

Министерство образования Республики Беларусь
Учреждение образования
«БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
ИНФОРМАТИКИ И РАДИОЭЛЕКТРОНИКИ»
Кафедра электронно-вычислительных машин

Лабораторная работа №6
“Функции BIOS”

Выполнил:
студент группы 724402
Чернявский Я.А.

Проверил:
к.т.н., доцент Селезнёв И.Л.

Минск
2019

1. Цель работы:

Изучить функции BIOS и организацию ввода-вывода через прерывания BIOS.

2. Описание алгоритма:

1. Вызвать прерывание INT 11h для получения конфигурации компьютера.
2. Вызвать прерывание INT 12h для получения объема основной оперативной памяти компьютера.
3. Вызвать прерывание INT 16h для ввода текста.
4. Вызвать прерывание INT 1Ah для считывания и установки даты и времени с часов реального времени.

3. Теоретические сведения:

Системный модуль ROM BIOS (System ROM BIOS) обеспечивает программную поддержку стандартных устройств PC, конфигурирование аппаратных средств, их диагностику и вызов загрузчика операционной системы. Системный модуль ROM BIOS в значительной степени привязан к конкретной реализации системной платы, поскольку именно ему приходится программировать все микросхемы чипсета системной платы.

Функции BIOS разделяются на следующие группы:

1. Инициализация и начальное тестирование аппаратных средств – POST.

Самоконтроль при включении питания (Power-On Self-Test - POST).

POST представляет собой встроенную диагностическую программу, которая проверяет наличие и правильность функционирования аппаратных средств до производства BIOS фактической загрузки. В ходе загрузки производится и дополнительное тестирование, например тест памяти, ход которого отображается на экране;

2. Настройка и конфигурирование аппаратных средств и системных ресурсов – CMOS Setup;

3. Автоматическое распределение системных ресурсов - PnP BIOS

BIOS является одним из трех компонентов системы, взаимодействие которых необходимо для реализации в PC средств Plug and Play (PnP). Здесь аппаратные средства PC, BIOS и операционная система, работая в тесном взаимодействии, автоматически идентифицируют и конфигурируют использование ресурсов аппаратными устройствами. Технология PnP сокращает число конфликтов ресурсов, устраняет использование перемычек и ручную настройку драйверов. BIOS играет при этом ключевую роль, так как фактически именно он

идентифицирует и конфигурирует карты расширения, а также передает информацию о конфигурировании операционной системе;

4. Идентификация и конфигурирование устройств PCI - PCI BIOS

Обычно шина большей частью PCI настраивается автоматически, но параметры этой секции могут пригодиться, когда необходимо реализовать конкретное поведение шины или карт расширения. Для настройки параметров данной секции требуются глубокие знания аппаратных средств компьютера, поэтому для большинства пользователей обычно подходят принимаемые по умолчанию значения параметров. Рекомендуется также иметь в компьютере средства автоматического конфигурирования;

5. Начальная загрузка (первый этап загрузки операционной системы) - Bootstrap Loader;

6. Обслуживание аппаратных прерываний от системных устройств (таймера, клавиатуры, дисков) –BIOS Hardware Interrupts;

7. Отработка базовых функций программных обращений (сервисов) к системным устройствам - ROM BIOS Services;

8. Поддержка управляемости конфигурированием - DMI BIOS;

9. Поддержка управления энергопотреблением и автоматического конфигурирования - APM и ACPI BIOS.

Все эти функции (или их часть) исполняет системный модуль BIOS, хранящийся в микросхеме ПЗУ или флэш-памяти на системной плате. Большинство сервисных функций выполняется в 16-битном режиме, хотя некоторые новые функции могут иметь и альтернативные вызовы для 32-битного исполнения.

Системный модуль BIOS должен обслуживать по вышеуказанным функциям все компоненты, установленные на системной плате: процессор, контроллер памяти (ОЗУ и кэш), стандартные архитектурные компоненты (контроллеры прерываний и DMA, системный таймер, системный порт, CMOS RTC), контроллер клавиатуры, а также набор стандартных периферийных контроллеров и адаптеров, даже если они и не установлены на системной плате. В этот набор входят графические адаптеры CGA и MDA, порты COM и LPT, контроллер НГМД, диски АТА (теперь уже обязательно двух каналов). Если на системной плате установлены дополнительные компоненты, например контроллер SCSI, графический адаптер SVGA, адаптер локальной сети, то их поддержка тоже должна быть в системном модуле BIOS.

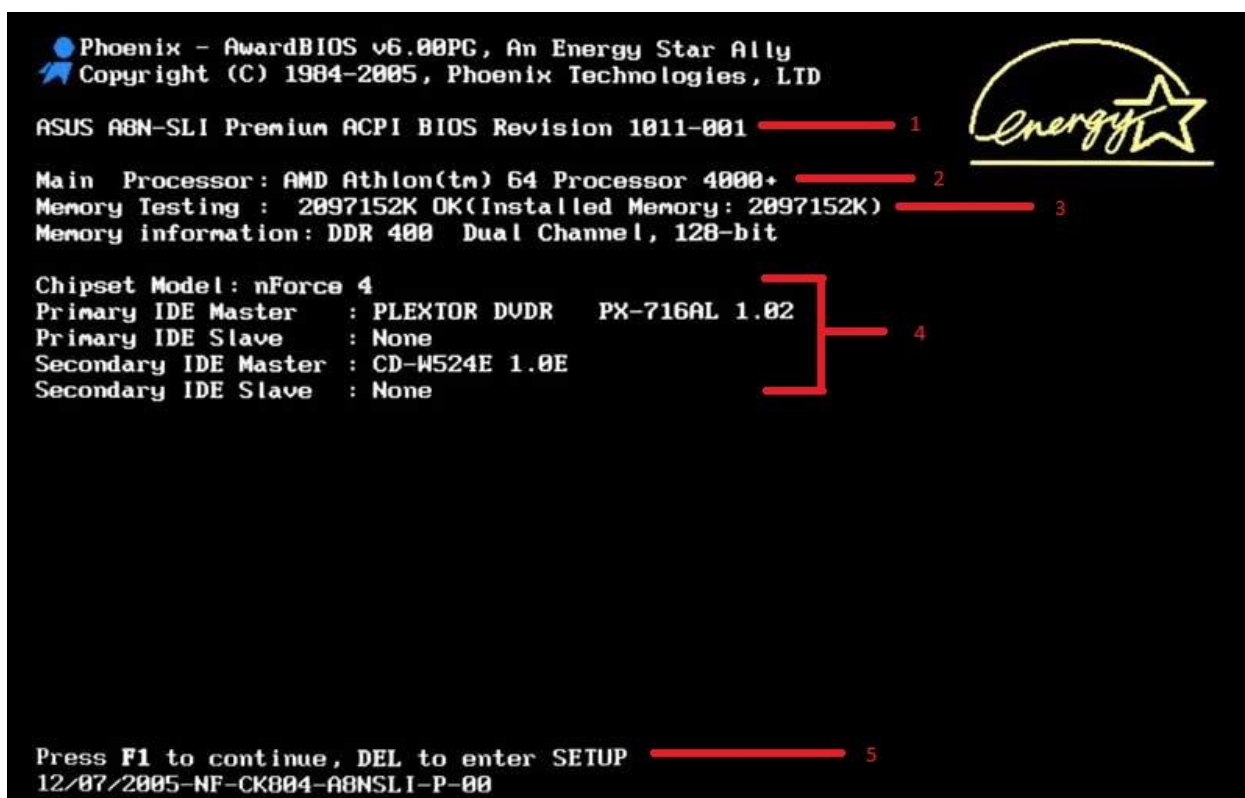


Рис.3.1– Скриншот экрана загрузки

1. Производитель BIOS и номер версии.

