;; Вызов процедуры sub\_2 (запрет прерываний)

020A:0746 E8 0070 call sub\_2 ; (07B9)

;; Сохранение регистров ES, DS, AX, DX

020A:0749 06 push es

020A:074A 1E push ds

020A:074B 50 push ax

020A:074C 52 push dx

;; В DS помещаем начало области данных BIOS (через буфер AX)

020A:074D B8 0040 mov ax,40h

020A:0750 8E D8 mov ds,ax

;; В ES помещаем адрес начала таблицы векторов прерывания

020A:0752 33 C0 xor ax,ax ; Zero register

020A:0754 8E C0 mov es,ax

;; Инкремент счетчика таймера

;; Инкремент младшей части счётчика таймера

020A:0756 FF 06 006C inc word ptr ds:[6Ch] ; (0040:006C=0A808h)

; 0040:006C = 046С адрес 4байтовой переменной ,

; располагающейся в области данных BIOS это счетчик таймера .

;; Если младшая часть счетчика времени !=0 переходим на loc\_16

; jnz (ZF = 0)

020A:075A 75 04 jnz loc\_16 ; Jump if not zero

;; Инкремент старшей части счётчика таймера

020A:075C FF 06 006E inc word ptr ds:[6Eh] ; (0040:006E=8)

; Если счетчик равен 0 , то увеличиваем часы , т. е. прошел час.

;( 0040:006E это часы)

;; Сброс счётчика таймера и выставление флага окончания суток

; Если час не прошел , то сравниваем

; 0040:006E с 24 ( это часы 18h == 24 )

020A:0760 loc\_16:

020A:0760 83 3E 006E 18 cmp word ptr ds:[6Eh],18h

; (0040:006E=8) Если еще не 24 , то переходим на loc\_17

020A:0765 75 15 jne loc\_17 ; Jump if not equal

020A:0767 81 3E 006C 00B0 cmp word ptr ds:[6Ch],0B0h

; (0040:006C=0A808h) если в младшей части счётчика 00B0h (176)

; значит в счётчике записано число 1573040 тиков == 24 часа

; Если != 176 , то переходим на loc\_17

; Обнуляем счетчик ( если прошел день )

020A:076D 75 0D jne loc\_17 ; Jump if not equal

020A:076F A3 006E mov word ptr ds:[6Eh],ax

; (0040:006E=8) обнуляем счётчик (старшая часть)

020A:0772 A3 006C mov word ptr ds:[6Ch],ax

; (0040:006C=0A808h) (младшая часть)

;; В ячейку 0040:0070 записываем единицу

; (Для фиксации о том , что новый день наступил )

020A:0775 C6 06 0070 01 mov byte ptr ds:[70h],1 ; (0040:0070=0)

; В младший байт регистра ax заносим 8

; ( т.к. ax до этого был равен 0 => 0 or 8 == 8 )

020A:077A 0C 08 or al,8

;; Декремент счетчика до отключения моторчика дисковода

020A:077C loc\_17:

; Если новый день не наступил , то

; Записываем регистр ax в стек.

; (Он м. б . равен 0 или 8 , в зависимости от того ,

; наступил новый день или нет )

020A:077C 50 push ax

; Ячейка с адресом 0000:0440 h с одержит время ,

; оставшееся до выключения двигателя .

; Декрементируем это время .

020A:077D FE 0E 0040 dec byte ptr ds:[40h] ; (0040:0040=8Dh)

; Если еще не равно нулю , то переходим на loc\_18

020A:0781 75 0B jnz loc\_18 ; Jump if not zero

; Если равено 0 , то двигатель НГМД отключается.

; Отправка сигнала отключения моторчика.

; Сброс флага отключания моторчика дисковода

020A:0783 80 26 003F F0 and byte ptr ds:[3Fh],0F0h ; (0040:003F=0)

020A:0788 B0 0C mov al,0Ch

020A:078A BA 03F2 mov dx,3F2h

; Порт 3F2 адрес порта цифрового упарвления (тип вывод) .

; НГМД накопитель на гибких магнитных дисках

; Порт 3F2h работает только на запись , это порт вывода .

; Мы отправляем в этот порт 0C (1100) .

; 2 бит поднят разрешение работы контроллера

; 3 бит поднят разрешение прерываний и прямого доступа к памяти

; 4..7 биты значение 1 в каждом разряде вызывает включение

; соответствующего двигателя НГМД

; (Инф. https://frolov-lib.ru/books/bsp/v19/ch1\_4.html )

; Инструкция OUT выводит данные из регистра AL или AX (ИСТОЧНИК)

; в порт ввода/вывода .

; Номер порта должен быть указан в ПРИEМНИКЕ.

020A:078D EE out dx,al ; port 3F2h, dsk0 contrl output

;; Проверка, установлен ли PF(parity flag), т.е. разрешен ли ответ на маскируемые прерывания

020A:078E loc\_18:

; Если счетчик таймера не равен нулю , то

; Возвращаем в ax содержимое , которое раньше положили .

020A:078E 58 pop ax

;; Проверяем флаг PF по адресу 0040:0314

; (0100, поднят 2 бит, отвечает за флаг PF, флаг четности)

020A:078F F7 06 0314 0004 test word ptr ds:[314h],4 ; (0040:0314=3200h)

;; если вызов маскируемых прерываний разрешен, переход к прямому вызову int 1Ch (в loc\_19)

020A:0795 75 0C jnz loc\_19 ; Jump if not zero

; LAHF перемещает младший байт регистра флагов EFLAGS в регистр AH.

