СООБЩЕНИЯ
ОБЪЕДИНЕННОГО
ИНСТИТУТА
ЯДЕРНЫХ
ИССЛЕДОВАНИЙ
ДУБНА

11 - 10449

П.Нойберт

ФОРМАЛЬНОЕ ОПРЕДЕЛЕНИЕ ЯЗЫКА КАМАК В МАТЕМАТИЧЕСКОМ ОБЕСПЕЧЕНИИ ЭВМ HP 2116C

1977

11 - 10449

П.Нойберт

ФОРМАЛЬНОЕ ОПРЕДЕЛЕНИЕ ЯЗЫКА КАМАК В МАТЕМАТИЧЕСКОМ ОБЕСПЕЧЕНИИ ЭВМ HP 2116C

Нойберт П. 11 - 10449

Формальное определение языка КАМАК в математическом обеспечении ЭВМ HP 2116C

Реализован язык КАМАК на базе ЭВМ HP 2116C и операционной системы ДОС. Использованы операторы в программах для обслуживания оборудования в стандарте КАМАК.

Операторы языка КАМАК расширяют язык FORTRAN. Этот принцип объединяет средства для программного обслуживания в стандарте КАМАК, а также для обработки входных или выходных данных. Компилятор транслирует полный объем языка, а библиотека подпрограмм, выполняющих операторы языка КАМАК, реализована для контроллера каркаса КК0Ø4.

Работа выполнена в Лаборатории ядерных проблем ОИЯИ.

Сообщение Объединенного института ядерных исследований. Дубна 1977

1. ВВЕДЕНИЕ

Описывается реализация предложенного комитетом ESONE [1] языка КАМАК на базе ЭВМ HP 2116C и операционной системы DOS или DOS III [2,3]. Операторы языка КАМАК расширяют язык FORTRAN или FORTRAN IV. Этот принцип объединяет средства для программного обслуживания оборудования в стандарте КАМАК, а также для обработки входных или выходных данных. Компилятор транслирует полный объем языка, а библиотека подпрограмм, выполняющих операторы языка КАМАК, реализована для контроллера каркаса КК0Ø4 [4].

2. ПРИНЦИПЫ ПРЕДСТАВЛЕНИЯ ЯЗЫКА

Синтаксическая нотация используется для описания структуры языка и его элементов. С ее помощью программист узнает, как он должен писать свою программу. Для синтаксической нотации продукционных правил используется, аналогично [1], модифицированная расширенная форма метаязыка Бэкуса-Наура. Ключевые слова языка КАМАК записываются большими буквами, промежуточные символы - маленькими буквами и тире. Описание семантики сделано в виде предложений.

Для представления правил используются следующие метаязыковые символы:

: : = - составной символ может читаться: "определяется как",

/ - "или",

{ } – означает группу, состоящую более чем из одного элемента, в качестве синтаксической единицы,

[ ] – содержимое этих скобок на выбор,

... – возможное повторение прежней синтаксической единицы,

CR–LF – символ конца оператора, обычно возврат каретки и перевод строки.

3. ОСНОВНЫЕ СИМВОЛЫ

Синтаксис:

integer ::= digit ...

identifier ::= letter [alphanumeric–character ...]

camac–identifier ::= spec–letter [alphanumeric–character ...]

character ::= syntax–character / non–syntax–character

syntax–character ::= alphanumeric–character / delimiter

non–syntax–character ::= ! / ? / < / > / [/] / ´´ / # / $ / % / & / . / ; / ^ / ' / – / @ /

alphanumeric–character ::= letter/digit

delimiter ::= : / , / (/) / ␣ / = / ´ / operator/CR–LF

letter ::= A/B/C/D/E/F/G/H/spec–letter/O/P/Q/R/S/T/U/V/W/X/Y/Z

spec–letter ::= I/J/K/L/M/N

digit ::= octal–digit/8/9

octal–digit ::= Ø/1/2/3/4/5/6/7

operator ::= adop/mulop

adop ::= + / –

mulop ::= * / /

Семантика:

1. Диапазон целых чисел $-32768 \leq$ integer $\leq +32767$

2. Длина идентификаторов ´´identifier´´ – до 5 знаков.

