PREKOMPILÁTOR JAZYKA FORTRAN PRO MATICOVÉ OPERACE

V. Šebesta, S. Kočková

1. Úvod

Maticový procesor EC 2345 je specializovaný přídavný procesor, který ve spojení se základním počítačem slouží ke zvýšení výkonnosti celého výpočetního systému. Přívlastek "maticový" vyjadřuje specializaci procesoru na numerické operace s maticemi a vektory nikoli typ architektury. Vyšší výkonnosti při provádění maticových operací je dosaženo specializací procesoru a využitím paralelismu ve formě zřetězení elementárních funkčních bloků (pipelining).

Použití maticového procesoru EC 2345 vyžaduje speciální znalosti programování (z kanálových instrukcí se sestavují kanálové programy). K zpřístupnění maticového procesoru širší programátorské veřejnosti slouží:

1. knihovny XMP a LINEQAP
2. prekompilátor maticových operací pro FORTRAN IV a FORTRAN 77.

Knihovna XMP obsahuje podprogramy realizující vybrané maticové a vektorové operace s využitím maticového procesoru EC 2345. Jedná se o násobení matice konstantou, vektorem zleva, zprava, maticí, sčítání - odčítání matic po prvcích, normu matice, některé operace s diagonálou matice, operace s pásovými maticemi, výpočet autokovariačních koeficientů a další. Podprogramy jsou napsány v jazyce FORTRAN (některé v jazyce ASSEMBLER), mají jednotné názvy Xčč (č - číslice 0-9) a vyvolávají se standardním příkazem

```
CALL Xčč (seznam parametrů)
```

Uživatelský popis podprogramů knihovny XMP včetně analýzy jejich výkonnosti je uveden v |3|.

Knihovna LINEQAP obsahuje podprogramy pro řešení soustav lineárních rovnic a inversi matice. Bližší informace viz |7|.

Prekompilátor jazyků FORTRAN IV a FORTRAN 77 pro maticové operace nabízí uživateli kromě použití maticového procesoru ještě zjednodušení zápisu programu při operacích s maticemi a s vektory dat.

2. Příkazy zdrojového programu

Prekompilátor čte zdrojový program, vyhledává příkazy, které se týkají maticových operací, a na základě jejich analýzy generuje výstupní program, který je vstupním programem pro kompilátory FORTRAN IV a FORTRAN 77.

Zdrojový program je program v jazyce FORTRAN IV nebo FORTRAN 77 obohacený o maticové příkazy:

1. specifikační
2. deklarační
3. přiřazovací

2.1. Specifikační příkaz

- má tvar

```
IMPLICIT ARRAY  a
```

kde a je jedno z písmen A, B, ... , Z.

- specifikuje počáteční písmeno identifikátorů matic a vektorů, které mohou vystupovat v maticových příkazech,
- musí být v programu umístěn jako první (v podprogramu hned za jménem subroutiny, předcházet smějí pouze komentáře), může být v každé programové jednotce pouze jediný.

2.2. Deklarační příkaz

- má tvar

  DIMENSION $a_1(k_1)$ |, $a_2(k_2)$ ... | ,

  kde a_i je identifikátor pole
  k_i je seznam mezí indexů, maximální počet mezí je roven dvěma,
- má standardní význam,
- všechny vektory a matice, které vystupují v maticových přiřazovacích příkazech, musejí být deklarovány tímto příkazem, mohou být nejvýše dvoudimenzionální.

2.3. Maticový přiřazovací příkaz

Maticový přiřazovací příkaz má tvar

a = b ,

kde a je matice
b je maticový aritmetický výraz.

Maticí se rozumí jedno nebo dvoudimenzionální pole. Maticový aritmetický výraz je výraz, vytvořený z prvotních maticových výrazů pomocí operátorů

* * (-1)	inverze
* * (T)	transpozice
*	násobení
+	sčítání
-	odečítání

- prvotní maticový výraz je +A, -A, kde A je identifikátor matice.

