

MSX CP/M

Версия [CP/M](#) для MSX

[Yamaha MSX CP/M Operation Manual](#), из комплекта [Ямаха КУВТ 2](#).

CP/M 2.2 Net-Shell Version

Версия 1987.10/29

Использовалась в классах [Ямаха КУВТ 2](#) для работы [Yamaha Локальная сеть, версия 3.0](#).

[MSX 2 CP/M 2.2 Net-Shell Classroom Network 3.0](#)

Ученик

Запуск

```
CALL CPM
```

или

```
_CPM
```

Выход из системы

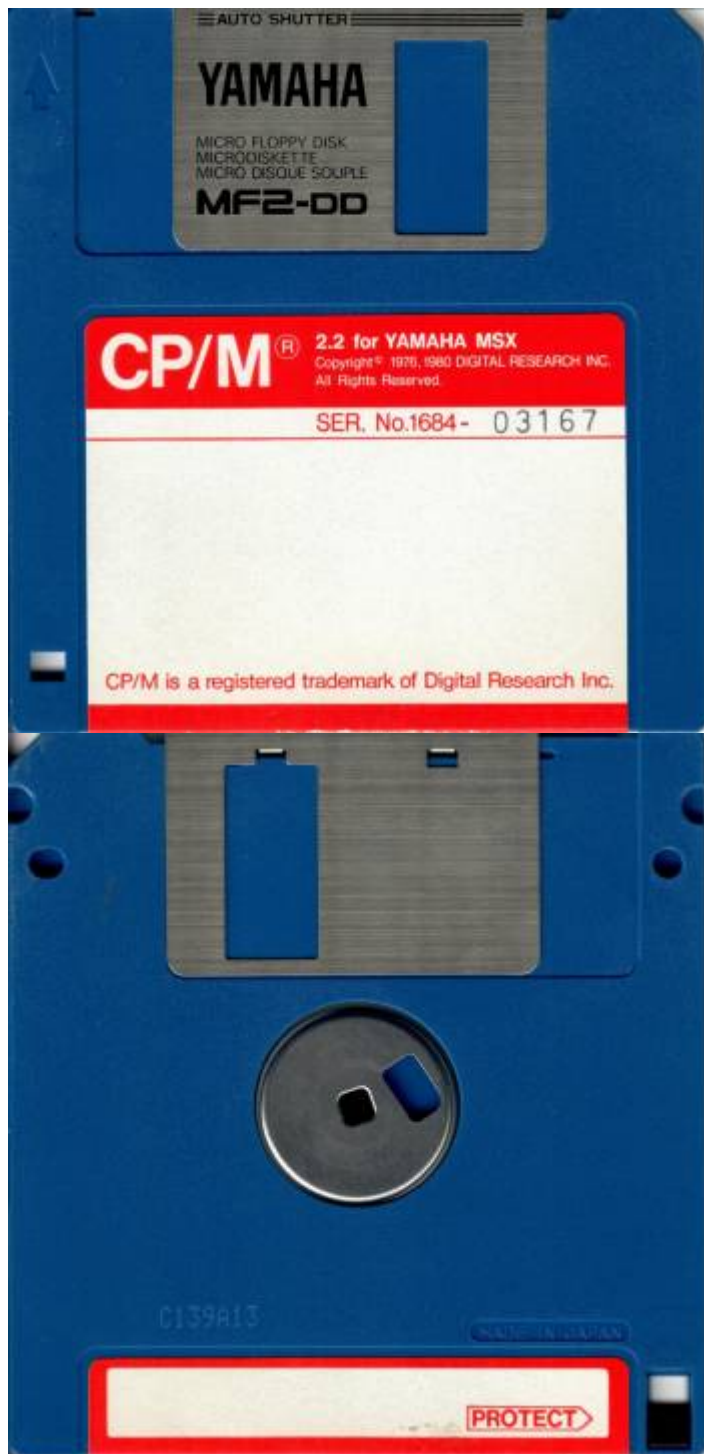
```
BASIC
```

 [Запуск в WebMSX](#)

Дамп ПЗУ, которое устанавливалась в [Yamaha YIS-503IIIR](#):

yis-503iiir_cpm.bin	MD5, SHA1
-------------------------------------	---------------------------

Учитель



Фотографии предоставил **Александр Страйстар**

С июня 2023 года, благодаря **SuperMax** доступен образ оригинального диска (считалось, что все такие дискеты давно переформатированы) для Yamaha YIS-805-128R2 :

[CP/M 2.2 for Yamaha MSX, диск SER. No.1684-03011](#)

Настройки для SteinBlume

[SteinBlume. CP/M Disk Image Explorer](#)