Дата BIOS: Дата BIOS помогает определить его возможности, так как даты введения некоторых возможностей хорошо известны.

Серийный номер BIOS: Обычно отображается внизу экрана. Так как BIOS адаптированы под конкретную материнскую плату, серийный номер можно использовать для определения конкретной материнской платы и версии BIOS.

2. Тип процессора (CPU): Обычно это общее семейство процессоров, например "Pentium" или "Pentium Pro" и т.д. Новые BIOS прямо распознают Intel-совместимые процессоры, а старые могут вывести "Pentium", даже когда процессор другой. Иногда BIOS отображает "Pentium-S", показывая, что процессор поддерживает расширенные средства управления мощностью (System Management Mode - SMM). Сейчас эти средства имеются почти во всех современных процессорах.

3. Тип, ёмкость и конфигурация памяти: В большинстве новых PC сообщается, сколько банков памяти обнаружено и какая технология памяти применяется. Например, можно увидеть на экране "EDO DRAM at Bank 1" или "FP: 0" (FPM DRAM) или что-то аналогичное.

4. Накопители IDE/ATA: Для каждого установленного накопителя сообщается его емкость и используемые им режимы обращения. Некоторые

BIOS сообщают также производителя накопителя, а большинство современных BIOS обнаруживает и показывает накопитель IDE (ATAPI) CD-ROM.

5. Клавиша программы настройки: Клавиша или клавиши, которые необходимо нажать для входа в программу настройки BIOS. Обычно это клавиша Del или F2.

Сервисы и другие векторы прерываний BIOS.

При инициализации таблицы прерываний BIOS отвечает за корректное заполнение части векторов, имеющих отношение к аппаратным средствам компьютера и сервисам BIOS. На некоторые из них могут быть просто установлены заглушки: вектор ссылается на код обработчика, содержащего единственную инструкцию возврата из прерывания - I RET. BIOS инициализирует векторы прерываний различных назначений:

Внутренние прерывания:

- ◆ Int 00h — деление на 0;
- ◆ Int 01h — пошаговый режим;
- ◆ Int 03h — точка останова;
- ◆ Int 04h — переполнение;
- ◆ Int 06h — недопустимая команда 286+;
- ◆ Int 07h — вызов отсутствующего математического сопроцессора (Numeric Processor Unit, NPU).

Аппаратные прерывания:

- ◆ Int 02h — немаскируемое прерывание;
- ◆ Int 08h — таймер 8253/8254;
- ◆ Int 09h — клавиатура;
- ◆ Int 0Ah — IRQ2/9;
- ◆ Int 0Bh — IRQ3;
- ◆ Int 0Ch — IRQ4;
- ◆ Int 0Dh — IRQ5;
- ◆ Int 0Eh — IRQ6 (контроллер гибких дисков);
- ◆ Int 0Fh — IRQ7;
- ◆ Int 70h — CMOS-таймер;
- ◆ Int 71h — IRQ9 (перенаправлено на Int 0Ah);
- ◆ Int 72h — IRQ10;
- ◆ Int 73h — IRQ11;
- ◆ Int 74h — IRQ12 (контроллер мыши PS/2);
- ◆ Int 75h — IRQ13 (исключение сопроцессора);

Рис.3.2– Векторы прерываний BIOS

- ◆ Int 76h — IRQ14 (контроллер жестких дисков);
- ◆ Int 77h — IRQ15.

ПРИМЕЧАНИЕ

Прерывания Int 70h–77h имеют место только в АТ.

Функции ROM BIOS (16-битные сервисы):

- ◆ Int 05h (F000:FF54h) — печать экрана;
- ◆ Int 10h — видеосервис (см. 10.6);
- ◆ Int 11h — чтение списка оборудования (слово из BDA 0040:0010h), возвращает в AX:
 - биты 15:14 — число обнаруженных LPT-портов (00 — 0, ..., 11 — 3);
 - бит 13 — резерв;
 - бит 12 — обнаружен игровой адаптер;
 - биты 11:9 — число обнаруженных COM-портов (000 — 0, ..., 111 — 7);
 - бит 8 — наличие контроллера DMA;
 - биты 7:6 — число обнаруженных НГМД (00 — 1, ..., 11 — 4);
 - биты 5:4 — активный видеорежим (00 — резерв, 10 — 80-колоночный цветной, 01 — 40-колоночный цветной, 11 — монохромный);
 - биты 3:2 — размер ОЗУ на системной плате (теперь обычно 00);
 - бит 1 — присутствие математического сопроцессора;
 - бит 0 — присутствие дисководов;
- ◆ Int 12h — размер непрерывной памяти;

5.2. Системный модуль ROM BIOS

171

- ◆ Int 13h — дисковый сервис (блочный ввод-вывод, см. 9.11);
- ◆ Int 14h — обслуживание COM-портов (см. 16.1);

Рис.3.3– Векторы прерываний BIOS

- ◆ Int 15h — АТ-функции (системный сервис, функции определяются значением AH/AX):
 - 00–03h — управление и обмен данными с кассетным магнитофоном (были когда-то и такие «стримеры»!) на старых ПК;
 - 4fh — перехват событий клавиатуры (см. 11.1);
 - 53xxh — сервисы расширенного управления энергопотреблением (APM);
 - 8300h — запуск таймера, устанавливающего флаг в заданной ячейке (см. 4.6);
 - 8301h — сброс того же таймера;
 - 84h — джойстик (см. 11.6);
 - 86h — программируемая задержка (см. 4.6);
 - 87h — перемещение блока расширенной памяти;
 - 88h — получение размера расширенной памяти;
 - 89h — переключение в режим V86;
 - C0h — получение системной конфигурации, при успешном выполнении (CF = 0, AH = 0) ES:BX указывает на таблицу данных конфигурации (табл. 5.3);
 - 80–82h, 85h, 90h, 91h — функции многозадачных ОС (BIOS устанавливает заглушки);
- ◆ Int 16h — клавиатурный ввод-вывод (см. 11.1);
- ◆ Int 17h — обслуживание LPT-портов (см. 15.4);
- ◆ Int 18h — процедура восстановления при неудаче начальной загрузки (прежде — ROM-Basic);
- ◆ Int 19h — начальная загрузка;
- ◆ Int 1Ah — системное время, дата, будильник (см. 4.6) и 16-битные вызовы сервисов PCI (см. 14.7);
- ◆ Int 1Bh — обработчик нажатия клавиш Ctrl+Break;
- ◆ Int 1Ch — процедура User Timer Interrupt, вызываемая обработчиком Int 08h каждые 55 мс; BIOS устанавливает простую заглушку (IRET), но программы могут перехватывать это прерывание; на время отработки процедуры *все аппаратные прерывания запрещены* (кроме NMI);

Рис.3.4– Векторы прерываний BIOS

INT 11h - Получить список оборудования.