3. Конец идентификатора определяется при помощи ограничителя, обычно пробел.

4. Следующие идентификаторы не разрешены для использования:

CMC digit digit (например, CMC12)
ICMCS,
ICMCC,
ICMBR

5. Ограничитель ´´␣´´ означает пробел.

4. СИСТЕМНЫЕ СИМВОЛЫ

Синтаксис:

system–symbol ::= special–system–symbol/fixed–system–symbol

special–system–symbol ::= special–directive/function/camac–flag/system–flag/num–typ

special–directive ::= BEGINCAMAC/ENDCAMAC

function ::= F([n1/n2/n3])

camac–flag ::= LAM/STATUS

system–flag ::= Q/X/S1/S2/S3/S4

num–typ ::= BIN/OCT/DEC

fixed–system–symbol ::= directive/software–symbol/camac–symbol/opcode

directive ::= CAMACSEGMENT/CAMACSEGMENTEND/CEQV/CNAME/CDMD/CDCL/CREF/CACT/EXT

software–symbol ::= EQV/L/EXIT/GO TO/GOTO/NAME/DEMAND/COMLEN/CAMLEN/CAMADR/IF/IFNOT

camac–symbol ::= G1/G2/GL/K/B/C/N/A/I/P/Q/R/S

opcode ::= opcode 1/opcode 2/ opcode 3/ opcode 4/ TRANSFER/LINK/IGNORE/INITIALISE/ENDLAMROUT/REPEAT

Семантика:

1. Символы "n1, n2, n3, opcode1, opcode2, opcode3, opcode4" определяются ниже.
2. Значения разных символов:
   - Q – сигнал Q контроллера каркаса,
   - X – сигнал X контроллера каркаса,
   - S1 – статус триггера "crate inhibit" после выполнения команды M(Ø)F(27) N(3Ø) A(9), где M – вид команды,
   - S2 – статус триггера "crate demand enabled" после выполнения команды M(Ø)F(27)N(3Ø)A(1Ø),
   - S3 – статус триггера "crate demand request" после выполнения команды M(Ø)F(27)N(3Ø),A(11),
   - S4 – не использовано,
   - K – логический номер канала, к которому подключен контроллер каркаса,
   - B – ветвь,
   - C – каркас,
   - N – номер станции,
   - A – субадрес,
   - I – номер разряда в слове КАМАК,
   - G1 – регистр группы 1,
   - G2 – регистр группы 2,
   - P – одновременное (параллельное) обращение к нескольким блокам КАМАК (для N и I),
   - Q – адресное сканирование,
   - R – передача массива при постоянном адресе, BQL,
   - S – передача массива при постоянном адресе, BQL. Момент передачи каждого слова внутри массива синхронизуется сигналом L, конец массива обозначается сигналом Q=0.

Обратите внимание на то, что буква Q в предложенном языке используется два раза. Компилятор определяет значение по контексту.

5. ОПРЕДЕЛЕНИЕ ЯЗЫКА

5.1. Структура программы

Синтаксис:

```
camac-program ::= compiler-control-statement CR-LF
                  program-control-statement CR-LF
                  non-camac-text
                  camac-segment
                  non-camac-text
                  END CR-LF

compiler-control-statement ::= CMC[4][,LC][,LF][,LA][,F][,B]

program-control-statement ::= identifier[(integer{ , integer}...)]

camac-segment ::= CAMACSEGMENT CR-LF
                  {[ text ] declaration-section}...
                  [ text ][ action-section]
                  CAMACSEGMENTEND CR-LF

non-camac-text ::= character ... CR-LF

text ::= ENDCAMAC CR-LF
         non-camac-text
         BEGINCAMAC CR-LF
```

Семантика:

1. Значения параметров в операторе управления работой транслятора:
   - CMC – язык КАМАК расширяет язык FORTRAN,
   - CMC4 – язык КАМАК расширяет язык FORTRAN IV,
   - LC – распечатка исходной программы,
   - LF – распечатка промежуточной программы на языке FORTRAN,
   - LA – распечатка промежуточной программы на языке ASSEMBLER,

F – вывод на ленту промежуточной программы на языке FORTRAN

B – вывод на ленту перемещаемой программы.