Для работы с этим образом дискеты на ПК можно воспользоваться [cpmtools](#), для этого нужно добавить в файл описания форматов *diskdefs* следующее:

[/etc/cpmtools/diskdefs](#)

```
diskdef cpm2kuvt-720
```

```
seclen 512
tracks 160
sectrk 9
blocksize 2048
maxdir 128
skew 3
boottrk 4
os 2.2
end
```

Примеры команд:

- Список содержимого диска:

```
cpmls -f cpm2kuvt-720 msx2_cpm22.dsk
```



Предшествующая попытка реконструировать дискету, предпринятая **Caro** — [cpm22msx.zip](#), [оригинал](#)

Её определение для cpmtools:

[/etc/cpmtools/diskdefs](#)

```
diskdef cpm22msx-720
seclen 512
tracks 160
sectrk 9
blocksize 2048
maxdir 128
skew 1
boottrk 4
os 2.2
end
```



Диск с надписью CP/M который шел в комплекте #24

CP/M 3.0



оригинал

1 дискета × 720 Кбайт, оригинал

2 дискеты × 360 Кбайт, оригинал

MSX2 with CP/M 3.0

CP/M Plus 3.0 Users Guide, оригинал

CP/M Plus 3.0 Users Guide (версия для печати), оригинал

Beer IDE

CP/M работает только с разделами фиксированного размера (360 либо 720Кб). Работы по реализации поддержки [Beer IDE 202](#) в CP/M продолжаются. Скачать рабочий образ HDD с CP/M 3.1

[здесь](#)

Утилиты



PIP

Программа обмена с периферийными устройствами.

Параметры запуска:

PIP	загрузка программы PIP в оперативную память и переход в режим диалога
-----	---

PIP y:=x:filename.ext PIP y:=x:filename.* PIP y:=x:*.ext PIP y:=x:*.*	копирование файла (файлов) с диска «х:» на диск «у:» (возможно использование метасимволов * и ?)
PIP y:newfile.ext= x:oldfile.ext	копирование с диска «х:» на диск «у:» с изменением имени файла «oldfile.ext» на «newfile.ext»
PIP y:newfile.ext= x:old1.ext, old2.ext, old3.ext	копирование с диска «х:» на диск «у:» с объединением файлов «old1.ext», «old2.ext» и «old3.ext» в файл «newfile.ext»
PIP CON:=x:filename.ext PIP TTY:=x:filename.ext PIP CRT:=x:filename.ext	копирование файла «filename.ext» с диска «х:» на указанное консольное устройство
PIP PUN:=x:filename.ext PIP PTP:=x:filename.ext PIP OUT:=x:filename.ext PIP AUX:=x:filename.ext	копирование файла «filename.ext» с диска «х:» на указанное устройство вывода
PIP LST:=x:filename.ext PIP LPT:=x:filename.ext PIP PRN:=x:filename.ext	копирование файла «filename.ext» с диска «х:» на указанное устройство печати
PIP x:filename.ext=CON: PIP x:filename.ext=TTY: PIP x:filename.ext=CRT:	копирование файла «filename.ext» на диск «х:» с указанного консольного устройства до ввода ^Z
PIP x:filename.ext=RDR: PIP x:filename.ext=PTR: PIP x:filename.ext=INP:	копирование файла «filename.ext» на диск «х:» с указанного устройства, работающего в режиме «только ввод», до ввода ^Z
Для перфоленты:	
PIP PUN:=NUL: PIP PUN:=NUL:,x:file.ext PIP PUN:=NUL:,x:file.ext,NUL: PIP PUN:=x:file.ext,NUL:	вывод начальных и хвостовых промежутков при копировании файла «file.ext» с диска «х:» на устройство, работающее в режиме «только вывод»
PIP PUN:=x:file.ext,E0F	вывод маркера конца файла при копировании файла «file.ext» с диска «х:» на устройство, работающее в режиме «только вывод»

Дополнительные параметры команды PIP:

(пишутся в конце командной строки в квадратных скобках)

[V]	проверить правильность копирования путем сравнения результирующего и исходного файлов
[E]	отображать копируемый файл на консоли
[Sstring^Z]	начать копирование после того, как в исходном файле будет обнаружена строка «string» (строку завершает ^Z)
[Qstring^Z]	завершить копирование после того, как в исходном файле будет обнаружена строка «string» (строку завершает ^Z)
[L]	преобразовать в процессе копирования все символы верхнего регистра в символы нижнего регистра
[U]	преобразовать в процессе копирования все символы нижнего регистра в символы верхнего регистра
[Dn]	удалять в процессе копирования в каждой строке все символы, расположенные после позиции с номером n
[Tn]	преобразовать в процессе копирования все символы табуляции кода ASCII в n пробелов
[F]	удалить в процессе копирования все символы перевода формата
[Pn]	вставить через каждые n строк в процессе копирования символы перевода формата
[N]	дополнить в процессе копирования строки соответствующими порядковыми номерами без ведущих нулей
[N2]	дополнить в процессе копирования строки соответствующими порядковыми номерами с ведущими нулями
[R]	копировать «системный» файл (определенный с помощью команды STAT)
[W]	удалить файл со статусом защиты «R/O» (устанавливается с помощью команды STAT), имеющего то же имя, что и выходной файл

