Vypo<65533>te efektivn<65533> adresu a tu ulo<65533><65533> v adresov<65533>m registru An.

P<65533><65533>klad pou<65533>it<65533>

```
LEA     12(A7), A7
LEA     2046(A0), SP
```

P<65533><65533>buzn<65533> instrukce

EXG, LINK, MOVE, MOVEM, MOVEQ, PEA, UNLK

LINK

* *$Header: /home/radek/cvs/forth-book/db-mc68k-inst/LINK,v 1.4 2003/12/28 18:21:57 radek Exp $*

Jméno

`LINK` — Link and Allocate

Data Movement

Přehled

```
LINK An, #disp
```

Popis

Ulo<65533><65533> obsah specifikovan<65533>ho adresn<65533>ho registru do z<65533>sobn<65533>ku. Potom na<65533>te aktualizovan<65533> ukazatel z<65533>sobn<65533>ku do adresn<65533>ho registru. Nakonec p<65533>i<65533>te hodnotu posunut<65533> k ukazateli z<65533>sobn<65533>ku.

Specifikov<65533>n<65533>m z<65533>porn<65533>ho posunut<65533> je na z<65533>sobn<65533>ku vyhra<65533>eno *-d* bajt<65533>.

$$SP - 4 \longrightarrow SP;\ An \longrightarrow (SP);\ SP \longrightarrow An;\ SP + d_n \longrightarrow SP$$

Poznámka: Instrukce LINK a **UNLK** UNLK mohou b<65533>t pou<65533>ity / jsou ur<65533>eny k udr<65533>ov<65533>n<65533> z<65533>et<65533>zen<65533>ho seznamu lok<65533>ln<65533>ch dat a parametr<65533> na z<65533>sobn<65533>ku p<65533>i vol<65533>n<65533> vno<65533>en<65533>ch podprogram<65533>.

P<65533><65533>klad pou<65533>it<65533>

```
LINK    A6, #-12
```

P<65533><65533>buzn<65533> instrukce

UNLK, JSR, RTS, BSR

LSL, LSR

* *$Header: /home/radek/cvs/forth-book/db-mc68k-inst/LSd,v 1.3 2003/12/28 18:21:57 radek Exp $*

Jméno

`LSL, LSR` — Logical Shift

skupina instrukc<65533>

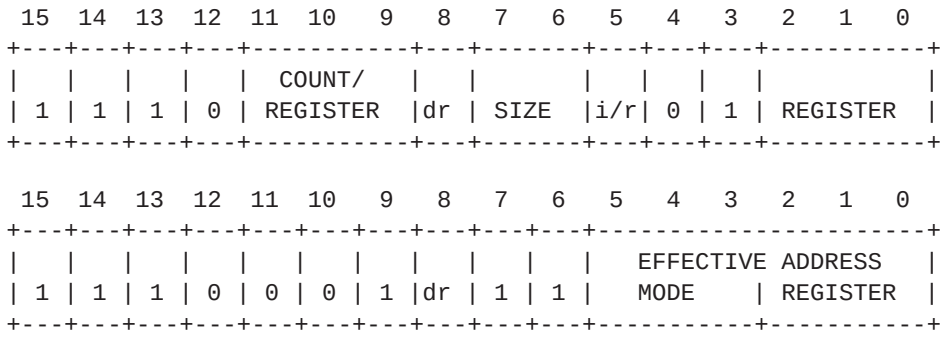
Přehled

```
LSd Dx, Dy
LSd #data, Dy
LSd ea
```

Popis

Ode<65533>te zdrojov<65533> operand od c<65533>lov<65533>ho a v<65533>sledek ulo<65533><65533> ve c<65533>li.

Form<65533>t instrukce



P<65533><65533>klad

```
*FIXME:uk<65533>zka pou<65533>it<65533>
```

P<65533><65533>buzn<65533> instrukce

*FIXME:

MOVE

* *$Header: /home/radek/cvs/forth-book/db-mc68k-inst/MOVE,v 1.3 2003/12/28 18:21:57 radek Exp $*

Jméno

`MOVE` — Move Data from Source to Destination

skupina instrukc<65533>

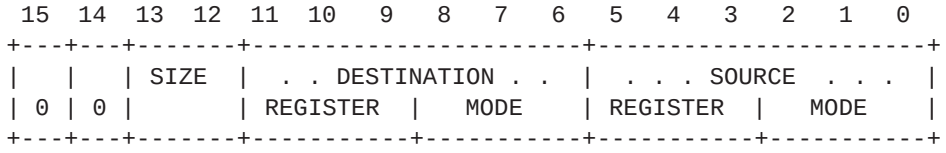
Přehled

```
MOVE ea, ea
```

Popis

P<65533>esune data ze zdroje (*source*) do c<65533>lov<65533> (*destination*) pozice, a nastav<65533> p<65533><65533>znaky podle p<65533>enesen<65533>ch dat. Velikost p<65533>en<65533>en<65533>ch dat m<65533>e b<65533>t specifikov<65533>na jako slabika (*byte*), slovo (*word*) nebo dlouh<65533> slovo (*long)*. P<65533><65533>znaky (*Condition Codes)*) jsou nastaveny takto: V a C jsou vynulov<65533>ny, N a Z jsou nastaveny podle dat, X nen<65533> ovlivn<65533>n.