2. Оператор управления программой определяется соответственно /2,3/.

3. Строки "non–camac–text" дают возможность включить в любое место программы операторы языка FORTRAN или FORTRAN IV соответственно "CMC" или "CMC4".

5.2. Декларативная часть

Синтаксис:

```
declaration–section ::= equivalence–section/
                        camac–naming–section/
                        demand–naming–section/
                        software–naming–section/
                        reference–section

equivalence–section ::= CEQV  CR–LF
                        {eqv–d1/eqv–d2} CR–LF

eqv–d1 ::= identifier EQV special–system–symbol

eqv–d2 ::= symbolic–constant = {integer/expression}

expression ::= term [{adop term}...]

term ::= primary [{mulop primary}...]

primary ::= constant / (constant { adop constant}...)

constant ::= integer/ symbolic–constant

symbolic–constant ::= identifier
```


Семантика:

Декларированные константы действуют только во время трансляции программы.

Примеры:

```
CEQV
B  EQV  BEGINCAMAC
LESEN EQV  F(Ø)
END  =23
```

Синтаксис:

```
camac–naming–section ::= CNAME  CR–LF
                         {cn–d1/cn–d2}  CR–LF
cn–d1 ::=  c–name [(range)] = address–set [G1/G2][mode]
c–name ::= camac–identifier
range  ::  =constant [:constant]
mode ::  = P/Q/R/S
address–set ::  = address–value[{,address–value}...]
address–value  ::  = address–component [address–component...]
address–component ::  = {c–name [(address–list)]}/
                         {c–type[(address–list)]}
c–type ::  = K/B/C/N/A/ I
address–list ::  = address–element [{,address–element }... ]
address–element  ::= constant [:constant [:constant]]
cn–d2  ::  = c–name [(size)] = EXT
size ::  =1 : constant
```

Семантика:

1. При отсутствии параметров используется:
   range : 1 : 1,
   size : 1 : 1,

группа регистра: G1,
вид операции КАМАК: одноразовая команда.

2. Элементы адресов (address-elements) позволяют подробно обозначать массивы. В случае использования трех элементов транслятор интерпретирует, "начиная с первого , - до второго, с шагом, равным третьему элементу"; в случае двух элементов предполагается, что шаг равен единице, для одного элемента адресов определяется один элемент массива.

3. Элемент адреса занимает в памяти два слова, т.е. 32 разряда.

4. С помощью "C(Ø)" задается использование контроллера каркаса KKØØ4.

Примеры:

```
CNAME
NCR1 =K(2Ø) B  C(2)
IZA=NCR1  N(19)  A(12)
IZST =NCR1  N(1)  A(8)  G2
IZST1 =IZST  I(5)
IZST2 = IZST  I(12)
ICCRP (1:1Ø)=NCR1  N(1:2Ø:2)  P
ICCRQ  (1:1Ø)=ICCRP  (1:1Ø)  Q
KA1 (1:4)=NCR1  N(1:2) A(Ø:1)
KA2 (1:4) =NCR1  A(Ø :1) N(1:2)
```

Необходимо соблюдать разницу в адресных полях KA1 и KA2 соответственно последовательности компонентов N и A:

KA1:

```
K(2Ø)  B(1)  C(2)  N(1)  A(Ø)
K(2Ø)  B(1)  C(2)  N(1)  A(1)
K(2Ø)  B(1)  C(2)  N(2)  A(Ø)
K(2Ø)  B(1)  C(2)  N(2)  A(1)
```

KA2:

```
K(2Ø)  B(1)  C(2)  N(1)  A(Ø)
K(2Ø)  B(1)  C(2)  N(2)  A(Ø)
K(2Ø)  B(1)  C(2)  N(1)  A(1)
K(2Ø)  B(1)  C(2)  N(2)  A(1)
```

Синтаксис:

```
demand-naming-section ::=CDMD  CR-LF
                          { dm-d1/dm-d2 }  CR-LF
dm-d1 ::= demand-name =c-name [ GL integer]
demand-name ::= camac-identifier
dm-d2 :: = demand-name =EXT
```

Семантика:

"GL integer" (например: GL12) указывает на разряд - I в слове состояния контроллера каркаса, т.е. на источник сигнала L в каркасе.