[Gn]	копировать файл (файлы) из области пользователя с номером «n» (от 0 до 15) в файл, расположенный в текущей области пользователя
[O]	рассматривать содержимое копируемого файла (файлов) как объектный (машинный) код
[B]	установить блочный режим копирования
[H]	использовать 16-ричный формат фирмы Intel при передаче данных на (с) устройство, работающее в режиме «только вывод» («только ввод»)
[I]	при передаче в 16-ричном формате фирмы Intel игнорировать все нулевые (NULL) записи
[Z]	в процессе копирования символов в коде ASCII устанавливает бит чётности в 0

SUBMIT

Запуск на выполнение последовательности команд ОС CP/M.

Параметры запуска:

```
SUBMIT filename a b c ...
```

запуск на выполнение командного файла «filename» (с расширением «.SUB») с предварительной заменой в тексте параметров:

- «%1» («\$1») значением «a»
- «%2» («\$2») значением «b»
- «%3» («\$3») значением «c»
- и т.д.

ASM & Z80

- ассемблер для микропроцессора I8080 (ASM.COM);
- ассемблер для микропроцессора Z80 (Z-80.COM).

Параметры запуска:

```
ASM filename.123
Z80 filename.123
```

ассемблировать исходный файл «filename» (с расширением .ASM) и создать файл с промежуточным кодом (с расширением .HEX). При этом дополнительные параметры трактуются следующим образом:

«1»	в этой позиции указывается имя диска, на котором находится исходный файл
«2»	в этой позиции указывается имя диска, на который надо записать выходной файл с расширением .HEX, или символ Z, если объектный файл не требуется
«3»	в этой позиции указывается имя диска, на который надо записать выходной файл печати с расширением .PRN, или символ X, если листинг ассемблирования надо вывести на экран

LOAD

Программа преобразования промежуточного файла в исполняемый машинный код.

Параметры запуска:

```
LOAD filename
```

преобразовать объектный файл с расширением .HEX в файл с исполняемым машинным кодом (с расширением .COM)

DUMP

Программа распечатки содержимого файла.

Параметры запуска:

```
DUMP filename.ext
```

вывести в 16-ричном формате содержимое файла «filename.ext» </code>

DDT & ZSID

Инструментальное средство отладки для микропроцессора I8080 (DDT.COM) или для микропроцессора Z-80 (ZSID.COM).

Параметры запуска:

```
DDT [filename.ext]  
ZSID [filename.ext]
```

загрузить программу DDT или ZSID и (если присутствует) файл с отлаживаемой программой «filename.ext» в память.

При загрузке файла печатается сообщение:

```
PC: nnnn      NEXT: kkkk
```

где:

- nnnn — текущее значение счетчика команд (PC),
- kkkk — адрес следующей свободной ячейки после загрузки файла.



Команды программ DDT и ZSID:

D Dstart Dstart,end	вывести в 16-чном формате содержимое следующих 192 ячеек памяти, начиная с текущей («D»), или с ячейки, расположенной по заданному адресу («Dstart»), или с ячейки «start» по ячейку «end» («Dstart,end»)
Fstart,end,data "data".	- записать во все ячейки памяти с адреса "start" по адрес "end" 16-чное значение "data".
Ifilename.ext	- создать блок управления файлом (FCB) и записать в него имя "filename.ext" для последующей загрузки файла с этим именем в память по команде R.
R Rstart	- загрузить содержимое файла, имя которого указано в команде I, в память, начиная с текущей ячейки ("R"), или с ячейки, расположенной по адресу "start" ("Rstart").
L Lstart Lstart,end	- реассемблировать и вывести содержимое следующих 192 ячеек памяти, начиная с текущей ("L"), или с ячейки, расположенной по заданному адресу ("Lstart"), или с ячейки "start" по ячейку "end" ("Lstart,end").
Sstart	- вывести в 16-чном формате и, возможно, изменить содержимое ячейки памяти, расположенной по адресу "start". Продолжить обработку следующих ячеек до ввода символа "."
Astart	- вставить, начиная с адреса "start", оператор языка Ассемблер. Продолжить обработку следующих ячеек до ввода символа "." или <CR>.
Mstart1,end1,start2	- копировать содержимое области памяти, расположенной с адреса "start1" по адрес "end1", в другую область па-

мента, расположенную, начиная с адреса "start2".

