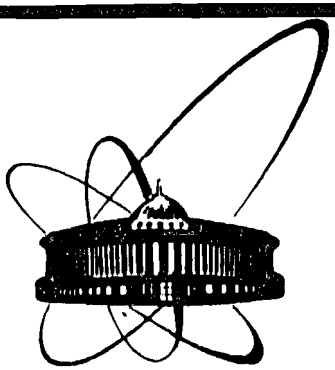


сообщения
объединенного
института
ядерных
исследований
дубна

11-85-354

Д.Вернер*, Л.И.Городничева, Д.Крайзелер,
В.П.Николаев, Д.Хенш*

КОМІ - ОПЕРАЦИОННАЯ СИСТЕМА РЕАЛЬНОГО ВРЕМЕНИ ДЛЯ МИКРОПРОЦЕССОРОВ

* Технический университет, Дрезден, ГДР

1985

ВВЕДЕНИЕ

KOMI /координатор микропроцессоров/ - управляющая система реального времени для микропроцессоров Z80/1/и INTEL 8080. KOMI управляет работой параллельных процессов:

- управление вычислительными процессами;
- управление обменом данных;
- управление распределением ресурсов;
- управление в режиме реального времени;
- обслуживание прерываний.

Существующие возможности системы можно использовать через системные директивы, которые пользователь включает в свою программу.

Общение оператора с системой происходит посредством диалога.

Вариант KOMI, содержащий все ее возможности, занимает немного меньше 4К постоянной памяти /ППЗУ/. Необходимый объем ОЗУ зависит от количества и длины векторов управления TASK и от количества семафоров и таймеров /около 1К/.

Максимальное время выполнения системной директивы в оптимальных условиях 1 мс.

Пользователь может сгенерировать систему согласно своим потребностям. Система KOMI имеет модульную структуру, и существует возможность включения в нее новых модулей.

Систему KOMI особенно удобно использовать для автоматизации технологических и лабораторных процессов.

1. TASK

TASK - это программа пользователя, оформленная специальным образом в системе KOMI.

Каждая TASK описывается через вектор управления (TSV). Для создания TASK необходимо подготовить следующие параметры.

1. Приоритет

Каждая TASK должна иметь приоритет P ($1 \leq P \leq 250$). (250 - высший, 1 - низший приоритет). KOMI реализует возможность вытеснения TASK с низким приоритетом появившейся TASK с высоким приоритетом. В случае появления двух TASK с одинаковыми приоритетами система работает по принципу FIFO (первый - вошел, первый - вышел).

2. Номер TASK

Чтобы определить TASK, надо присвоить ей номер от 1 до 127. По этому номеру можно сделать изменения в определенной TASK.

3. Номер семафора /только для определенных TASK/

Во время существования TASK система может предоставлять ей определенные ресурсы.

Ресурсы предоставляются TASK через семафоры, когда она находится в состоянии готовности при условии, что требуемые ресурсы не заняты в это время другой TASK.

4. Адрес параметров /только для определенных TASK/

Адрес параметров необходим для осуществления возможности перемещения данных. При этом описываемой TASK будет передан начальный адрес блока данных /параметров/. Когда TASK будет в состоянии активности, этот адрес будет записан в H-L регистры.

5. Стартовый адрес

Через стартовый адрес происходит связь конкретной программы с созданной TASK. Стартовый адрес находится не в TSV, а в стеке, связанном с данной TASK. Начальный указатель стека находится в TSV. Каждая программа, которая управляется TASK, должна начинаться командой непосредственной загрузки указателя стека (LXI SP, STACK). Последней командой программы должна быть команда перехода (JMP) к системной директиве TERMINATE, которая осуществляет окончание TASK и выход в систему. В любой момент выполнения TASK ее стек должен иметь свободными не менее 16 байт, которые необходимы для спасения TASK во время прерывания TASK. Пример: программа, которая управляется TASK.

```
TASKX:  LXI     SP, STACK       загрузка указ.стека
                                  :
        JMP     TERM            конец TASK

        DS      16              16 байт /для спасения/,
STACK:  EQU                     область стека
```

В системе KOMI TASK может находиться в различных состояниях. Для каждого состояния создается список. Связь происходит через слово связи в TSV.