Примеры:

```
CDMD
LAM1 =IZST
LAM2 = ICRA  GL  8
```

Синтаксис:

```
software–naming–section :: = CDCL   CR–LF
                             {cd–dl/cd–d2/cd–d3}  CR–LF
cd–dl : :={COMLEN/CAMLEN}d–item [{, d–item}...]
d–item :: = v–item/a–item/l–item
v–item :: = variable–name
a–item :: =array–name (size)
l–item :: = list–name ( size) L
variable–name :: =identifier
array–name :: = identifier
list–name :: = identifier
cd–d2 :: =CAMADR  p–item [{ ,p–item }...]
p–item :: = p –name [( size) ]
p – name :: = camac–identifier
cd–d3 :: =p–name[(range)]= c–name [(range)]
```

Семантика:

Значения ключевых слов:

COMLEN – Computerlength, длина слова в 16 разрядов,

CAMLEN – Camaclength, длина слова в 24 разряда, (с контроллером KKØØ4 – только 16 разрядов),

CAMARD – определение массива адресов аппаратуры KAMAK.

Примеры:

```
CDCL
COMLEN  MA,MI, NE, KOBU1 (1:2Ø),LISTB(1:1Ø)L
CAMLEN IC, IT, IBUC1 (1:1Ø)L, ILIA (1:2Ø )
CAMADR ICCCC (1:1Ø )
ICCCC(1:5 )= ICCRP (1:5)
ICCCC(6:1Ø) =ICCRQ (1:5 )
```


Синтаксис:

```
reference–section :: =CREF   CR–LF
                      {c–ref 1/c–ref2 }  CR–LF
c–ref 1  :: = NAME  c–name [{,c–name }... ]
c–ref 2 ::  =DEMAND  demand–name[ {,demand–name}...]
```

Семантика:

Определение названий модулей и источников сигналов L в оборудовании в стандарте KAMAK для использования в других "camac–segment" –сегментах одной программы, которая состоит из многих сегментов.

Примеры:

```
CREF
NAME ICR, ICRR, IFCA
DEMAND  LAM2
```

5.3. Исполнительная секция

Синтаксис

```
action–section :: = CACT  CR–LF
      [text]{ action–statement  CR–LF [ text ]}...
action–statement : : =[label]{{[repeat–statement]
      {transfer–statement/control–statement}}/
      branch–statement/org–statement}
      label  :: = integer
      repeat–statement : : = REPEAT (rep–item)
```

rep–item : : = integer/variable–name

transfer–statement : : = t–a1/t–a2/t–a3/t–a4/t –a5

t–a1 : : = opcode 1 external–reference internal–reference

opcode1 : : = F(n1)/READ/READCLR/READCOMP/READLAM/READSTAT

n1 : : = Ø/1/2/3/4/5/6/7/8/27

t–a2 : : = opcode2 { number/internal–reference } external–reference

opcode 2 : : = F (n2)/WRITE/SETSEL/CLEARSEL

n2 : : = 16/17/18/19/2Ø/21/22/23

number : : = symbolic–constant/binary/octal/decimal

binary : : = BIN ´{ Ø/1 }...´

octal : : = OCT ´octal–digit...´

decimal : : = DEC ´digit...´

t–a3 : : = TRANSFER source destination

source : : = external–reference

destination : : = external–reference

t–a4 : : = {F(Ø)/READ } external–reference listname EXIT label

t–a5 : : = {F(16)/WRITE } list–name external–reference EXIT label

external–reference : : = {c–name [(range)]}/{p–name [(range)]}/
direct–reference/demand–name

direct–reference : : = K (constant) [B (constant)]
[C (constant)] N (constant)
A (constant) [I (constant)] [G1/G2]

internal–reference : : = variable–name/array–name [(range)]/
list–name [(range)]