Při vytváření maticových přiřazovacích příkazů platí běžná algebraická pravidla, navíc je nutné dodržovat několik dalších pravidel:

- přiřazovat lze jen matice shodných rozměrů,
- všechny vystupující matice musejí:
  - být typu REAL*4, deklarované příkazem DIMENSION
  - musí mít identifikátor se stejným počátečním písmenem udaným příkazem IMPLICIT ARRAY
- maticový přiřazovací příkaz musí být na samostatném štítku:

dobře	špatně

IF (podmínka) THEN
(maticový přiřaz.příkaz)
ELSE
(maticový přiřaz.příkaz)
ENDIF	

IF (podmínka) GO TO 10 ⋮ 10 AA=AB+AC	IF (podmínka) AA=AB+AC

Maticový aritmetický výraz na pravé straně přiřazovacího příkazu

- má jednotlivé operandy navzájem odděleny maticovými operátory

např.	dobře	špatně
	-AB * A	-ABA (identif.jiné matice)
	A * (-AB)	A * -AB

- počet operátorů v jednom maticovém aritmetickém výrazu je omezen na 30,
- pro jednotlivé maticové operace platí:
  - invertovat lze maticí čtvercovou, regulární
  - násobit lze jen matice typu (M,N), (N,K)
  - sčítat a odečítat lze matice shodných rozměrů
  - je možno násobit matici konstantou typu REAL*4.

Příklad

Jsou-li uvedeny příkazy

```
IMPLICIT ARRAY A
DIMENSION AA(20,20),AB(20,20),AC(20),AD(1,20),A(20),B(20),
```

pak následující maticové přiřazovací příkazy jsou:

dobře	špatně
A = AC	AC = AD
AA = AB	A = B
AC = AD**(T)	AA = AD*AB
AA = (AB**(-1)+AC*AD)* const	

3. Popis prekompilátoru

3.1. Analýza zdrojového programu

Prekompilátor čte příkazy zdrojového programu a vyhledává příkazy:

```
IMPLICIT ARRAY,
DIMENSION ,
```

Maticový přiřazovací příkaz,

```
END ,
```

zpracovává je a generuje výstupní program pro kompilátory FORTRAN IV a FORTRAN 77.

Příkaz IMPLICIT ARRAY slouží k identifikaci matic, které budou vystupovat v maticových přiřazovacích příkazech.

Příkaz DIMENSION poskytuje prekompilátoru, stejně jako kompilátoru FORTRAN IV a FORTRAN 77 informaci o počtu dimenzí deklarovaného pole a o skutečných hodnotách jednotlivých indexů. Na základě tohoto příkazu jsou pro specifikovaná pole vytvořeny prekompilátorem tabulky vnějších matic.

Maticový přiřazovací příkaz

Při zpracování maticového přiřazovacího příkazu je třeba nejprve vyhodnotit maticový aritmetický výraz na pravé straně přiřazovacího příkazu a potom provést vlastní přiřazení. Každý maticový aritmetický výraz se při vyhodnocování rozpadá do posloupnosti elementárních maticových operací:

A. Přiřazení $A = \pm B$

Pro prvky $a_{i,j}$ matice A(M,N)

platí $a_{i,j} = \pm b_{i,j}$, $i = 1,\ldots,M$, $j = 1,\ldots,N$.

B. Násobení konstantkou $A = k * B$

Pro prvky $a_{i,j}$ matice A(M,N)

platí $a_{i,j} = k \cdot b_{i,j}$, $i = 1,\ldots,M$, $j = 1,\ldots,N$.

C. Sčítání, resp. odčítání $A = B \pm C$

Pro prvky $a_{i,j}$ matice A(M,N)

platí $a_{i,j} = b_{i,j} \pm c_{i,j}$, $i = 1,\ldots,M$, $j = 1,\ldots,N$.

D. Násobení $A = B * C$

Pro prvky $a_{i,j}$ matice A(M,N)

platí $a_{i,j} = \sum_{k} b_{i,k} \cdot c_{k,j}$, $i = 1,\ldots,M$, $j = 1,\ldots,N$.

E. Transpozice $A = B ** (T)$

Pro prvky $a_{i,j}$ matice A(M,N)

platí $a_{i,j} = b_{j,i}$, $i = 1,\ldots,M$, $j = 1,\ldots,N$.

F. Inverze $A = B ** (-1)$

Výsledná matice $A = B^{-1}$ je matice inverzní v obvyklém slova smyslu, t.j.

$B * B^{-1} = I$.

Pořadí,ve kterém se operace vyhodnocují, je dáno běžnými algebraickými pravidly s respektováním priority:

nejvyšší 1. inverze, transpozice
2. násobení
3. sčítání, odčítání.