Существуют следующие состояния TASK:

- несуществующее: вектор управления (TSV) для данной TASK пустой;
- пассивное: условие, необходимое для продолжения TASK, не выполнено. При определенной генерации система KOMI может различать несколько видов пассивного состояния;

- "готов к работе": TASK ждет, когда процессор будет свободен;
- активное: TASK выполняется в процессоре.

Существенной частью системы KOMI является переключатель состояний TASK. Существуют специальные системные директивы, которые активируют переключатель состояний. Он перепишет TASK из очереди, где она находилась, в очередь, соответствующую ее новому состоянию. Это будет сделано с учетом приоритета TASK. При этом переключатель состояний выбирает TASK с самым большим приоритетом и делает ее активной.

2. СТРУКТУРЫ ДАННЫХ

В работе KOMI-системы используют различные структуры данных, которые характеризуют состояния всех компонентов системы.

2.1. Вектор управления TASK (TSV)

Каждая программа, которая будет выполняться в системе KOMI, должна иметь свой вектор управления, который описывает информацию о программе, необходимую для KOMI, и который в процессе работы TASK содержит информацию о всех ее актуальных состояниях. TSV содержит от 7 до 15 байт, которые имеют следующие значения:

1 2 3 4 5 6 7 8 9 10

1 - слово связи /2 байта/ со следующим элементом очереди состояния;
2 - приоритет;
3 - номер семафора для управления ресурсами системы;
4 - номер TASK;
5 - адрес параметров;
6 - указатель стека;
7 - адрес TSV TASK, которая должна работать после окончания данной TASK;
8 - байт состояния;
9 - номер семафора, к которому была привязана TASK в момент прерывания;
10- адрес TSV TASK, связанной с данной TASK и созданной с помощью директивы CREATE-STOP.

Все подчеркнутые параметры используются только в определенных случаях.

2.2. Указатели состояния очереди

В системе KOMI существуют очереди для каждого состояния TASK. Каждая очередь имеет свое определенное имя /рис.1/.

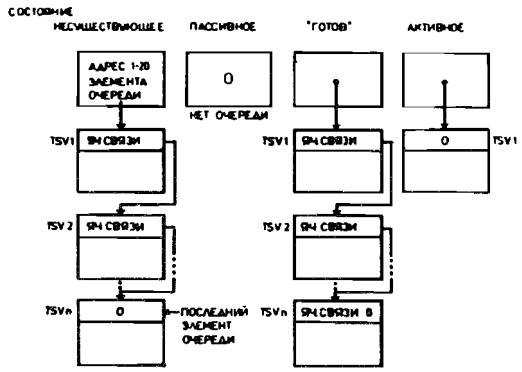


Рис.1

2.3. Семафоры

Семафоры служат для координации различных действий между параллельными процессами. Длина семафора равна 3 байт. Он состоит из управляющей переменной /1 байт/ и ссылки на первый элемент очереди /список TASK, ожидающих что-то из ресурсов системы/ - 2 байта /рис.2/.

KOMI различает три вида семафоров: семафоры ресурсов; семафоры событий; семафоры, управляющие передачей данных.

Структура этих семафоров одинакова, а их содержимое зависит от выполняемых операций.

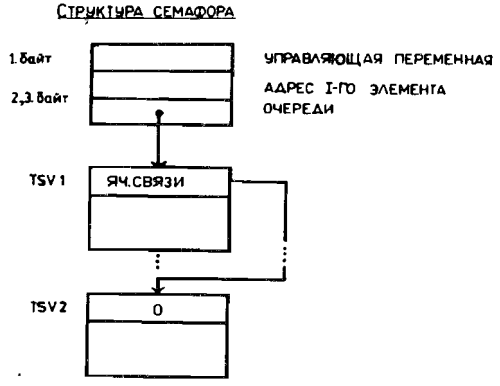


Рис.2

Семафоры ресурсов осуществляют связь между TASK и необходимыми для нее ресурсами. Значение управляющей переменной:
- ">ø" - число свободных ресурсов;
- "<ø" - количество необслуженных запросов /количество TASK в очереди.