Прежде чем пытаться работать с каким-либо устройством ввода/вывода, следует убедиться в том, что оно есть в составе оборудования компьютера. В процессе инициализации тестовые модули, находящиеся в BIOS, динамически определяют состав аппаратного обеспечения машины и записывают конфигурацию системы в специально отведенную для этого ячейку памяти. Программа, вызывая прерывание INT 11h, получает в регистре **AX** содержимое этой ячейки. Каждый бит в слове конфигурации отвечает за соответствующее устройство.

Анализируя слово конфигурации, программа может узнать, входят ли в состав оборудования компьютера дискеты и если входят, то сколько дисководов имеется в наличии, присутствует ли арифметический сопроцессор, какой начальный режим дисплейного адаптера используется, сколько в системе принтеров, адаптеров последовательного интерфейса RS232, подключен ли игровой адаптер (джойстик)?

Обычно прикладная программа не работает сама с аппаратурой, а пользуется услугами операционной системы. При обращении к стандартной аппаратуре через операционную систему программа пользователя получит признак ошибки, если запрашиваемое устройство отсутствует. Программы, составленные на языке программирования Си при использовании библиотеки эмуляции арифметического сопроцессора, сами определяют, имеется сопроцессор или нет, и не пытаются пользоваться отсутствующим устройством. Операции сопроцессора эмулируются центральным процессором, и программа просто работает медленнее. Но если программа обращается непосредственно к портам ввода/вывода отсутствующего устройства, это может привести в лучшем случае к зависанию системы.

INT 12h - Получить размер основной памяти.

Сказанное выше справедливо и по отношению к оперативной памяти. Для работы некоторых программ требуется достаточное количество памяти. Прерывание INT 12h возвращает в регистре **AX** количество имеющихся блоков памяти размером в один килобайт. Анализируя эту величину, программы могут при нехватке памяти либо вывести на экран соответствующее сообщение и отказаться от работы, либо изменить алгоритмы работы, организовав, например, "виртуальную" память на диске или просто записывая в файл промежуточные результаты. Если Ваш компьютер оборудован расширенной памятью (адресное пространство этой

памяти находится выше границы в 1 мегабайт), размер этой памяти в килобайтах можно узнать, вызвав прерывание INT 15h со значением регистра AX, равным 8800h.

INT 16h - Обслуживание клавиатуры.

Обработчик прерывания INT 16h выполняет несколько функций, связанных с обслуживанием клавиатуры. Мы не будем сейчас перечислять эти функции, они будут подробно описаны в главе, посвященной клавиатуре. С помощью функций обслуживания клавиатуры можно выполнить ввод кода нажатой клавиши как с ожиданием нажатия, так и без ожидания. В последнем случае функция сразу после вызова возвращает код нажатой клавиши или признак того, что никакая клавиша не нажималась.

Заметим, что символы, введенные с клавиатуры, помещаются в специальный клавиатурный буфер. Функция ввода символа без ожидания нажатия на клавишу проверяет состояние буфера - есть в нем символы, или нет. Если в буфере есть символы, первый помещенный в буфер символ возвращается программе. Этот символ затем может быть считан функцией ввода с ожиданием нажатия - фактически ожидания при этом не будет.

Для очистки буфера клавиатуры также можно использовать пару описанных выше функций: сначала программа проверяет пуст ли буфер, и, если он не пуст, считывает символ. Считанный символ никуда не помещается (теряется). После считывания символа программа опять проверяет содержимое буфера и так до тех пор, пока клавиатурный буфер не окажется пустым.

Для машин класса не ниже AT обработчик прерывания INT 16h выполняет и другие функции: установку задержки, запись символов в буфер клавиатуры, обслуживание расширенной клавиатуры.

Обслуживание дисковой подсистемы.

Прерывание INT 13h предназначено для обслуживания жестких и флоппи-дисков. Многочисленные функции прерывания INT 13h выполняют все операции по вводу/выводу на диски.

00h - Сброс дисковой системы. Эта функция выполняет установку в исходное состояние всей дисковой системы или выбранного дискового устройства. Используется обычно перед началом работы с устройством.

01h - Получить состояние дисковой системы. Эта функция позволяет проверить результат выполнения предыдущей операции. Если операция

завершилась аварийно, при помощи этой функции можно определить код ошибки.

02h/03h - Чтение/запись секторов. Выполняется чтение секторов в оперативную память компьютера или запись информации из памяти в сектора диска. Сектор задается для выбранных устройства, дорожки и головки. Программа должна также задать количество читаемых/записываемых секторов.

04h - Проверка секторов. Функция проверяет сектора на правильность циклической контрольной суммы, CRC (Cyclic Redundancy Check); записи содержимого секторов в память не происходит.

Другие функции прерывания INT 13h.

Среди других функций прерывания INT 13h - форматирование дорожки, позиционирование головки на заданную дорожку диска, тестирование и предварительная установка диска, запуск диагностики контроллера и многое другое. Описание этих функций мы отложим до глав, посвященных файловой системе. **Вывод на принтер (параллельный порт).**

BIOS содержит простейшую поддержку принтера - три функции прерывания INT 17h. Это функция 01h - инициализация принтера, 02h - опрос состояния принтера и 00h - вывод символа на принтер. Поскольку к персональному компьютеру можно подключить несколько последовательных портов, при обращении к принтеру следует указывать номер порта.

Обслуживание последовательного порта связи

Функции прерывания INT 14h обслуживают порт последовательной передачи данных RS232. С помощью этих функций можно задавать формат и скорость передачи данных, определять состояние портов и, конечно, выполнять побайтную передачу данных.

INT 1Ah – Работа с системными часами.

Этот сервис предоставляет доступ к системным часам. PC BIOS работает со "счетчиком тиков" - числом 55-мс интервалов, прошедших с момента включения или сброса PC. AT BIOS предоставляет также доступ к значениям часов реального времени, которые постоянно обновляются независимо от работы процессора и хранятся в CMOS-памяти компьютера AT.