G - выполнение программы: полностью, без прерываний;
Gstart начиная с адреса "start";
Gstart,end начиная с адреса "start", с прерыванием по адресу "end";
Gstart,end1,end2 начиная с "start", с прерыванием в "end1", либо в "end2";
G,end начиная с текущей ячейки, с прерыванием по адресу "end";
G,end1,end2 начиная с текущей, с прерыванием в "end1", либо в "end2".

Unumber - выполнить "number" машинных инструкций и по завершению вывести содержимое всех регистров ЦП.

Tnumber - трассировать выполнение "number" машинных инструкций, распечатывая содержимое всех регистров ЦП после выполнения каждой инструкции.

Xregister - распечатать содержимое регистра "register" или бита условия (регистр F: C,Z,M,E,I) и изменить его, если необходимо.

Ha,b - использование встроенного калькулятора 16-чных чисел:
а и b - 16-чные числа. Команда после <CR> выдает два числа: a+b и a-b.

Дополнительные команды:

^S - приостанов;
^P - установка эхо-печати;
^N - отмена эхо-печати;
^C - выход из программы с ОС CP/M.

Примечание:

В отлаживаемой программе для возврата в DDT или ZSID ставится не RET, а RST #07.

DESPool

Программа печати в фоновом режиме (DESPool.COM).

Параметры запуска:

DESPool - загрузка программы с диска в память. После загрузки вводится ^F, после чего программа спрашивает файл для печати:

File name: (надо ввести имя файла)

Далее начинается печать и появляется подсказка системы:

A> или др.

LINK-80

Программа-сборщик фирмы Microsoft (L80.COM).

Функции программы:

1. Загрузка перемещаемых объектных модулей (.REL).
2. Вычисление абсолютных адресов для всех локальных ссылок внутри модулей.
3. Разрешение всех неразрешённых глобальных ссылок между загруженными модулями.
4. Сохранение всех отлинкованных (соединённых) загруженных модулей в едином исполняемом (машинные коды) файле с расширением .COM.
5. Создание таблицы символов (файл с расширением .SYM).

Параметры запуска:

```
A>L80 <CR>      или      A>L80 file1.ext,file2.ext,...,fileN.ext/E <CR>
*█
```

Ключи программы LINK-80:

/G	Передача управления сформированной программе после того, как закончена её сборка, затем возврат в CP/M после завершения программы. Можно указать стартовый адрес запуска программы: /G:nnnn, где nnnn — 16-чное число (адрес запуска).
/E	Завершение работы L80 и возврат в CP/M после выполнения заданных действий. Можно указать стартовый адрес запуска программы: /E:nnnn, где nnnn — 16-чное число (адрес запуска). После завершения выполнения программы (при определенном адресе запуска) управление передается CP/M.
/N	Все предварительно загруженные программы и подпрограммы должны быть сохранены в файле, имя которого предшествует этому параметру. Другая форма: /N:P — в файл записывается только содержимое области транзитных программ (ТРА). Если этого ключа нет, то выходного файла (с расширением .COM) не создается.

/P/R/D - /P устанавливает начальный адрес сформированной программы и области данных; /D - начальный адрес только области данных (если /P используется совместно с /D, то он указывает только начальный адрес программы). /R используется для возврата L80 в начальное состояние.

/S - Указывает, что файл сразу после него является библиотечным. L80 будет просматривать библиотечные файлы, созданные программой LIB80 и искать те модули, которые могли быть, но еще не использованы в процессе сборки.

/U/M - /U печатает неопределенные внешние имена; /M - все внешние ссылки.

/O/H - Установка системы счисления: /O - 8-чная, /H - 16-чная.

/X/Y - /X создает вместо исполняемого машинного кода результирующий файл с непеременяемым 16-чным объектным кодом (т.е. вместо файла .COM файл .HEX). /Y создает таблицу символов (файл .SYM), которая используется при отладке программы (/Y применяется только совместно с /E).

Примечание:

При соединении с ассемблерными программами последние должны ассемблироваться без окончания: .END <label>, т.к. <label> при этом считается начальным адресом всей программы. Поэтому возникает конфликт.

A>L80 PROG,MYASM,PROG/N/E

PROG.REL - создается компиляторами BASIC, Си и др.

MYASM.REL - создается макроассемблером M80.

PROG.COM - итоговый файл работы L80.