Семафоры событий разрешают различным TASK ожидание события, необходимого для этих TASK. Во время ожидания события эти TASK

блокируются. Когда произойдет ожидаемое событие, все TASK, нуждающиеся в этом событии, разблокируются. Значение управляющей переменной семафора ">ø" - количество необходимых событий, которые еще не произошли.

Семафоры, управляющие передачей данных, разрешают обмен блоками данных между TASK. Значение управляющей переменной семафора:
- ">ø" - количество блоков данных;
- "<ø" - количество ожидающих TASK.

2.4. Таймеры

Таймер используется для работы в режиме реального времени. Таймер занимает 8 байт ОЗУ. TASK, нуждающаяся в таймере, получает его при условии, что таймер был свободен.

Структура таймера:

1 2 3 4

1 - счетчик тактов таймера;
2 - период повторений;
3 - системная директива, которая должна выполняться;
4 - адрес TSV TASK.

3. ДИРЕКТИВЫ СИСТЕМЫ

Отдельные директивы используются в TASK как подпрограммы. После вызова директивы все регистры будут спасены в стеке TASK. После выполнения директивы будет восстановлено содержимое регистров TASK.

Выбор необходимых для данной конфигурации системы директив происходит во время генерации системы.

Существует три вида директив:
- директивы для организации TASK;
- директивы для управления ресурсами, синхронизации и передачи данных между TASK;
- специальные директивы.

3.1. Директивы для организации TASK

Эти директивы непосредственно управляют TASK.

CREATE - директива создания TASK.

Определение TASK происходит через блок параметров, которые описывают всю информацию, необходимую для создания TASK. На регистрах D-E необходимо указать адрес блока параметров. Блок параметров: - приоритет /1 байт/ ;
- номер семафора /1 б/ *;

- номер TASK /1б/;
- адрес буфера данных /2б/ * ;
- стартовый адрес /2б/;

* - эти параметры указываются в случае, если при генерации системы они были разрешены.

CREATE-STOP - директива создания TASK с ожиданием завершения другой TASK.

Блок параметров аналогичен директиве CREATE.

Если в следующем примере для создания TASK2 использовать директиву CREATE-STOP, то TASK1 будет ждать завершения выполнения TASK2, и только потом работа TASK1 будет продолжена.

Пример создания TASK2 с помощью TASK1.

```
TASK1:    LXI   SP, STCK1          указан.стека
            :
            :
          LXI   D,PARA2            адрес приготовлен-
            :                      ного блока пара-
            :                      метров
          CALL CRT                 вызов сист. под-
                                   программы
          JMP   TERM
            :
            :                      окончание
          DS 16
STCK1:    EQU                      стек
PARA2:    DS    5                  приоритет

                                   номер семафора для
                                   управления опреде-
                                   ленным ресурсом

          DB    2                  номер TASK
          DW    BUFF               адрес для передачи
                                   данных в TASK2
          DW    TASK2              стартовый адрес
TASK2:    LXI   SP,STCK2           указатель стека
          JMP   TERM               окончание
          DS    16                 стек
STCK2:    EQU
```

CREATE-DELAY - директива, позволяющая начать выполнение TASK через определенное время или /и/ делать ее запуск периодически через определенные интервалы времени. Необходимо в регистрах D-E указать адрес блока параметров.

Блок параметров:

- число тактов до первого запуска /2б/;

- число тактов для периода между повторными запусками /2б/;
- следующие параметры аналогичны директиве CREATE.

Число тактов до первого запуска TASK не должно быть = Ø.