Семантика:

При выполнении операторов с передачей данных используются следующие функции F:

OPCODE	A OR I	REG.-GROUP	FUNC.-CODE	REMARKS
READ	A A	G1 G2	Ø 1	
READCLR	A	G1	2	G2 → ERROR
READCOMP	A	G1	3	G2 → ERROR
READLAM	A	-	8	G1 OR G2 → ERROR
READSTAT	A	-	27	G1 OR G2 → ERROR
WRITE	A A	G1 G2	16 17	
SETSEL	A A	G1 G2	18 19	
CLEARSEL	A A	G1 G2	21 23	
TRANSFER	A A A A	G1 G2 G1 G2	Ø 1 16 17	SOURCE DESTINATION

Примеры:

```
CACT
REPEAT (4)
READCLR ICT (1:1Ø) IBUF1 (1:1Ø )
READLAM LAM1 ILOG
F(16) OCT 3471Ø625´K(2Ø) B(1) C(2) N(19) A(8)
WRITE MLIST NCRT (2:8) EXIT 1295
```

Синтаксис:

control–statement :: = c–a1/c–a2

c–a1 :: = opcode3 external reference

opcode 3 :: = F(n3)/CLEAR/CLEARLAM/ENABLE/DISABLE/
EXECUTE/SET

n3 :: = 9/1Ø/11/12/13/14/15/18/19/21/23/24/25/26/28/29/3Ø/31

c–a2 :: = opcode4 external–reference

opcode 4 :: = INITIALISE/SETINHIBIT/CLEARINHIBIT/
ENABLEINT/DISABLEINT/CLEARSYS

При выполнении управляющих операторов используются следующие функции F:

OPCODE	A OR I B OR C	REG.- GROUP	FUNC.- CODE	REMARKS
CLEAR	A A I I	G1 G2 G1 G2	9 11 21 23	
CLEARLAM	A I	- G2	10 23	G1 OR G2 → ERROR WITH A(12)
ENABLE OR SET	A I I	- G1 G2	26 18 19	G1 OR G2 → ERROR
DISABLE	A I I	- G1 G3	24 21 23	G1 OR G2 → ERROR
EXECUTE	A	-	25	G1 OR G2 → ERROR
INITIALISE	B C	- -	- 26	GENERATION OF 'BZ' WITH N(28) A(8)
SETINHIBIT	C	-	26	WITH N(30) A(9)
CLEARINHIBIT	C	-	24	WITH N(30) A(9)
ENABLEINT	B C	- -	- 26	ENABLE 'BD' INPUT WITH N(30) A(10)
DISABLEINT	B C	- -	- 24	DISABLE 'BD' INPUT WITH N(30) A(10)
CLEARSYS	C	-	26	WITH N(28) A(9)

Примеры:

F(25) K(1Ø) C(Ø) N(12) A(Ø)

CLEAR KRATE (1:23)

CLEARSYS KRATE

ENABLEINT KRATE

Синтаксис:

branch–statement :: = [condition flag] GO TO label

condition :: = IF/IFNOT

flag :: = {camac–flag external–reference }/system–flag

org–statement :: = o–a1/o–a2/o–a3

o–a1 :: = LINK (demand–name, label)/
IGNORE (demand–name,label)

o–a2 :: = INITIALISE (list–name)

o–a3 :: = ENDLAMROUT

Семантика:

1. Операторы "LINK" и "IGNORE" соединяют или отделяют источники сигнала L и соответствующие программные модули, их обслуживающие.

2. Оператор "INITIALISE (list–name) подготавливает массив вида "List", так что следующий доступ происходит к первому элементу.

3. Оператор "ENDLAMROUT" заканчивает программный модуль для обслуживания сигнала L.