020A:0797 9F lahf ; Load ah from flags

020A:0798 86 E0 xchg ah,al ; Обмен

020A:079A 50 push ax ; Записываем AX в стек

;; иначе, косвенный вызов 1Сh - как процедуры командой call и переход к loc\_20

; (1C \* 4 = 70h )

020A:079B 26: FF 1E 0070 call dword ptr es:[70h] ; (0000:0070=6ADh)

020A:07A0 EB 03 jmp short loc\_20 ; (07A5)

020A:07A2 90 nop

;; прямой вызов пользовательского прерывания по таймеру

020A:07A3 loc\_19:

020A:07A3 CD 1C int 1Ch ; Timer break (call each 18.2ms)

;; после инициализации системы вектор INT 1Ch указывает на команду IRET

; сброс контроллера прерываний

020A:07A5 loc\_20:

020A:07A5 E8 0011 call sub\_2 ; (07B9)

; Сброс контроллера прерываний (mov al , 20h ; out 20h , a l ) из методички .

; Не обходимо отметить , что прерывание int 1Ch вызывается обработчиком

; прерывания int 8h

; до сброса контроллера прерывания , поэтому во время выполнения

; прерывания int 1Ch все аппаратные прерывания з апрещены.

; В частности , запрещены прерывания от клавиатуры.

020A:07A8 B0 20 mov al,20h ; ' '

020A:07AA E6 20 out 20h,al ; port 20h, 8259-1 int command

; al = 20h, end of interrupt

; (указывает на завершение обработки прерывания для данного прерывания)

; посылаем в порт контроллера прерываний команду завершить прерывание

; Инструкция OUT выводит данные из регистра AL или AX (ИСТОЧНИК) в порт ввода-вывода

;; восстановление значений регистров

020A:07AC 5A pop dx

020A:07AD 58 pop ax

020A:07AE 1F pop ds

020A:07AF 07 pop es

;; прыжок в адрес 020A:064C

020A:07B0 E9 FE99 jmp loc\_1 ; (064C)

; ---

020A:064C 1E push ds

020A:064D 50 push ax

; ---

020A:06AA 58 pop ax

020A:06AB 1F pop ds

020A:06AC CF iret ; Interrupt return

sub\_2 proc near

;; Сохранение значений регистров

020A:07B9 1E push ds

020A:07BA 50 push ax

;; В DS помещаем начало области данных BIOS

020A:07BB B8 0040 mov ax,40h

020A:07BE 8E D8 mov ds,ax

;; Сохранение значений флагов (значение флагового регистра в регистр АН)

020A:07C0 9F lahf ; Load ah from flags

020A:07C1 F7 06 0314 2400 test word ptr ds:[314h],2400h ; (0040:0314=3200h)

; Команда TEST логическое и без изменения операнда

; (Меняются только флаги) .

; 2400 = 10010000000000. Поднят ли флаг 10ый или 13ый?

; 10 DF Direction Flag Флаг направления .

; Контролирует поведение команд обработки строк .

; Если установлен в 1 , то строки

; обрабатываются в сторону уменьшения адресов ,

; если сброшен в 0 , то наоборот .

; 12 и 13 IOPL I/O Priilege Level

; Уровень приоритета ввода/вывода .

; Если не равно 0 переходим на loc\_22

020A:07C7 75 0C jnz loc\_22 ; Jump if not zero

;; Сброс флага IF

; На все время выполнения команды, снабженной таким префиксом, будет заб-

; локирована шина данных, и если в системе присутствует другой процессор, он не

; сможет обращаться к памяти, пока не закончится выполнение команды с префик-

; сом LOCK.

; LOCK - делаеи следующую команду неделимой.

; and 2 раза обращается к памяти. 1 раз он считывает значение по адресу 0040:0314

; Затем он изменяет его и еще раз обращаяется к памяти на запись.

; Мы делаем ее неделимой, чтобы в этот промежуток, когда мы выполняем непосредственно

; Саму логическую операцию, никто не смог влезть в этот участок памяти (мы его как раз блокируем).

020A:07C9 F0> 81 26 0314 FDFF lock and word ptr ds:[314h],0FDFFh ; (0040:0314=3200h)

;; Восстановление значений флагов

020A:07D0 loc\_21:

020A:07D0 9E sahf ; Store ah into flags

; Команда sahf копирует разряды 7, 6, 4, 2 и 0 регистра АН в регистр флагов процессора,

; устанавливая тем самым значения флагов SF, ZF, AF, PF и CF соответственно.

; Команда не имеет операндов.

;; Восстановление значений регистров

020A:07D1 58 pop ax

020A:07D2 1F pop ds

020A:07D3 EB 03 jmp short loc\_23 ; (07D8)

;; Сброс IF, т. е. запрет прерываний с помощью команды cli

; CLI сбрасывает флаг IF

; Если сбросить этот флаг в 0, то процессор перестанет обрабатывать прерывания от внешних устройств.

; Обычно его сбрасывают на короткое время для выполнения критических участков программы.

; (маскируемые - прерывания, которые можно запрещать установкой соответствующих битов в регистре маскирования прерываний)

020A:07D5 loc\_22:

020A:07D5 FA cli ; Disable interrupts

020A:07D6 EB F8 jmp short loc\_21 ; (07D0)

;; Выход из программы

020A:07D8 loc\_23:

020A:07D8 C3 retn

sub\_2 endp