После завершения работы L80 выдает следующую информацию:

```
DATA <prog-start> <prog-end> <bytes>
<free-bytes> FREE BYTES
<start-adr> <prog-end> <num-of-pages>
```

где: <prog-start> - 16-чный адрес начала программы;
<prog-end> - 16-чный адрес конца программы;
<bytes> - 10-чное число байт в программе;
<free-bytes> - объем свободной (оставшейся) памяти;
<start-adr> - 16-чный адрес запуска (не всегда равен <prog-start>);
<num-of-pages> - 10-чное число страниц по 256 байт в программе.

LIB-80

Управление библиотечными файлами, содержащими произвольное число модулей в перемещаемом объектном коде, созданных компиляторами фирмы Microsoft: BASIC, Си, Macro-80 и др. (LIB80.COM)

Функции программы:

1. Объединение (группирование) различных программ на ассемблере, которые являются подпрограммами для трансляторов.
2. Создание библиотек времени исполнения (runtime), содержащих специфические модули, необходимые в процессе исполнения программ.

Параметры запуска:

```
A>LIB80 <CR>      или      A>LIB80 имена-файлов/ключи <CR>
*■
```

Создание библиотечного файла:

```
A>LIB80 filelib=file1,file2,...,fileN
```

где: filelib - имя создаваемого библиотечного файла (по умолчанию .REL, можно .LIB);
file1,file2,...,fileN - список имен файлов (только перемещаемых объектных), входящих в библиотеку (по умолчанию .REL).

Примечание: После каждого file? можно перечислить имена модулей, входящих в исходный файл: ... ,file3<module1,module3,...,moduleN>,file4...

Ключи программы LIB-80:

(обычно ставятся после имени файла, к которому они относятся)

/E - Завершение работы LIB80 (в режиме командных строк) и возврат в CP/M.

Используется только при создании нового библиотечного файла или при изменении существующего. В остальных случаях для выхода используют ^C (реинициализация системы). Это важно, т.к. /E переименовывает создаваемый файл .LIB в .REL и уничтожает предыдущую версию. Если /E использовать с существующим файлом, и этот файл не обновлен, то он будет удален. Этого не будет, если рядом с исходным указан результирующий файл.

/R - Изменение расширения имени обрабатываемого файла .LIB на .REL. Нужны те же меры предосторожности, что и при /E. Использовать только при создании библиотечного файла. Выполняет то же, что и /E, но не выходит в CP/M, а попадает в командный режим. Применяется, если, завершив обработку текущего файла, надо продолжить работу с LIB80.

/L - Выдача на экран списка всех модулей, содержащихся в указанном файле, и определение всех внешних имен, имеющих в модуле.

/U - Выдача на экран списка всех неопределенных внешних имен, найденных при однократном просмотре библиотечного файла. Если в библиотечном файле какой-либо модуль содержит внешнее имя, которое относится к предыдущему модулю, то /U выдает это имя на экран.

/C - Отмена всех введенных ранее команд, без завершения работы LIB80. Создаваемый библиотечный файл уничтожается и программа начинает работу заново. Полезен, если определен некорректный модуль или неправильно введена последовательность модулей.

/O - Установка 8-чной системы счисления. Используется совместно с /L.

/H - Установка 16-чной системы счисления. Используется после /O, т.к. 16-чная система устанавливается по умолчанию.

CREF-80

Построение отчета о перекрестных ссылках для программ на языке Ассемблера для ОС CP/M (CREF80.COM).

Функция программы:

Обработка специального файла печати, созданного макроассемблером M80.COM для получения списка межмодульных ссылок и точек определения внешних имен. Отчёт используется далее при отладке. В итоге создается файл с нумерацией строк и таблицей ссылок с номерами строк, где обнаружен каждый символ. В каждой строке флагом «#» отмечается символическое имя, которое входит в нее в качестве первой лексемы. Для каждого символического имени в отчете представлено его значение, назначенное ему M80.

Параметры запуска:

```
A>CREF80 filename.ext=filename
```

где: filename.ext - имя результирующего файла (расширение по умолчанию .LST, можно назначать вывод на экран - TTY: и печать - LST:);
filename - имя исходного файла с расширением .CRF, созданного M80.

MBASIC

Интерпретатор языка BASIC (MBASIC.COM).

Параметры запуска:

```
A>MBASIC [filename.ext][/ключи]
```

где: filename.ext - имя BASIC файла (расширение по умолчанию .BAS).