Примеры:

LINK (LAM1, 1ØØ)
IGNORE (LAM2, 381)
INITIALISE (LIST)

6. ОШИБКИ, ВОЗНИКАЮЩИЕ ПРИ ТРАНСЛЯЦИИ ПРОГРАММ НА ЯЗЫКЕ КАМАК

Кроме диагностики ошибок, на языке FORTRAN или FORTRAN IV могут появляться сообщения компилятора КАМАК в следующей форме: C–E XXXX:YYYY+ZZZZ, где:

XXXX – номер ошибки (обозначения см. ниже),

YYYY – ближайшая к ошибке метка, пройденная при трансляции,

ZZZZ – количество строк между ''YYYY'' и строкой, в которой находится ошибка (строки с комментариями не считаются).

Значения номера ошибки:

20 – неопределенный оператор,
21 – неразрешенный идентификатор,
22 – неправильная структура в КАМАК-сегменте,
23 – неопределенный идентификатор,
24 – дважды использованный идентификатор,
25 – неправильный диапазон чисел,
26 – исходная программа слишком велика или сложна,
27 – неправильный оператор ''c–ref1'' или ''c–ref2'',
28 – переполнение служебных массивов компилятора,
29 – продолжение строки не разрешается,
30 – не использовано,
31 – неправильный оператор DOS ''$PR,CMC...'',
32 – неправильный оператор управления работой транслятора,
33 – неправильный оператор управления программой,
34 – логический номер канала ЭВМ, на котором подключен контроллер KKØØ4, не определен,
35 – неправильная структура в программе,
36 – компилятор языка FORTRAN не находится в системной библиотеке,
37 – неправильная передача промежуточной программы между компиляторами языка КАМАК и FORTRAN,
38 – в случае ввода исходной программы с бумажной ленты до конца ленты нет оператора ''END$''.

Ввод дополнительных лент разрешается командой ''$GO,Ø'', трансляция продолжается командой ''$GO,1''.

Во время выполнения программ, написанных на языке КАМАК, могут возникать следующие сообщения:

```
*
*  name CMC–RUN–ERR:   XXXX
*       AT K:YY  B:YY  C:YY  N:YY  A:YY F(OR I):YY,
*
```

где:

name – название программы,
XXXX – номер ошибки (о значении см. ниже),
YY – компоненты адреса КАМАК.

Значения номера ошибки:

Ø1 – неопределенный источник сигнала L после прерывания,
Ø2 – внутренняя ошибка во время трансляции,
1Ø – нелегальный компонент в адресе в группе регистров,
11 – нелегальный компонент в адресе для вида передачи,
12 – нелегальный компонент в адресе в позиции I,
13 – нелегальный компонент в адресе в позиции A,
14 – нелегальный компонент в адресе в позиции N,
15 – нелегальный компонент в адресе в позиции C,
16 – нелегальный компонент в адресе в позиции B,
2Ø – после выполнения цикла в контроллере нет сигнала X,
21 – после выполнения специальных команд в контроллере нет сигнала Q,
22 – количество внутренних и внешних ссылок в операторе не совпадает.
23 – неправильный формат внутренних ссылок (напр., ''CAMLEN'' и логическая переменная).

Следующие номера имеют временное значение:

9998 – ''system–flag S4'' не используется,
9999 – сейчас используется только контроллер KKØØ4.

В заключение автор выражает благодарность А.Н.Синаеву, С.В.Медведю и сотрудникам секторов 1 и 4

Отдела новых научных разработок ЛЯП ОИЯИ за полезные обсуждения вопросов при разработке транслятора.

ЛИТЕРАТУРА

1. CAMAC. Proposal for a CAMAC Language
CAMAC–Bulletin, No.5, Nov. 1972
2. Disc Operating System DOS
Hewlett–Packard 02116 -91748, Oct. 1970.
3. Disc Operating System DOS III
Hewlett–Packard 24307–90006, Jun. 1975.
4. Журавлев Н.И. и др. ОИЯИ, 10-8754, Дубна, 1975.

Рукопись поступила в издательский отдел
16 марта 1977 года.