Пример: TASK должна запускаться через каждые 1,5 с.
Первый запуск - через 7 с. Тактовый период равен 10 Hz.

```
          LXI   D,PARA             адрес параметров
          CALL  CRTD               вызов директивы
PARA:     DW    7Ø                 первый запуск через 7 с
          DW    15                 период перезапуска через
                                   1,5 с
          DB    12                 приоритет
          DB    Ø                  семафор не используется
          DB    Ø                  номер TASK не используется
          DW    Ø                  адрес буфера данных не нужен
          DW    TASK D             стартовый адрес
```

ERASE - появление этой директивы прекращает выполнение TASK, созданной с помощью директивы CREATE-DELAY. В регистрах D-E необходимо указать адрес блока параметров, который использовался директивой CREATE-DELAY.

TERMINATE - окончание TASK. Завершение выполнения TASK происходит по директиве TERMINATE. При этом будет выполнено:

- разблокировка TASK, ожидающей завершения данной TASK;
- освобождаются ресурсы, используемые данной TASK в соответствии с семафором;
- стирается TSV законченной TASK.

Для вызова директивы TERMINATE используется команда JMP.

STOP - директива останавливает выполнение TASK. Регистр E должен содержать номер TASK, которую надо остановить. Директива STOP будет игнорирована, если TASK не существует. Если регистр E равен Ø, то остановится TASK, издавшая директиву STOP.

CONTINUE - директива, обратная STOP. Для продолжения TASK, остановленной по директиве STOP, следует издать директиву CONTINUE. В регистре E указывается номер TASK.

DELAY - временная задержка работы TASK. В регистрах D-E указывается количество тактов таймера.

KILL - эта директива вызывает принудительное окончание TASK. Происходит стирание TSV этой TASK и освобождение необходимых ей ресурсов. Если какая-то TASK ждет конца выполняющейся в данный момент TASK, то она тоже будет закончена. Те ресурсы, которые заказаны в директиве REQUEST, не будут освобождены. В регистре E должен содержаться номер TASK.

3.2. Директивы для распределения ресурсов, синхронизации и перемещения сообщений

Для предоставления ресурсов используются следующие директивы:

REQUEST - предоставляет TASK требуемый ресурс системы. В регистре E необходимо указать номер семафора. При требовании ресурса по директиве REQUEST происходит декремент управляющей переменной семафора данного ресурса. Если результат после декремента <Ø, то ресурс уже был занят, и TASK будет ждать выполнения другой TASK директивы RELEASE.

RELEASE - освобождение ресурса. В регистре E указывается номер семафора. При освобождении ресурса по директиве RELEASE происходит инкремент управляющей переменной семафора данного ресурса. Если результат после инкремента ≤0, то ресурс предоставляется той TASK из очереди, которая имеет самый большой приоритет.

Пример: Ресурс ARITH - арифметическая подпрограмма. Она может использоваться многими TASK. В данный момент времени ее может использовать только одна TASK. ARITH управляется через семафор номер 2.

```
MVI  E,2              номер семафора
CALL REQU             директива REQUEST
 :
 :
CALL ARITH            вызов подпрограмм ARITH
 :
 :
MVI  E,2              номер семафора
CALL RLSE             директива, освобождающая ARITH.
```

Для синхронизации событий используются директивы:

AWAIT - происходит ожидание одного или нескольких событий. В регистре E необходимо указать номер семафора. Если управляющая переменная семафора >Ø, то события, которых ждет активная TASK, еще не выполнены.

CAUSE - директива, обратная AWAIT. Регистр E содержит номер семафора. Когда произойдет событие, которого ждут какие-то TASK, управляющая переменная семафора декрементируется. Если значение этой переменной ≤Ø, то все TASK, блокированные ожиданием, освобождаются. Значение управляющей переменной >Ø дает информацию о количестве событий, которые еще не произошли.