Form<65533>t instrukce



P<65533><65533>klad

```
*FIXME:uk<65533>zka pou<65533>it<65533>
```

P<65533><65533>buzn<65533> instrukce

*FIXME:

MOVEM

* *$Header: /home/radek/cvs/forth-book/db-mc68k-inst/MOVEM,v 1.3 2003/12/28 18:21:57 radek Exp $*

Jméno

`MOVEM` — Move Multiple Registers

Data Movement

Přehled

```
MOVEM list ea

MOVEM ea list
```

Popis

Ulo<65533><65533> obsah specifikovan<65533>ch registr<65533> do pam<65533>ti, nebo naopak provede obnoven<65533> specifikovan<65533>ch registr<65533> z pam<65533>ti.

P<65533><65533>klad pou<65533>it<65533>

```
MOVEM.L D2-D7/A1-A6, -(A7)
MOVEM.L (A7)+, D2-D7/A1-A6
```

MOVEQ

* *$Header: /home/radek/cvs/forth-book/db-mc68k-inst/MOVEQ,v 1.3 2003/12/28 18:21:57 radek Exp $*

Jméno

`MOVEQ` — Move Quick

skupina instrukc<65533>

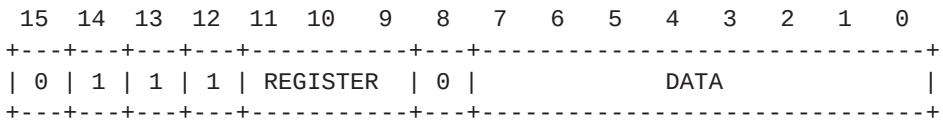
Přehled

```
MOVEQ #const, Dn
```

Popis

Rychl<65533> napln<65533>n<65533> registru Dn malou konstantou.

Form<65533>t instrukce



P<65533><65533>klad

```
*FIXME:uk<65533>zka pou<65533>it<65533>
```

P<65533><65533>buzn<65533> instrukce

*FIXME:

MULS

* *$Header: /home/radek/cvs/forth-book/db-mc68k-inst/MULS,v 1.1 2004/03/06 00:33:20 radek Exp $*

Jméno

`MULS` — n<65533>soben<65533> se znam<65533>nkem

skupina instrukc<65533>

Přehled

`Instrukce` *`arg`*

Popis

*FIXME:Popis instrukce

P<65533><65533>klad

```
*FIXME:uk<65533>zka pou<65533>it<65533>
```

P<65533><65533>buzn<65533> instrukce

*FIXME:

PEA

* *$Header: /home/radek/cvs/forth-book/db-mc68k-inst/PEA,v 1.3 2003/12/28 18:21:57 radek Exp $*

Jméno

PEA — Push Effective Address

Data Movement

Přehled

PEA ***ea***

Popis

Vypo<65533>te efektivn<65533> adresu a tu ulo<65533><65533> na z<65533>sobn<65533>k.

`SP - 4 ⟶ SP; ea ⟶ (SP)`

P<65533><65533>klad pou<65533>it<65533>

```
PEA     -12(A6)
PEA     -8(A6)
PEA     -4(A6)
```

P<65533><65533>buzn<65533> instrukce

LEA

RTS

* *$Header: /home/radek/cvs/forth-book/db-mc68k-inst/RTS,v 1.5 2003/12/28 18:21:57 radek Exp $*

Jméno

RTS — Return from Subroutine

skupina instrukc<65533>

Přehled

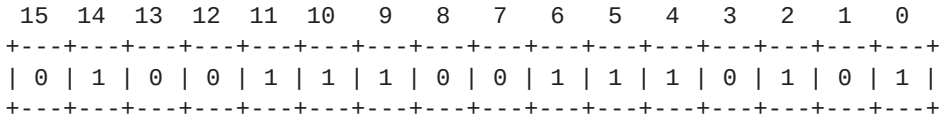
```
RTS
```

Popis

Obnov<65533> p<65533>edchoz<65533> hodnotu <65533><65533>ta<65533>e instrukc<65533> ze z<65533>sobn<65533>ku. Aktu<65533>ln<65533> hodnota <65533><65533>ta<65533>e je ztracena/zapomenuta.