АН сервис

00H читать часы (счетчик тиков) выход: CX, DX =
счетчик тиков с момента сброса. CX - старшая часть
значения.

AL = 0, если таймер не переполнялся за 24 часа с
момента сброса.

замечание: часы обновляются каждые 1193180/65536 (ў
18.2) тиков в секунду.

	тиков в секунде	18
тиков в минуте	1092	
тиков в часе	65543	
тиков в сутках	1573040	

01H установить часы (счетчик тиков) вход: CX,
DX = счетчик тиков. CX - старшая часть значения.

02H |АТ| читать время из "постоянных" (CMOS) часов
реального времени

выход: CH = часы в коде BCD (пример: CX = 1243H =
12:43)

CL = минуты в коде BCD

DH = секунды в коде BCD выход: CF
= 1, если часы не работают

03H |АТ| установить время на "постоянных" (CMOS) часах
реального времени

вход: CH, CL = часы, минуты в коде BCD

DH = секунды в коде BCD

DL = 1 для опции "единиц светового дня"

```

-----
---
-----
04H |AT| читать дату из "постоянных" (CMOS) часов
реального времени      выход: CH = столетие в коде BCD
(пример: CX = 1987H = 1987)
          CL = год в коде BCD
          DH = месяц в коде BCD (пример: DX = 0312H = 12-е
марта)
          DL = день в коде BCD
выход: CF = 1, если часы не работают
-----
---
-----

```

```

05H |AT| установить дату на "постоянных" (CMOS) часах
реального времени
      вход: CH, CL = столетие, год в коде BCD
          DH, DL = месяц, день в коде BCD
-----
---
-----

```

```

06H |AT| установить сигнал часов реального времени. В
указанное время вызывается      пользовательская программа
по вектору прерывания INT 4aH. Лишь один сигнал      может
быть активен в каждый момент времени.
      вход: CH, CL = часы, минуты в коде BCD
DH      = секунды в коде BCD      выход: CF = 1, если
часы не работают или сигнал уже активен
-----
---
-----

```

```

07H |AT| сбросить сигнал часов реального времени.
это позволяет вам отменить      один сигнал перед
установкой другого.

```

Функции прерывания INT 1Ah обслуживают часы, имеющиеся в каждом компьютере. С их помощью вы можете установить время и дату, опросить текущее состояние часов. Вы можете работать с часами реального времени, которые имеются на машинах класса не ниже AT.

Для АТ можно установить на заданное время "будильник" - в нужный момент будет вызвано прерывание "будильника" с номером 4Ah. Обработчик прерывания INT 4Ah может подать звуковой сигнал или вывести на экран предупреждающее сообщение.

Перезагрузка операционной системы

Вызов прикладной программой прерывания INT 19h приведет к перезагрузке операционной системы.

Системный сервис для машин класса АТ.

Прерывание INT 15h использовалось в компьютерах IBM PC и IBM PC Jr для управления кассетным накопителем на магнитной ленте (функции 0-3). Для машин класса АТ и более высокого класса прерывание INT 15h имеет и другое назначение. С его помощью обслуживается расширенная клавиатура, выполняется программная задержка, задаваемая в микросекундах, обслуживается расширенная память. Кроме того, одна из функций прерывания INT 15h переводит процессор 80286 или 80386 в защищенный режим. Заметим, что вернуть процессор обратно в реальный режим можно только сигналом начального сброса. Это же относится и к арифметическому сопроцессору 80287.

Функция C0h прерывания INT 15h выдает дополнительные сведения о конфигурации аппаратных средств компьютера.

Системный модуль BIOS должен обслуживать по вышеуказанным функциям все компоненты, установленные на системной плате: процессор, контроллер памяти (ОЗУ и кэш), стандартные архитектурные компоненты (контроллеры прерываний и DMA, системный таймер, системный порт, CMOS RTC), контроллер клавиатуры, а также набор стандартных периферийных контроллеров и адаптеров, даже если они и не установлены на системной плате. В этот набор входят графические адаптеры CGA и MDA, порты COM и LPT, контроллер НГМД, диски АТА (теперь уже обязательно двух каналов). Если на системной плате установлены дополнительные компоненты, например, контроллер SCSI, графический адаптер SVGA, адаптер локальной сети, то их поддержка тоже должна быть в системном модуле BIOS.

4. Листинг кода программы:

```
//724402-2 Chernyavsky Y.A.  
//Laboratornaia rabota 6: Funkcii BIOS
```

```
#include <bios.h>
```

```

#include <time.h>
#include <stdlib.h>
#include <stdio.h>
#include <memory.h>
#include <conio.h>
#include <dos.h>

int bcd1bin(char *bcd) //BCD to BIN -> 1 byte
{
return (((*bcd) & 0x0f) + 10 * (((*bcd) & 0xf0) >> 4));
}

int bcd2bin(char *bcd) // BCD to BIN -> 2 byte
{
return (bcd1bin(bcd) + 100 * bcd1bin(bcd + 1));
}

int bin1bcd(int bin, char *bcd) // BIN to BCD -> 1 byte
{
int i;
i = bin / 10;
*bcd = (i << 4) + (bin - (i * 10));
return i;
}

void indidication(void)
{
printf("724402-2 Chernyavsky Y.A.\nLabaratornaia rabota 6 Funkcii BIOS\n");

printf(" 1 - Calling the INT 11h (PC config) \n 2 - Calling the INT 12h (RAM)\n 3 -
Calling the INT 16h (TXT Input)\n 4 - Calling the INT 1Ah (Set data)\n ESC - exit\n");
//1/read list of equipment 2/size of continuous memory 3/keyboard input output 4/time and
data
}

typedef struct _HDWCFG
{
unsigned HddPresent : 1; //0 diskovod
unsigned NpuPresent : 1; //1 matematicheskiy soprotsessor
unsigned AmountOfRAM : 2; //2-3 razmer OZU
unsigned VideoMode : 2; //4-5 aktivnyy videorezhim
unsigned NumberOfdd : 2; //6-7 chislo obnaruzhennykh NGMD

```

```

unsigned DmaPresent : 1; //8 nalichie kontrollera DMA
unsigned NumberOfCom : 3; //9-11 chislo COM-portov
unsigned GamePresent : 1; //12 igrovoy adapter
unsigned JrComPresent : 1; //13 rezerv
unsigned NumberOfLpt : 2; //14-15 chislo LPT-portov
} HDWCFG;

char* tobin(char* str, unsigned char t)
{
    int mask, i;
    for (i = 0, mask = 0x80; mask != 0; mask >>= 1, i++)
    {
        str[i] = t & mask ? '1' : '0';
    }
    str[8] = '\0';
    return str;
}

char* tohex(char* str, unsigned char t)
{
    char hex[16] = { '0', '1', '2', '3', '4', '5', '6', '7', '8', '9', 'A', 'B', 'C', 'D', 'E', 'F' };
    str[2] = '\0';
    str[1] = hex[t & 0xF];
    str[0] = hex[t >> 4];
    return str;
}

void showWord(unsigned int uword)
{
    char high[9];
    char low[9];
    tohex(high, uword >> 8);
    tohex(low, uword & 0x00FF);
    printf("\nAX: hex(%s%s)", high, low);
    tobin(high, uword >> 8);