Příkazy stejné priority jsou prováděny zleva doprava, konvence používání závorek je stejná jako v jazyce FORTRAN.

Pro ukládání mezivýsledků maticového aritmetického výrazu, t.j. výsledků elementárních maticových operací, je vymezeno pomocné pole s identifikátorem,

který je vytvořen zdvojením písmene specifikovaného příkazem IMPLICIT ARRAY. Stejným identifikátorem nesmí být tedy označena jiná proměnná v téže programové jednotce. Do pomocného pole jsou postupně ukádány matice, které vznikají jako mezivýsledky maticových elementárních operací a pracovní matice nutné pro operaci inverze - tzv. vnitřní matice. Pomocné pole je rozděleno pomocí ukazovátek s identifikátory NMLKJI(i), která v pomocném poli určují první slovo pro uložení i-té vnitřní matice. Přiřazování vnitřních matic je dynamické, totéž místo pomocného pole může být využito několikrát.

Příkaz END poskytuje prekompilátoru stejně jako kompilátoru informaci o ukončení programové jednotky. Prekompilátor vynuluje všechny čítače a buď zpracovává další programovou jednotku nebo zpracování ukončí.

3.2. Příklad zpracování maticového přiřazovacího příkazu

Předpokládejme následující posloupnost příkazu:

```
IMPLICIT ARRAY X
DIMENSION XB2(M,M),XV(M,M),XABC(M,M),XA3(M,M)
:
XB2=((XV*ABC**(-1))+XA3**(T))**(-1)
:
```

Maticový aritmetický výraz bude vyhodnocen provedením pěti elementárních maticových operací. K uložení mezivýsledků je určeno pomocné pole s identifikátorem XX.

1) Pro výsledek první elementární operace XABC**(-1) bude vyhrazena první vnitřní matice XPOM1, uložená od počátku pomocného pole XX(NMLKJI(1)), kde NMLKJI(1)=1.
   XPOM1=XABC**(-1)
2) Výsledek druhé elementární operace bude uložen jako druhá vnitřní matice XPOM2 v poli XX od indexu NMLKJI(2)=NMLKJI(1)+M*M,
   XPOM2=XV*XPOM1,
   zároveň první vnitřní matice XPOM1 bude uvolněna k dalšímu použití.
3) Výsledek třetí elementární operace bude uložen opět jako první vnitřní matice XPOM1
   XPOM1=XA3**(T).
4) Výsledek čtvrté elementární operace bude uložen jako třetí vnitřní matice XPOM3 v poli XX od indexu
   NMLKJI(3)=NMLKJI(2)+M*M
   XPOM3=XPOM2+XPOM1,
   místa v poli XX vyhrazená maticím XPOM1 a XPOM2 budou uvolněna.
5) Výsledek poslední elementární operace, tedy celého maticového výrazu, bude uložen do předepsaného pole
   XB2=XPOM3**(-1).

Pro operaci inverze je navíc vyhrazeno vždy v poli XX pracovní pole rozměru M*M+2, které je po provedení inverze okamžitě uvolněno.

Schematicky

XB2=((XV*XABC**(-1)) + XA3**(T))**(-1)

XPOM1 uložená od XX((MNLKJI(1))

XPOM2 uložená od XX(MNLKJI(2)) - uvolní se XPOM1

XPOM1 uložená od XX(MNLKJI(1))

XPOM3 uložená od XX(MNLKJI(3)) - uvolní se XPOM2 a XPOM1

XB2

3.3. Subparametry pro průběh prekompilace a výpočtu

Při volání prekompilátoru je možno ovlivnit průběh prekompilace a výpočtu volbou subparametrů parametru PARM:

Parametr PARM příkazu EXEC má tvar:

PARM= [SOURCE | NOSOURCE] [,EC 2345 | ,NOEC2345] [,SIZE1=... | ,10000] [,SIZE2=... | ,500]

SOURCE — Hodnota SOURCE způsobí opis zdrojového programu do souboru
NOSOURCE — FT16F001

EC2345 — NOEC2345 způsobí zpracování maticových aritmetických operací centrálním procesorem, zatímco standardní hodnota EC2345 znamená požadavek na využití maticového procesoru
NOEC2345

SIZE1=... — Hodnota SIZE1 udává velikost pomocného pole pro ukládání mezivýsledků
SIZE1=10000 — a pracovních polí

SIZE2=... — Hodnota SIZE2 udává velikost pole CCWA pro uložení kanálového programu
SIZE2=500

Standardní hodnoty jsou podtrženy.