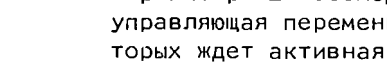
Пример: синхронизация нескольких TASK. Семафор 4 указывает на трехкратную работу TASK1. Работа TASK2 должна начаться после этого события. При инициализации значение управляющей переменной семафора установлено 3.


```
TASK1:  LXI   SP,STACK1          указатель стека
         :
         .
        MVI   E,4                номер семафора
        CALL  CAUSE              сообщение о событии
        JMP   TERM               конец TASK1

TASK2:  LXI   SP,STACK2          указатель стека
        MVI   E,4                номер семафора
        CALL  AWAIT              все события произошли?
         :
        JMP   TERM               конец TASK2
```

Передача данных между TASK происходит по следующим директивам:
ASK - директива требования и ожидания данных. E-регистр содержит номер семафора; H-L-регистры указывают адрес блока данных. Директива ASK декрементирует управляющую переменную семафора и передает TASK адрес блока данных. Если управляющая переменная <Ø, то требуемого блока данных не существует, и TASK будет блокирована. Если блоки данных существуют /минимально - 1/, то управляющая переменная >Ø. Эти блоки данных получит та TASK, которая находилась в состоянии ожидания дольше других.
SEND - директива, с помощью которой передается блок данных. E-регистр содержит номер семафора; H-L-регистр - адрес блока данных. Директива SEND инкрементирует управляющую переменную семафора и реализует передачу блока данных для TASK. Если несколько TASK ждут блока данных, то его получит та TASK, которая дольше других находилась в состоянии ожидания. Если не существует TASK, ожидающих этого блока, то он будет передан формально по номеру семафора. Активная TASK, передающая этот блок, всегда останется в активном состоянии. Блок данных организует пользователь.

Блок данных имеет следующую структуру:



1 - ячейка связи,
2 - байты данных.

3.3. Специальные директивы

SETPRIO рег. E содержит номер TASK,
         рег. D - новый приоритет.

Директива SETPRIO меняет приоритет TASK во время ее существования. С помощью этой директивы можно изменить приоритет другой TASK. В случае, когда TASK меняет приоритет для себя, регистр E должен быть равен Ø.

Пример: Существуют TASK с разными приоритетами.

```
         :
        MVI   E,33       номер TASK
        MVI   D,1Ø       новый приоритет
```

```
            CALL  PRIO      директива
              :
              :
SETSEMA       рег. E содержит номер семафора
              рег. D - новое значение переменной управляющей
              семафора.
```

С помощью SETSEMA можно менять переменную семафора.
После инициализации управляющие переменные семафоров устанавливаются в "1".

OWN С помощью этой директивы TASK получает адрес своего TSV.

```
SETTIME  E-рег. содержит час или день;
         D-рег.  -"-     мин или мес.;
         A-рег.  -"-     с или год;
```

с помощью этих директив устанавливаются время или дата.

Пример:

```
            MVI  A, Ø     ;  с
            MVI  D, 25    ;  мин
            MVI  E, 11    ;  час.
            CALL SETT
```

TIME - рег. D-E содержит адрес буфера /8 байт/, где находится
DATE в ASCII-коде значение времени или даты.
Эти директивы берут установленное время или дату по директивам SETT или SETD для выдачи на печать.
Формат буфера: час мин с
                день мес. г

Пример:

```
            LXI  D, ZEIT    буфер
            CALL TIME       директива
              :
              :
       ZEIT: DS   8
```

3.4. Возможности контроля выполнения некоторых системных директив

Контроль можно осуществлять анализом C-флага. C-флаг устанавливается по окончании выполнения директивы.

Установка C-флага в "1" дает информацию о ненормальном завершении работы директивы.

Сист. директива	C=1
CREATE	Нет свободной памяти для размещения TSV
CREATE-STOP	-"-
STOP	система не нашла TASK с указанным номером
SETSEMA	значение управляющей переменной семафора S<0 или S>1. В этом случае регистр D содержит прежнее значение S.

SETPRIO	система не нашла TASK с указанным номером
CREATE-DELAY	нет свободного таймера
DELAY	-"-
ERASE	не найден таймер, по которому выполнялась директива CREATE-DELAY
KILL	не найдена TASK с указанным номером

4. СЛУЖЕБНАЯ ПРОГРАММА ДЛЯ ВВОДА ДИРЕКТИВ С КЛАВИАТУРЫ /СЛУЖЕБНАЯ TASK/

Эта программа организована в системе как TASK. Существуют 2 возможности использования служебной TASK:

- служебная TASK находится в состоянии ожидания /TASK имеет низкий приоритет/;
- в момент необходимости служебную TASK надо активизировать /через прерывание от клавиатуры/.