(SP) ⟶ PC; SP + 4 ⟶ SP

Form<65533>t instrukce



P<65533><65533>klad

```
*FIXME:uk<65533>zka pou<65533>it<65533>
```

P<65533><65533>buzn<65533> instrukce

BSR, JSR

SUB

* *$Header: /home/radek/cvs/forth-book/db-mc68k-inst/SUB,v 1.3 2003/12/28 18:21:57 radek Exp $*

Jméno

SUB — Subtract

skupina instrukc<65533>

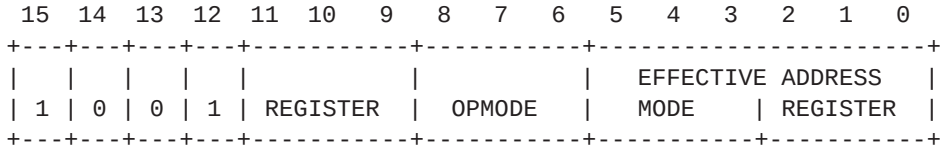
Přehled

```
SUB ea, Dn
SUB Dn, ea
```

Popis

Ode<65533>te zdrojov<65533> operand od c<65533>lov<65533>ho a v<65533>sledek ulo<65533><65533> ve c<65533>li.

Form<65533>t instrukce



P<65533><65533>klad

```
*FIXME:uk<65533>zka pou<65533>it<65533>
```

P<65533><65533>buzn<65533> instrukce

*FIXME:

SWAP

* *$Header: /home/radek/cvs/forth-book/db-mc68k-inst/SWAP,v 1.3 2003/12/28 18:21:57 radek Exp $*

Jméno

SWAP — Swap Register Halves

skupina instrukc<65533>

Přehled

```
Instrukce Dn
```

Popis

Prohod<65533> doln<65533>ch a horn<65533>ch 16 bit<65533> registru Dn.

```
Register 31 - 16 ⟵⟶ Register 15 - 0
```

P<65533><65533>klad

```
*FIXME:uk<65533>zka pou<65533>it<65533>
```

P<65533><65533>buzn<65533> instrukce

*FIXME:

TST

* *$Header: /home/radek/cvs/forth-book/db-mc68k-inst/TST,v 1.3 2003/12/28 18:21:57 radek Exp $*

Jméno

TST — Test an Operand

skupina instrukc<65533>

Přehled

TST *ea*

Popis

Porovn<65533> operand s nulou a nastav<65533> „condition codes“ podle v<65533>sledk<65533> porovn<65533>n<65533>. Velikost operandu je slabika, slovo, nebo dlouh<65533> slovo.

X	nen<65533> ovlivn<65533>n
N	je-li operand z<65533>porn<65533> tak 1, jinak 0
Z	je-li operand nulov<65533> tak 1, jinak 0
V	0
C	0

P<65533><65533>klad

```
TST.W   D0
TST.W   8(A6)
```

P<65533><65533>buzn<65533> instrukce

*FIXME:

UNLK

* *$Header: /home/radek/cvs/forth-book/db-mc68k-inst/UNLK,v 1.4 2003/12/28 18:21:57 radek Exp $*

Jméno

`UNLK` — Unlink

Data Movement

Přehled

```
UNLK An
```

Popis

Obnov<65533> ukazatel z<65533>sobn<65533>ku ze specifikovan<65533>ho adresov<65533>ho registru. Pot<65533> obnov<65533> obsah adresov<65533>ho registru z dlouh<65533>ho slova ulo<65533>en<65533>ho na z<65533>sobn<65533>ku.

```
An ⟶ SP; (SP) ⟶ An; SP + 4 ⟶ SP
```

P<65533><65533>klad pou<65533>it<65533>

```
UNLK   A6
```

P<65533><65533>buzn<65533> instrukce

LINK, JSR, RTS, BSR

Příloha B. Seznam lid<65533> jen<65533> se kolem Ruby vyskytovali <65533>i vyskytuj<65533>

* *$Header: /home/radek/cvs/forth-book/db-people/dbheader.xml,v 1.1 2002/12/18 23:25:07 radek Exp $*
* *Tajn<65533>, nezve<65533>ej<65533>ovat*

Leo Brodie *$RCSfile: Leo_Brodie,v $ $Date: 2002/12/18 23:25:07 $*

FIXME: P<65533>r slov k <65533>lov<65533>ku. Autor dokumentu „Starting Forth“

Adresa: FIXME:<bla@firma.com>

FIXME: Autor programu: (xref)

Příloha C. Různé příklady

* *rcsinfo="$Header: /home/radek/cvs/forth-book/ap-ruzne.xml,v 1.3 2005/10/20 05:33:42 radek Exp $"*

epigraf

Text kapitoly

C.1. Různe způsoby psaní komentářů

Příklad C-1. Různé způsoby psaní komentářů

Podle příspěvku „Comment delimiter consensus“ od Brenda Paysana (<bernd.paysan@gmx.de>) ze dne 2000-08-20 zaslaného do konference comp.lang.forth (news://comp.lang.forth).

```
: ?refill source nip >in @ = IF refill 0= ELSE false THEN ;
: parse* ( char &rarr; eol? match?)
    parse + source + over <> swap 1- c@ '* = ;
: parse*del ( char &rarr; ) >r
    BEGIN r@ parse* and 0= WHILE ?refill UNTIL THEN rdrop ;
: (* ') parse*del ;
: /* '/ parse*del ;
: \* '\ parse*del ;
```