Velikost SIZE1 závisí na nejsložitějším maticovém aritmetickém výrazu, který se v programu vyskytuje, resp. na jeho náročnosti na paměť (viz 3.1). Hodnota SIZE2 musí být větší než největší řádkový index všech matic, které mají být invertovány či transponovány. Navíc, všechny programy, které mají být společně sestaveny (linkovány), musejí být prekompilátorem přeloženy se stejnou hodnotou subparametru SIZE2. Při nedostatečných velikostech SIZE1, SIZE2 jsou při výpočtu podána příslušná chybová hlášení (viz 3.5).

Použití maticového procesoru může značně zrychlit provádění aritmetických operací s maticemi a vektory dat "velkých" rozměrů, naopak pro matice a vektory "malých" rozměrů nejen že není výhodné, ale může vést i ke zpomalení výpočtu. Proto byla pro každou maticovou operaci určena dolní mez rozměrů pole, pro kterou je použití maticového procesoru výhodné. Jestliže uživatel hodnotou subparametru (EC2345) zvolí zpracování maticových aritmetických operací maticovým procesorem, potom operace s maticemi o rozměrech rovných alespoň této mezi jsou zadány k výpočtu maticovému procesoru, aritmetické operace s maticemi menších rozměrů jsou zpracovány centrálním procesorem. Při udání hodnoty NOEC2345 jsou všechny maticové aritmetické operace programu prováděny centrálním procesorem.

3.4. Generovaný program

Do vstupního programu budou generovány příkazy pro deklaraci oblastí:

CCWA typu REAL*8 - pro uložení kanálového programu. Standardní velikost 500 dvouslov může být změněna subparametrem SIZE2

OCWA typu REAL*8 - pro pole operandů kanálového programu. Jeho délka je vždy třikrát větší než pole CCWA (instrukce kanálového programu má zpravidla 3 operandy)

Pomocné pole typu REAL*4 - pro ukládání mezivýsledků standardní délky 10 000 slov (resp. SIZE1). Jeho identifikátor tvoří zdvojené písmeno v příkaze IMPLICIT ARRAY

NMLKJI typu INTEGER - pro funkci ukazovátka v pomocném poli,
ve tvaru
```
COMMON CCWA(...),OCWA(...),pomocné pole(...)
DIMENSION NMLKJI(30).
```

Dále je generován příkaz

DATA NMLKJI(1)/1/ - první vnitřní matice je vždy uložena do počátku pomocného pole.

Při výskytu maticového přiřazovacího příkazu jsou generovány příkazy pro vymezení vnitřních matic, výpočet a kontrolu jejich dimenzí tvaru:

```
NMLKJI(i)=NMLKJI(i-1)+M*N
IF(NMLKJI(i).GT.10 000) STOP 20
```

a příkazy volání subroutin

```
CALL XF1(...) - pro přiřazení matic
CALL XF2(...) - pro násobení matice konstantou
CALL XF3(...) - pro sčítání (odčítání) matic
CALL XF4(...) - pro násobení matic
CALL XF5(...) - pro transpozici matice
CALL XF6(...) - pro inverzi matice.
```

Příklad zpracování zdrojového programu prekompilátorem

Mějme následující program zadaný prekompilátoru se standardními subparametry:

```
      IMPLICIT ARRAY X
      DIMENSION XABC(20,20),XA1(20),XA2(1,20),XA3(20,20),XV(20,20),XB2(20,20)
      K=20
      DO 11 I=1,K
      DO 10 J=I,K
      XABC(I,J)=1.0
      XA3(I,J)=...
   10 CONTINUE
      XA1(I)=...
      XA2(1,I)=...
   11 CONTINUE
      CALL NASOB(XA1,XA2,XV,K)
      XB2=((XV*XABC**(-1))+XA3**(T))**(-1)
      :
      END
```

```
      SUBROUTINE NASOB(AX,AB,AC,K)
      IMPLICIT ARRAY A
      DIMENSION AX(K),AB(1,K),AC(K,K)
      AC=AX*AB
      RETURN
      END
```