Ключи программы MBASIC:

/F:nnnn - максимальное число файлов с данными, которое может быть открыто в BASIC программе (если отсутствует, то nnnn равно 3).
/M:nnnn - максимальный объем оперативной памяти, который доступен MBASIC, резервируется область памяти для программ на Ассемблере Intel 8080, которая находится перед интерпретатором.
/S:nnnn - максимальный объем записи для файла прямого доступа; nnnn - 10-чное число, количество байт в одной записи (если отсутствует, то этот объем равен 128 байтам).

BASCOM

[Компилятор языка BASIC](#) (BASCOM.COM).

Параметры запуска:

```
A>BASCOM file1.ext,file2.ext=file3.ext
```

где: file1.ext - результирующий файл с промежуточным кодом (расширение по умолчанию .REL);
file2.ext - файл печати (расширение по умолчанию .LST, можно назначить вывод на экран - TTY: или на печать - LST:);
file3.ext - исходный файл с текстом программы на BASIC'e (расширение по умолчанию .BAS).

Ключи программы BASCOM:

/C - ослабление ограничений на нумерацию строк (не применять совместно с /4);
/D - формировать средства отладки, используемые затем для обнаружения ошибок при выполнении программы;

/E - в исходной программе содержатся операторы "ON ERROR GOTO" с командой "RESUME <номер строки>";

/N - отмена вывода результатов ассемблирования объектного кода в файл печати;

/O - замена библиотеки времени исполнения BASLIB.REL на OBSLIB.REL, которая будет основной библиотекой при работе LINK-80, когда этой программе задан параметр /E или /G;

/S - запись строк в кавычках в результирующий файл с объектным кодом (.REL), а не в область данных ОП;

/T - применение соглашений BASIC 'а, поддерживаемого интерпретатором BASIC-80 версии 4.51 Microsoft (не применять совместно с /C);

/X - в исходной программе есть операторы "ON ERROR GOTO" с командой "RESUME", "RESUME 0", "RESUME NEXT";

/Z - использовать по возможности коды операций микропроцессора Z-80;

/4 - применение синтаксиса BASIC 'а, поддерживаемого интерпретатором BASIC-80 версии 4.51 Microsoft (не применять совместно с /C).

S-BUG

Инструментальное средство отладки для микропроцессора Z80 (S-BUG.COM).

Параметры запуска:

S-BUG [filename.ext] - загрузить программу S-BUG и (если присутствует) файл с отлаживаемой программой "filename.ext" в память.

Команды программы S-BUG: (см. команды DDT и ZSID)

```
? Help
A Assemble A [address]
C Trace over CALL C[N] [count]
D Dump memory D [addr] [addr]
E Format FCB E string
F Fill in memory F addr addr dd
G Go user program G [addr] [/addr[(count)] [addr[(count)]]...]
H Hex math H expr [expr]
I Input I [port]
IO Input/Output IO [port]
J Simple Memory check J addr addr
JS Speed Memory Test JS addr addr
K Set permanent break point K addr[(count)] [addr[(count)]]...
Display permanent break point K
KX Reset permanent break point KX [addr]...
L Disassemble L [addr] [addr]
M Move memory to memory M addr addr addr
N Search in memory N addr addr bb [bb]...
O Output O [port] data
P Execute macro command P
Define macro command P command[:command]...
PX Reset macro command PX
R Read from Disk R [offset]
S Substitute memory S [addr]
T Trace user program T[N] [count]
V Verify memory V addr addr addr
W Write to Disk W addr addr
X Display user register X
Xr Examine user register
r= F A B C D E H L BC DE HL S P X Y SP PC IX IY
F' A' B' C' D' E' H' L' BC' DE' HL'
Y All symbol display Y
Display symbol selective Y symbol(with wild card chr)
YR Append symbol from disk YR file-name[.TYPE]
YS Define symbol YS symbol
YX Kill symbol YX [symbol]
```

ZAE Abort entry Enable	ZAE
ZAD Abort entry Disable	ZAD

[cpm_utils.msx](#)

[cpm_utils.zip](#)

Сравнительная характеристика файловых структур операционных систем CP/M, MSX и MS DOS



М.Г.Эпиктетов — Сравнительная характеристика файловых структур операционных систем [CP/M](#), [MSX-DOS](#) и [MS DOS](#).

Структура диска [CP/M](#):

Системные дорожки	← при загрузке в память читается первый сектор файловая система эти дорожки не трогает!
Область оглавления	← начало файловой системы на диске
Область данных	
...	

Имеется описатель диска (Disk Parameter Block), который полностью определяет физический формат дискеты (количество секторов, системных дорожек, элементов оглавления и пр.), однако его положение в системной области не определено — BIOS при работе с файловой системой должен получить его от вызываемой программы.