```

```

tobin(low, uword & 0x00FF);
printf("\tbin(%s%s)\n", high, low);
}

void main()
{
    long hour, minute, second;

    int key, pick;

    union REGS rg;

    HDWCFG HdwCfg;

    unsigned uword;

    char *month_to_text[] =
    { "Jan", "Feb", "Mar", "Apr", "May", "Jun", "Jul", "Aug", "Sep", "Oct", "Nov", "Dec" };

    clrscr();

    indidication();

    while (key != 27)//Poka ne nazhata klavisha ESC
    {
        key = getch();

        if (key == 27) exit(1);

        if (key == 49)//ESli 1, to
        {
            printf("\n\n Calling the INT 11h \n");

            uword = (unsigned int)rg.x.ax; // Poluchaem slovo konfiguratsii i
sohranyaem

            showWord(uword);

            rg.h.ah = 0x0; // Vyzyvaem preryvanie INT 11h dlya polucheniya
            int86(0x11, &rg, &rg); // slova konfiguratsii komp'yutera

            uword = (unsigned int)rg.x.ax; // Poluchaem slovo konfiguratsii i
sohranyaem

            memcpy(&HdwCfg, &uword, 2); // ego v strukture HdwCfg

            if (HdwCfg.HddPresent)

                printf("\nHDD present");//diskovod

            if (HdwCfg.NpuPresent)

                printf("\nNPU present");//matematicheskii soprotssessor

```

```

printf("\nRAM banks: %d", HdwCfg.AmountOfRAM); //razmer OZU
printf("\nVideo Mode: %d", HdwCfg.VideoMode); //aktivnyy videorezhim
printf("\nNumver of FDD: %d", HdwCfg.NumberOfFdd + 1); //chislo
obnaruzhennykh NGMD

if (HdwCfg.DmaPresent)
    printf("\nDMA present");//nalichie kontrollera DMA

printf("\nNumber of COM ports: %d", HdwCfg.NumberOfCom); //chislo
COM-portov

if (HdwCfg.GamePresent)
    printf("\nGame Adapter present");//igrovoy adapter

if (HdwCfg.JrComPresent)
    printf("\nPCjr Com present");//rezerv

printf("\nNumber of LPT ports: %d", HdwCfg.NumberOfLpt); //chislo
LPT-portov

printf("\nPress Any Key...");
getch();
clrscr();
indidication();
}

if (key == 50)//Esli 2 , to
{
    printf("\n\nCalling the INT 12h \n");
    rg.h.ah = 0x0;// Vyzyvaem preryvanie INT 12h dlya opredeleniya
    int86(0x12, &rg, &rg); // ob''ema osnovnoy operativnoy pamyati
    printf("\nRAM installed: %d Kbytes", (unsigned int)rg.x.ax);
    printf("\nPress Any Key...");
    getch();
    clrscr();
    indidication();
}

if (key == 51)//Esli 3 , to
{
    printf("\n\nCalling the INT 16h \n");

```



```

waiting\n");

printf("\n 1 - Input a symbol\n 2 - Input a string\n 3 - Input with

while (pick < 49 || pick>51)
{
    pick = getch();
}

switch (pick)
{
case 49:
    printf("\nInput the symbol: ");
    rg.h.ah = 0; // Vyzyvaem preryvanie INT 16h dlya
    int86(0x16, &rg, &rg); // vvoda simvola
    printf("\nSymbol is %c", rg.h.al); //i vyvoda ego na ekran
    break;
case 50:
    printf("\nPRESS ESC TO EXIT\nThe string is: ");
    while (rg.h.ah != 1)
    {
        rg.h.ah = 0; // Vyzyvaem preryvanie INT 16h dlya
        int86(0x16, &rg, &rg); // vvoda stroki
        printf("%c", rg.h.al); //i vyvoda ego na ekran
    }
    break;
case 51:
    printf("PRESS ESC TO EXIT\nInput the string: ");
    while (rg.h.ah != 1)
    {
        rg.h.ah = 0; // Vyzyvaem preryvanie INT 16h dlya
        int86(0x16, &rg, &rg); // vvoda stroki
        delay(1000);
        printf("%c", rg.h.al); //i vyvoda ego na ekran
    }
    break;
}

```

```

        case 27:

            break;

    }

    pick = 0;

    rg.h.ah = 0;

    printf("\nPress Any Key...");

    getch();

    clrscr();

    indidication();

}

if (key == 52)//Esli 4, to
{

    printf("\n\nCalling the INT 1Ah \n");

    printf("\n1.See the current time and date\n2.Set the time\n");

    while (pick < 49 || pick>50)

    {

        pick = getch();

    }

    switch (pick)

    {

        case 49:

            rg.h.ah = 0x02;//Schityvanie vremeni s chasov real'nogo
vremeni

            int86(0x1a, &rg, &rg);

            printf("\nTime is: %02.2d:%02.2d:%02.2d\n",

                bcd1bin(&(rg.h.ch)), //znachenie chasov

                bcd1bin(&(rg.h.cl)), //znachenie minut

                bcd1bin(&(rg.h.dh)) //znachenie sekund

            );

            rg.h.ah = 0x04;//Schityvanie daty s chasov real'nogo vremeni

            int86(0x1a, &rg, &rg);

            printf("\nDate is: %d day,%s,%d year.\n",

```

mesyatsa

```
        bcd1bin(&(rg.h.d1)), //znachenie dnya
        month_to_text[bcd1bin(&(rg.h.dh)) - 1], //znachenie
        bcd2bin(&(rg.h.cl)) //znachenie goda
    );
    break;
case 50:
    do
    {
        printf("\n Enter hour (0-23): ");
        scanf("%li", &hour);
    } while (hour > 23 || hour < 0);
    do
    {
        printf("\n Enter minute (0-60): ");
        scanf("%li", &minute);
    } while (minute > 59 || minute < 0);
    do
    {
        printf("\n Enter second (0-60): ");
        scanf("%li", &second);
    } while (second > 59 || second < 0);
    rg.h.ah = 0x03; //Ustanovka vremeni
    bin1bcd(hour, &(rg.h.ch)); //znachenie chasov
    bin1bcd(minute, &(rg.h.cl)); //znachenie minut
    bin1bcd(second, &(rg.h.dh)); //znachenie sekund
    int86(0x1a, &rg, &rg);
    break;
}
pick = 0;
printf("\nPress Any Key...");
getch();
clrscr();
```

```
        indidication();  
    }  
}  
}
```