Generovaný program

```
      REAL*8 CCWA,OCWA
      COMMON CCWA(500),OCWA(1500),XX(10 000)
      DIMENSION NMLKJI(30)
      DATA NMLKJI(1)/1/
      DIMENSION XABC(20,20),XA1(20),XA21,20),XA3(20,20),XV(20,20),XB2(20,20)
      K=20
      DO 11 I=1,K
      DO 10 J=1,K
      :
   11 CONTINUE
      CALL NASOB(XA1,XA2,XV,K)
      NMLKJI(2)=NMLKJI(1)+20*20
      IF(NMLKJI(2).GT.10 000) STOP 20
      NMLKJI(3)=NMLKJI(2)+20*20+2*20
      IF(NMLKJI(3).GT.10 000) STOP 20
      NMLKJI(3)=NMLKJI(2)+20*20
      IF(NMLKJI(3).GT.10 000) STOP 20
      NMLKJI(4)=NMLKJI(3)+20*20
      IF(NMLKJI(4).GT.10 000) STOP 20
      NMLKJI(5)=NMLKJI(4)+20*20+2*20
      IF(NMLKJI(5).GT.10 000) STOP 20
      CALL XF6(XX(NMLKJI(1)),XABC,XX(NMLKJI(2)),20,XX(NMLKJI(2)+20*20),
     *      XX(NMLKJI(2)+20*20+20))
      CALL XF4(XX(NMLKJI(2)),XV,XX(NMLKJI(1)),20,20,20)
      CALL XF5(XX(NMLKJI(1)),XA3,20,20)
      CALL XF3(XX(NMLKJI(3)),XX(NMLKJI(2)),XX(NMLKJI(1)),20,20,1)
      CALL XF6(XB2,XX(NMLKJI(3)),XX(NMLKJI(4)),20,XX(NMLKJI(4)+20*20),
     *      XX(NMLKJI(4)+20*20+20))
      :

        SUBROUTINE NASOB(AX,AB,AC,K)
      REAL*8 CCWA,OCWA
      COMMON CCWA(500),OCWA(1500),AA(10 000)
      DIMENSION NMLKJI(30)
      DATA NMLKJI(1)/1/
      DIMENSION AX(K),AB(1,K),AC(K,K)
      CALL XF4(AC,AX,AB,K,K,1)
      RETURN
      END
```

3.5. Syntaktické kontroly a chybová hlášení prekompilátoru

Syntaxe aritmetických maticových výrazů je kompilatorem kontrolována a při nalezení chyby jsou generována následující chybová hlášení:

-NO IMPLICIT ARRAY DECLARATION
-MULTIPLE IMPLICIT ARRAY DECLARATION
-MORE THEN 19 ADDITIONAL CARDS
-NO INPUT FILE
-NO END CARD
-UNEXPECTED CHARACTER
-NO LIMITS ON DIMENSION CARD
-RIGHT PARENTHESIS MISSING
-UNEXPECTED CHARACTER IN THE FIELD OF LETTERS AND NUMBERS
-MORE THEN 6 CHARACTERS IN THE NAME
-TOO COMPLICATED EXPRESSION
-WRONG USE OF =
-WRONG OPERATOR BEHIND =
-WRONG OPERATOR BEHIND (
-WRONG OPERATOR BEHIND)
-WRONG PARENTHESIS
-WRONG OPERATOR BEHIND + OR - OR *
-WRONG OPERATOR BEHIND **(T) OR **(-1)
-UNEXPECTED CHARACTER BEHIND NAME
-MATRIX NAME WAS NOT DECLARED
-MISSING END OF STATEMENT
-THE NUMBERS OF LEFT AND RIGHT PARENTHESIS ARE DIFFERENT
-MISSING =
-NO MATRIX IN MULTIPLICATION
-ATTEMPT TO ADD, SUBTRACT OR TRANSMIT DIFFERENT TYPES OF MATRICES
-THE DIMENSIONS OF MATRICES ARE WRONG XXXXXX YYYYYY
-MORE THEN 30 MATRICES IN THE ARRAY XX
-MORE THEN 30 MATRIX OPERATIONS IN THE STATEMENT
-MORE THEN 30 CONSTANTS IN THE STATEMENT
-FIELD OF PROGRAM PARAMETERS ON EXEC CARD IS TOO LONG
-PARAMETER SIZE2 TOO SMALL

Je-li ve fázi výpočtu předčasně ukončena úloha s hodnotou uživatelského návratového kódu 20, znamená to, že pomocí parametru SIZE1 (standardně 10 000) bylo rezervováno nedostatečně velké pole pro výsledky jednotlivých elementárních maticových operací.