Весь диск, кроме системных дорожек, разбит на группы — последовательности физических секторов размером 128×2^{11} байт (от 1Кб до 16Кб, на Корвете — 2Кб). Группа является неделимым квантом файловой системы (аналогично кластеру в [MS DOS](#)).

Оглавление диска имеет фиксированный размер (на Корвете — 2 группы, на 128 элементов). На файл может выделяться несколько элементов каталога (экстентов), каждый размером по 32 байта:

Usr	Имя файла + расширение	Ext	1	2	Размер
Номера групп, занимаемых файлом (на 16Кб)					

где

- Usr — код пользователя (USER) от 0 до 15. Значение поля, равное 0E5H означает, что файл удалён.
- Имя... — 11 байт, задающих имя и расширение файла в разобранном виде (без точки, дополненное пробелами).
В старшем бите трёх символов расширения указываются атрибуты файла (\$SYS, \$R/O и ?...).
- Ext — номер экстента. Поскольку в директории есть место описание положения только 16Кб, для более длинных файлов заводится несколько элементов каталога, отличающихся номером экстента.
- 1, 2 — два байта зарезервировано для системных нужд.
- Размер — размер файла в 128-байтовых записях.

В зависимости от размера диска на номер группы может отводиться разное число байт. На Корвете номер группы занимает 2 байта (таким образом, один экстент содержит 8 групп, как и было обещано).

(?) имеется предположение, что директорию можно свободно двигать в пределах первых 16 групп, но это не задокументировано и не проверено экспериментально (в DPB имеется 16 бит, в которых положение оглавления указано единицами, но обычно в 1 установлено несколько старших битов).

Структура диска [MSX-DOS](#) и [MS DOS](#):

Boot sector	← при загрузке читается в память, содержит программу начальной загрузки и параметры, описывающие физический формат дискеты: размеры сектора и кластера, количество зарезервированных секторов и FAT'ов, корневой директории
Reserved sectors	← могут отсутствовать
FAT N 1	
FAT N 2	← может отсутствовать
...	
Root directory	
Data area	
...	

MSX-DOS = **MS DOS**, прошедший хорошую «предпродажную подготовку»: из **MS DOS** убрали зарезервированные сектора, поддиректории, количество FAT и размер директории фиксированны, и т.д.

Формат загрузочного сектора:

```

00:  jmp  начальная загрузка ; команда процессора 8086 !
03:  db   "YD-640  "          ; ASCII string of OEM name
0B:  dw   Sector_size         ; размер сектора (в байтах)
0D:  db   Cluster_size        ; размер кластера (в секторах)
0E:  dw   Res_sect            ;
10:  db   Num_FATs            ; количество FAT'ов
11:  dw   Root_lenght         ; размер корневой директории
13:  dw   Total_sectors       ; (количество файлов)
15:  db   Media_descr         ; описание носителя
16:  dw   Sect_FATs           ; размер FAT'ов (в секторах)
18:  dw   Sect_track          ; кол-во секторов на дорожке
1A:  dw   Num_heads           ;
1C:  dw   Num_hidden          ; кол-во зарезервированных секторов на диске.
1E:  программа начальной загрузки

```

FAT (File Allocation Table) содержит описание того, как расположены файлы на диске. В области данных физические сектора объединены в кластеры, их нумерация начинается (почему? то) с 002, поэтому первые 3 или 4 байта FAT свободны (в первом байте лежит media descriptor). На каждый кластер (неделимый квант при записи файла, на MSX — 2 сектора, 1K6) диска отводится 12 или 16 бит (число бит определяется автоматически в зависимости от количества кластеров на диске), в которых содержится:

- 0 — пустой кластер,
- -9 — bad cluster,
- -1 — конец цепочки (последний кластер файла),
- иначе — номер следующего кластера в цепочке.

Корневая директория (в MSX других нет) содержит по 32 байта на каждый файл (независимо от длины) в следующем формате:

Имя файла+расширение	Атрибуты	Дата	Время	First	Размер файла
----------------------	----------	------	-------	-------	--------------

где

- Имя... — 11 байт, задающих имя и расширение файла в разобранном виде (без точки, дополненное пробелами).
Первый байт, равный 0E5h означает, что файл удалён из директории.
- Атрибуты 1 байт:
 - 1 = Read only (нет в MSX)
 - 2 = Hidden
 - 4 = System
 - 8 = Volume label entry
 - 16 = Subdirectory entry
 - 32 = Archive

- Дата/Время создания файла (по 2 байта)
- Зарезервировано (на MSX всегда лежат нули)
- First — номер первого кластера файла (начало цепочки для данного файла) — 2 байта.
- Размер — размер файла в символах (4 байта => максимально большой файл может занимать 4Мб-1).