5. Скриншоты выполнения программы:

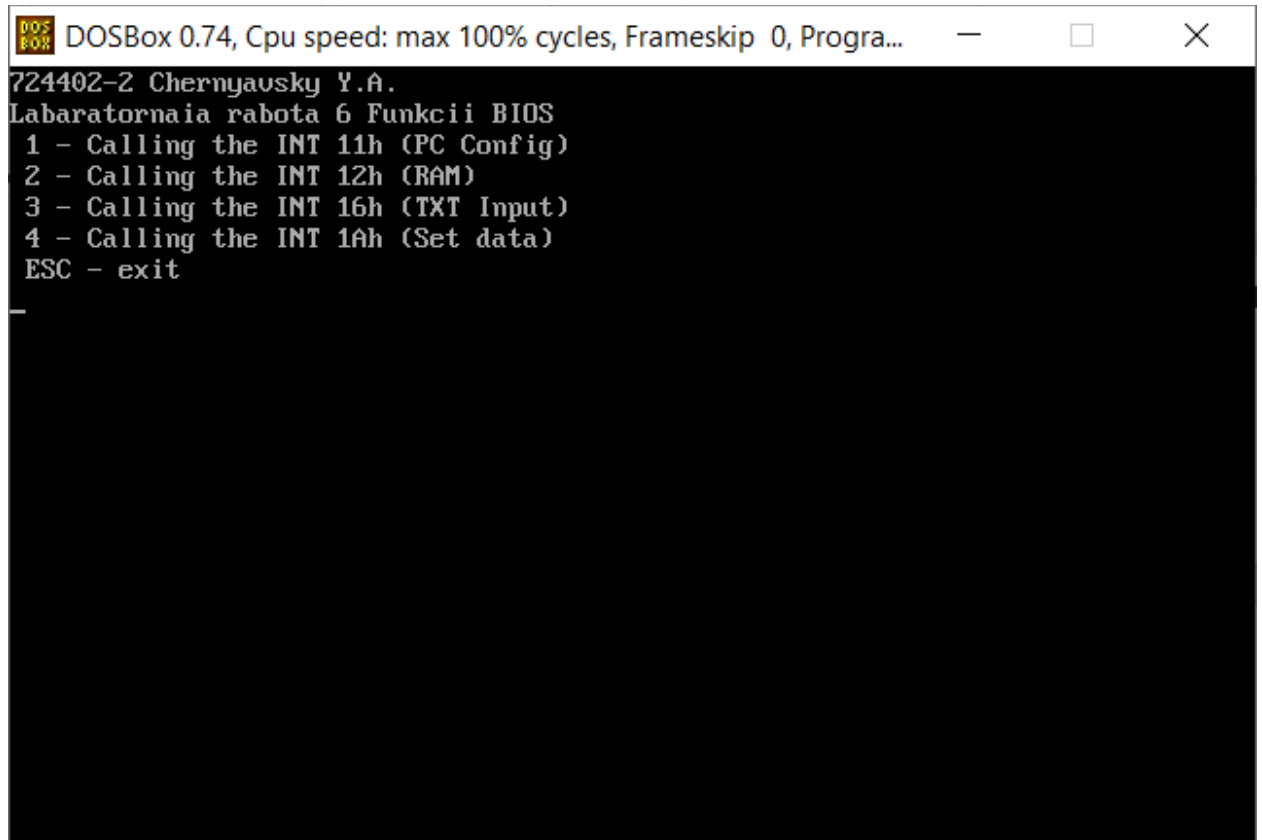
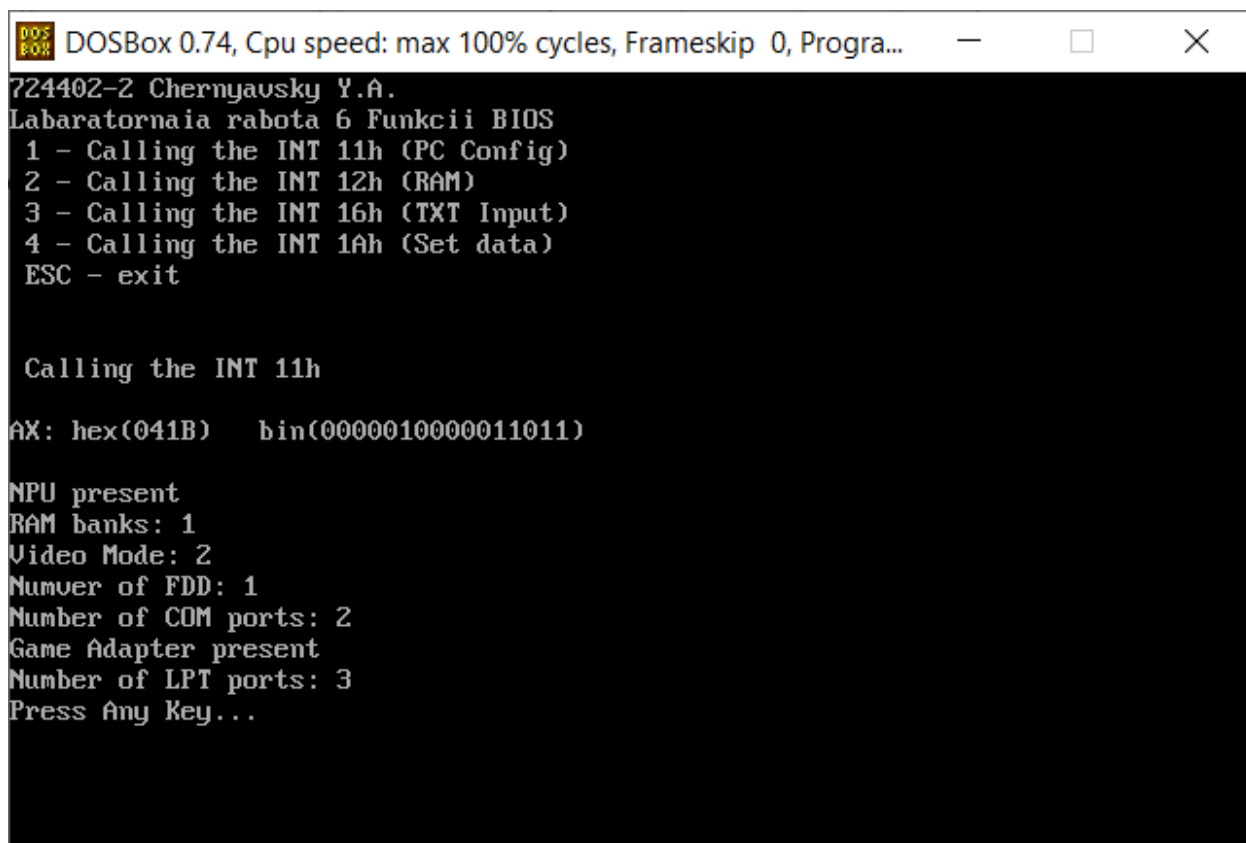


Рис.5.1 – Начало программы



DOSBox 0.74, Cpu speed: max 100% cycles, Frameskip 0, Progra...