4. Procedura FOMA86

Při zpracování zdrojového programu je třeba vyvolat prekompilátor před vlastním kompilátorem.
K provedení všech čtyř kroků -
- prekompilace (programem FOMA86)
- kompilace (programem FORTG či jiným kompilátorem) jazyka FORTRAN IV nebo FORTRAN 77
- sestavení (spojovacím programem IEWL)
- výpočtu

byla sestavena katalogizovaná procedura FOMA86.
Symbolickým parametrem procedury je jméno fortranského kompilátoru, pro FORTRAN 77 je to PG = FORTVS, pro FORTRAN IV standardní hodnota PG = IGIFORT.

Procedura FOMA86

```
//FOMA86    PROC  PG=IGIFORT
//FORT      EXEC  PGM=FOMA86,REGION=160K
//STEPLIB   DD    DSN=CVS10.KOSE,DISP=SHR
//FT16F001  DD    SYSOUT=A
//FT11F001  DD    DSN=&MEZ1,UNIT=DISK,...
//FT05F001  DD    DDNAME=SYSIN
//FORT1     EXEC  PGM=*.PG,COND=(4,LT,FORT)
//SYSUT1    DD    ...
//SYSPRINT  DD    SYSOUT=A
//SYSPUNCH  DD    SYSOUT=B
//SYSLIN    DD    DSN=&LOADSET,UNIT=DISK,...
//SYSIN     DD    DSN=*.MEZ1,DISP=(OLD,DELETE)
//LKED      EXEC  PGM=IEWL,COND=((4,LT,FORT),(4,LT,FORT1))
//SYSLIB    DD    DSN=SYS1.FORTLIB,DISP=SHR
//          DD    DSN=SYS1.APAMLIB,DISP=SHR
//          DD    DSN=CVS10.KOSE,DISP=SHR
//SYSLMOD   DD    DSN=*.GOSET(MAIN),...
//SYSPRINT  DD    SYSOUT=A
//SYSLIN    DD    DSN=&LOADSET,...
//GO        EXEC  PGM=*.LKED.SYSLMOD,COND=(...)
//FT05F001  DD    ...
//FT06F001  DD    ...
//FT07F001  DD    ...
//ARRAY     DD    UNIT=600
//APSPMSG   DD    SYSOUT=A
//FOMA86    PEND
```

Volání procedury

```
//JOBLIB DD DSN=CVS10.KOSE,DISP=SHR
//S1     EXEC   FOMA86[,PG=...] [,PARM.FORT='...']
//                     [,REGION.GO=     ]
//FORT.SYSIN DD *
        :
    program
        :
/*
//
```

5. Závěr

Prekompilátor maticových operací slouží k usnadnění práce s poli dat při sestavení zdrojového programu a zprostředkovává použití maticového procesoru pro efektivní zpracování maticových operací. Katalizovaná procedura FOMA86 umožňuje snadné použití prekompilátoru centrálního procesoru, zvláště při práci s vektory a maticemi větších rozměrů.

6. Litaratura

[1] Král, J.: Umění psát kompilátor. Sborník semináře MOP86, MFF UK, Praha 1985.

[2] Müller, K., Vogel, J.: Gererování cílového programu. Sborník semináře MOP85, MFF UK, Praha 1985.

[3] Semerdžjan, M.A., Nalbrudjan, Ž.S.: Matričnyj processor EC2345. Moskva, Financy i statistika, 1984.

[4] Strakoš, Z.: Použití maticového procesoru EC2345 k řešení vědecko-technických úloh. Práce ke kandidátskému minimu, Praha 1984.

[5] Strakoš, Z.: Nová metoda hodnocení výkonnosti paralelních algoritmů. Kandidátská disertační práce SVT ČSAV, Praha 1985.

[6] Strakoš, Z., Kočková, S.: XMP - knihovna podprogramů pro maticové a vektorové operace s použitím maticového procesoru EC2345, VZ-219, SVT ČSAV, Praha 1986.

[7] Strakoš, Z.: Aplikační programové vybavení maticového procesoru EC2345, Práce SVT ČSAV č. 20, SVT ČSAV, Praha 1986.