В системе **MS DOS** можно заводить поддиректории — обычные (с точки зрения файловой системы) файлы, имеющие формат корневой директории, но неограниченный размер.

Недостатки **CP/M** и достоинства **MS DOS**:

1. В стандарте **CP/M** не описано, в каком месте системной области содержится информация о формате диска. Из-за этого диск, записанный на Корвете в Микродосе нельзя прочесть стандартными средствами в **CP/M 2.2** (у них разное число системных дорожек).
С другой стороны, диски, записанные на MSX в **MSX-DOS** легко читаются и на APRICOTe и на ATARI.
2. Из-за того, что **CP/M** не поддерживает на диске таблицу свободного места (она каждый раз строится в памяти), становится невозможной поддержка поддиректорий (иначе расходы на формирование таблицы станут слишком велики — надо будет просканировать несколько файлов, расположенных в разных местах диска). Таким образом, **CP/M** рассчитана на работу только с гибкими дисками.
3. Надёжность **CP/M**, судя по всему, ниже — ошибка в директории ведёт к потере файла. В **MS DOS** фатальной является только ошибка в FAT (после ошибки в директории потеряются имена файлов и время создания, все остальное можно восстановить), но они могут храниться в нескольких экземплярах и постоянно сравниваться.
4. Мелкие достоинства **MS DOS**: размер файла в байтах, а не в записях; имеется поле даты создания файла (это очень полезно, например, для программы Make).

Примечание 1

128 байт в данном контексте это размер логического сектора, принятый в **CP/M** как стандартный размер минимального блока информации.

Размер файлов в **CP/M** всегда кратен числу занятых им логических секторов.

В **CP/M** не задаётся какой либо стандарт на структуру данных, которые хранятся на физических носителях, кроме одного — кратность 128 байтам.

Размер физических секторов определяется свойствами и аппаратными особенностями устройств, применяемых для хранения информации.

Если вспомнить о накопителях на гибких магнитных дисках, то размер физических секторов на них всегда был кратен 128 байтам и был равен 128, 256, 512 или 1024 байта.

Кроме размера физического сектора в **CP/M** задаётся ещё размер кластера — блока физических секторов — на которые разбивается весь объём физического носителя.

Минимальный размер кластера равен объёму физического сектора, но при большом объёме носителя увеличивается, из-за ограничений максимального размера информации хранимой на устройстве.

— Камиль Каримов (caro)

[compare_fs_cpm_msx_ms-dos.msx](#)

Краткое описание входных точек **CP/M**

CP/M находится в слоте 83h. Для запуска **CP/M** необходимо включить в программу фрагмент:

```
;-----
rst    30h    ;Запуск CP/M
defb    83h
defw    405Ah
;-----
```

При запуске **CP/M** переносит себя с 4200h на 00C4h и запускается с адреса 16C4h, по которому находится таблица JP.

Кроме того, дубль CP/M переносится с 4200h на C400h, причем все обращения идут именно туда.

Адреса подпрограмм, к которым обращается CP/M при прерываниях:

5B1Bh	SETADR	Установка адресов для прерываний	
DA03h	CALL0	Адрес для	JP 0
CC06h	CALL5		JP 5
DA45h	CALLC		JP 0Ch
DA4Bh	CALL14		JP 14h
DA51h	CALL1C		JP 1Ch
DA57h	CALL24		JP 24h
DB47h	CALL30		JP 30h

Таблица адресов BDOS находится по адресу CC47h в формате:

```
Low byte ; High byte
Low byte ; High byte
.....
```

[cpm_entry_points.msx](#)

ZCPR2

1 дискета × 720 Кбайт

ZCPR (the Z80 Command Processor Replacement) — это альтернативный командный процессор для CP/M. Он обладает некоторыми преимуществами в сравнении со штатным CCP.

ZCPR2 был адаптирован Камилом Каримовым (Caro) для работы на MSX под CP/M 2.2 в феврале 2022 года.

Игры

Этот [диск с играми](#) появился тут благодаря усилиям **Камиля Каримова(Caro)** и **ATroubleshooter**.

 [Открыть диск с играми в WebMSX](#)

-  [CatChum](#)
-  [Ladder](#)
-  [Zork](#)
-  [Bowling](#)
-  [Gorilla](#)
-  [Hitchhiker's Guide to the Galaxy](#)

Ссылки

 [Встроенная CP/M система в ученические Ямахи КУВТ2](#)

 [Диск с надписью CP/M который шел в комплекте, сообщение #24](#)

 [CP/M на MSX](#)

¹⁾

Смотри [Примечание 1](#)

https://sysadminmosaic.ru/msx/cp_m/cp_m

2023-12-01 23:37