724402-2 Chernyavsky Y.A.
Labaratornaia rabota 6 Funkcii BIOS

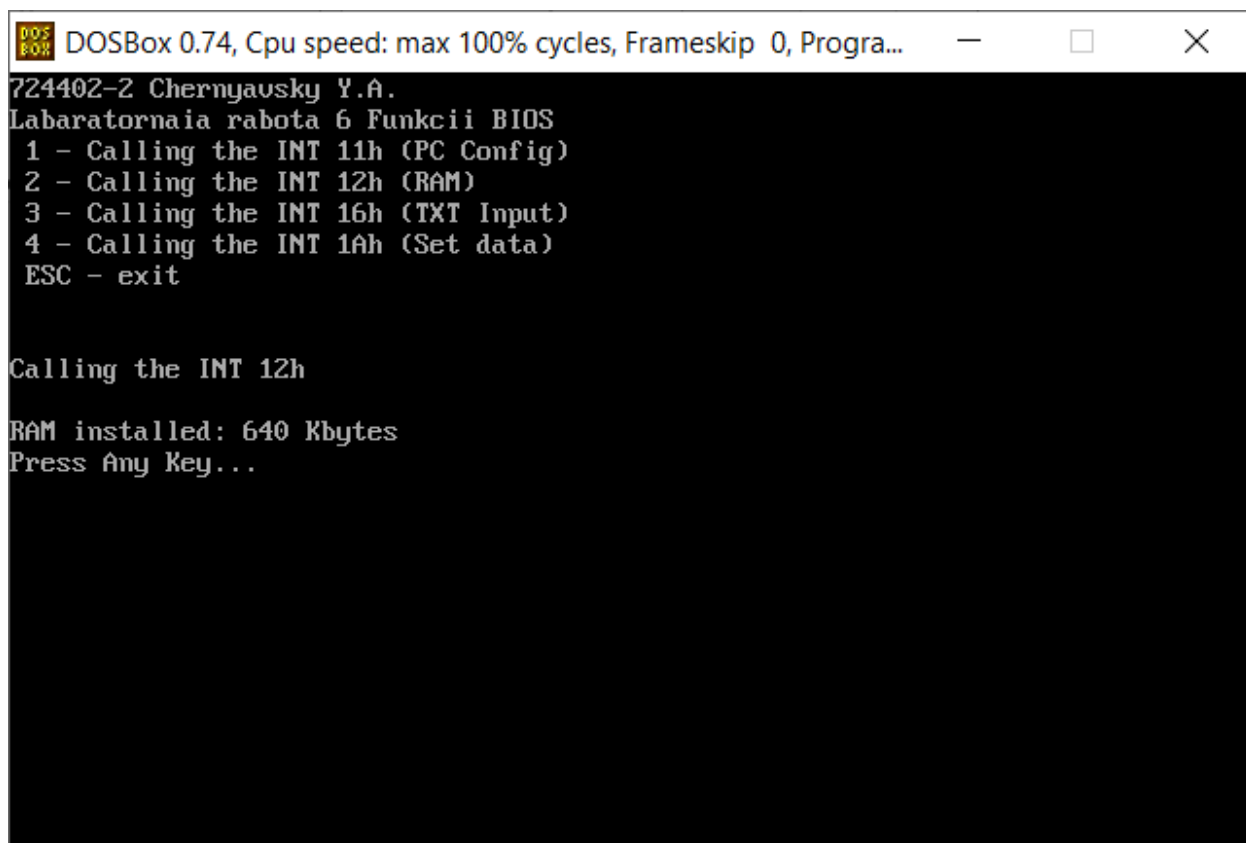
- 1 - Calling the INT 11h (PC Config)
- 2 - Calling the INT 12h (RAM)
- 3 - Calling the INT 16h (TXT Input)
- 4 - Calling the INT 1Ah (Set data)
- ESC - exit

Calling the INT 11h

AX: hex(041B) bin(000000100000011011)

NPU present
RAM banks: 1
Video Mode: 2
Numver of FDD: 1
Number of COM ports: 2
Game Adapter present
Number of LPT ports: 3
Press Any Key...

Рис.5.2– Показ списка оборудования



DOSBox 0.74, Cpu speed: max 100% cycles, Frameskip 0, Progra...

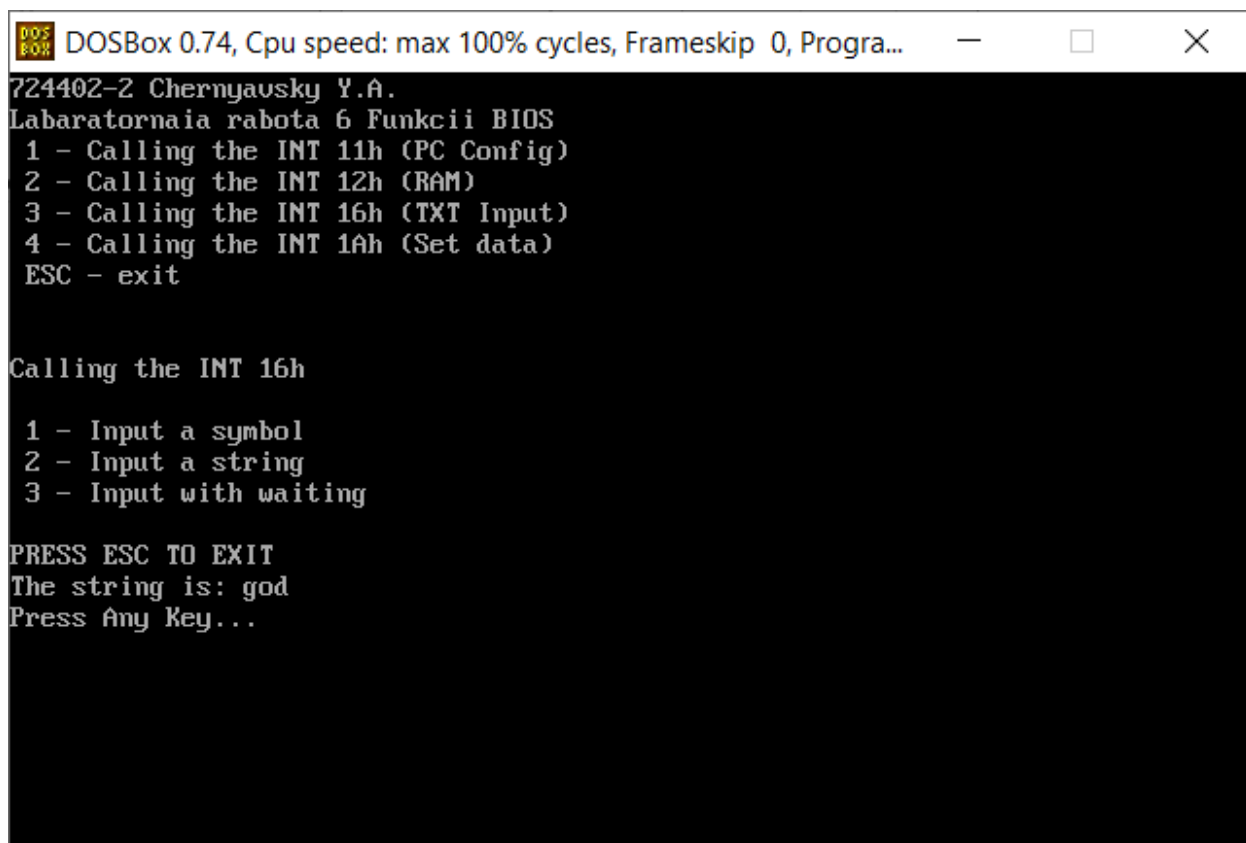
724402-2 Chernyavsky Y.A.
Labaratornaia rabota 6 Funkcii BIOS