Директивы, которые можно ввести с клавиатуры:

1. <CR> - выход из служебной TASK
2. CREATE - создается TASK
   CRT PRIO (D), SEMA (D), TANR (D)
   PARA (DD), START ADR (DD) <CR>
3. KILL - принудительное окончание TASK и освобождение ресурсов KIL TANR (D) <CR>
4. CREATE-DELAY - периодический запуск TASK
   CDC DEL (DD), PERIOD (DD),...
   параметры директивы CREATE <CR>
5. ERASE - стирает таймер и поэтому прекращает периодическое создание TASK
   ERS TANR(D) <CR>
6. STOP - остановить выполнение указанной TASK,
   STR TANR (D) <CR>
7. CONTINUE - продолжение выполнения TASK, остановленной директивой STOP
   CNT TANR(D) <CR>
8. SETTIME - установить время
   SET ST(D),MI(D),YY(D) <CR>
9. SETDATE - установить дату
   STD TT(D),MM(D),YY(D),<CR>
10. Вывод даты и времени.
    DATE <CR>
11. Вывод списка TASK, находящихся в состоянии готовности.
    DRDY <CR>
12. Вывод списка TASK, которые были остановлены командой STOP.
    DST P <CR>

13. Вывод списка TASK, которые ждут определенного семафора.
DSEMNR(D) <CR>

Обозначения:
"D" - десятичное число 0÷255;
"DD" - -"- 0÷65535;
"O" - восьмеричное число;
"H" - шестнадцатиричное число;
"B" - двоичное число.

5. СИСТЕМА ПРЕРЫВАНИЙ (IBR)

Во время работы TASK может возникнуть прерывание. Состояние прерванной TASK будет спасено системной программой SAVE. Запись состояния TASK происходит в стек этой TASK. Затем SAVE переключается на системный стек. Программа обработки прерывания работает с системным стеком. Выход из состояния IBR происходит с помощью системной программы LOAD, которая восстановит состояние прерванной TASK и активизирует ее. Во время выполнения IBR могут выполняться любые действия, но с учетом временных затрат /они должны быть минимальными/.

Вызов системных директив в IBR отличается от вызова их в TASK пользователей. Вызов SAVE и LOAD должны находиться внутри IBR.

Пример: структура IBR

```
IBRØ:  CALL SAVE        - спасение состояния
       MVI E,2O
  CALL CONT Ø
  .
  JMP  LOAD             - восстановление состояния
                          прерванной
```

ЗАКЛЮЧЕНИЕ

В настоящее время в составе системы автоматизированного управления головной части коллективного ускорителя тяжелых ионов /КУТИ-20/ используется микро-ЭВМ КМ001 с микропроцессором КР580ИК80А /INTEL 8080//2/. На основе этой микро-ЭВМ разработан синхронизатор КУТИ-20, обеспечивающий синхронизацию работы технологических систем ускорителя. Благодаря возможностям контроллера прерываний микро-ЭВМ реализован мультипрограммный режим работы процессора микро-ЭВМ. Поскольку микро-ЭВМ сама по себе не имеет никаких средств, обеспечивающих координирование нескольких асинхронных функций, эти функции координируются непосредственно прикладными программами.

Вероятность ошибки при программировании таких функций весьма велика, что затрудняет включение дополнительных задач в ре-

жим мультипрограммирования. Для исключения этих трудностей при разработке программного обеспечения системы управления с применением микро-ЭВМ линейного ускорителя электронных колец /ЛУЭК/ будет использована операционная система KOMI для организации координирования параллельных процессов.

ЛИТЕРАТУРА

1. Höntsch D., Werner D. Echtzeitbetricbssyśtem KOMI 3.3 Eine Einführung. TU Dresden, Sekt. Informationsverarbeitung, Bereich Bechnersysteme, 1983.
2. Сидоров В.Т., Синаев А.Н., Чурин И.Н. ОИЯИ, 10-12481, Дубна, 1979.

Рукопись поступила в издательский отдел
13 мая 1985 года.