- 1 - Calling the INT 11h (PC Config)
- 2 - Calling the INT 12h (RAM)
- 3 - Calling the INT 16h (TXT Input)
- 4 - Calling the INT 1Ah (Set data)
- ESC - exit

Calling the INT 12h

RAM installed: 640 Kbytes
Press Any Key...

Рис.5.3 –Размер непрерывной памяти



DOSBox 0.74, Cpu speed: max 100% cycles, Frameskip 0, Progra...

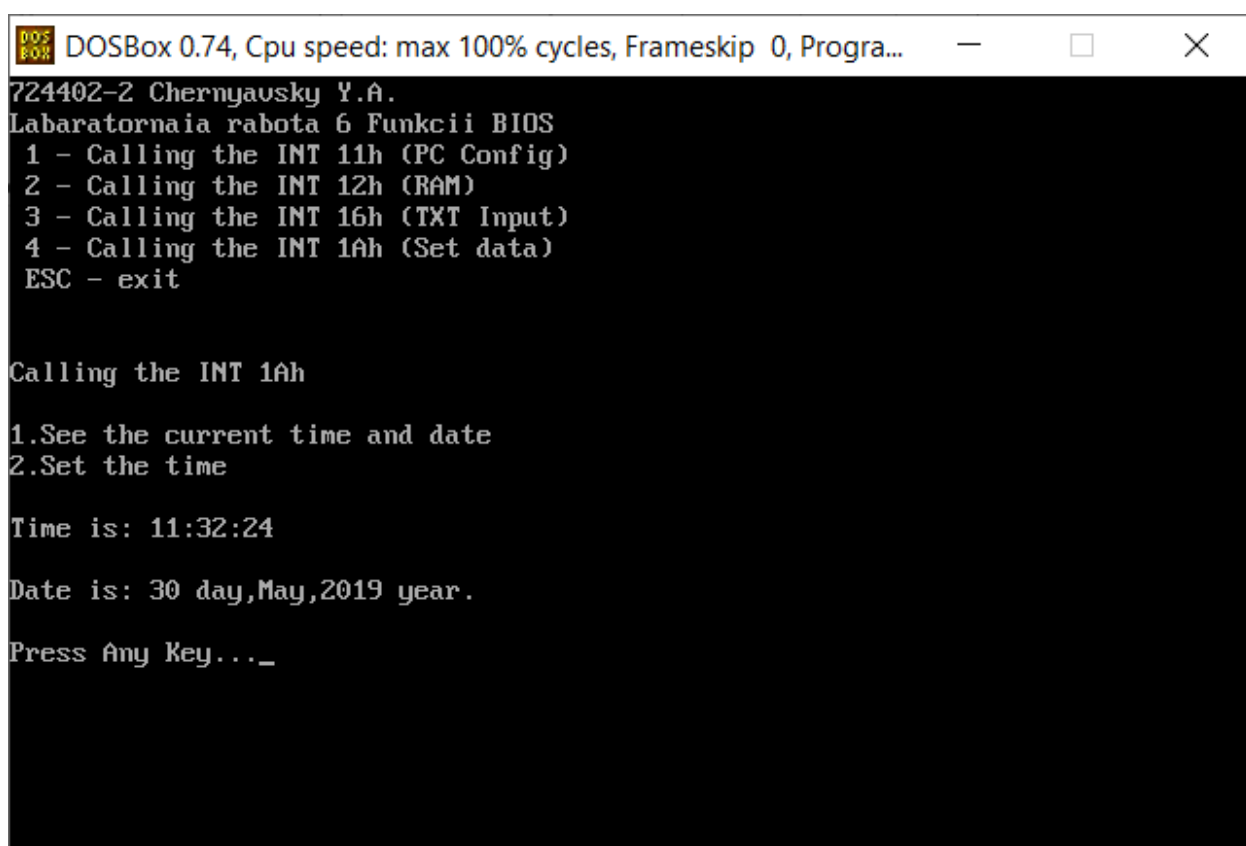
724402-2 Chernyavsky Y.A.
Labaratornaia rabota 6 Funkcii BIOS
1 - Calling the INT 11h (PC Config)
2 - Calling the INT 12h (RAM)
3 - Calling the INT 16h (TXT Input)
4 - Calling the INT 1Ah (Set data)
ESC - exit

Calling the INT 16h

1 - Input a symbol
2 - Input a string
3 - Input with waiting

PRESS ESC TO EXIT
The string is: god
Press Any Key...

Рис.5.4 –Ввод строки



DOSBox 0.74, Cpu speed: max 100% cycles, Frameskip 0, Progra...

724402-2 Chernyavsky Y.A.
Labaratornaia rabota 6 Funkcii BIOS
1 - Calling the INT 11h (PC Config)
2 - Calling the INT 12h (RAM)
3 - Calling the INT 16h (TXT Input)
4 - Calling the INT 1Ah (Set data)
ESC - exit

Calling the INT 1Ah

1. See the current time and date
2. Set the time

Time is: 11:32:24
Date is: 30 day, May, 2019 year.
Press Any Key..._

Рис.5.5 –Вывод даты и времени

6. Вывод

В ходе выполнения лабораторной работы были изучены функции BIOS, использованы векторы прерываний различных назначений, получены данные о компьютере и его составляющих.