Министерство образования и науки РФ

Государственное образовательное учреждение
высшего профессионального образования

Владимирский государственный университет

Кафедра вычислительной техники

ВЫЧИСЛИТЕЛЬНЫЕ СРЕДСТВА ИНФОРМАЦИОННЫХ СИСТЕМ

Методические указания к лабораторным работам

Составитель
В. И. БЫКОВ

Владимир 2010

УДК 004.382(075.8)
ББК 32.973.202я 73
В94

Рецензент
Доктор технических наук, профессор
кафедры информационных систем и информационного менеджмента
Владимирского государственного университета
Р. И. Макаров

Печатается по решению редакционного совета
Владимирского государственного университета

Вычислительные средства информационных систем : метод. В94 указания к лаб. работам / Владим. гос. ун-т ; сост. В. И. Быков. – Владимир : Изд-во Владим. гос. ун-та, 2010. – 43 с.

Содержат четыре лабораторные работы по дисциплине «Вычислительные средства информационных систем», требования к оформлению отчета и вопросы для самоконтроля. Тематика работ охватывает все разделы дисциплины и, прежде всего, раздел системы ввода – вывода.

Предназначены для студентов третьего курса всех форм обучения специальностей 230201 – информационные системы и технологии (системы компьютерной графики и мультимедиа технологии) и 230202 – информационные технологии в образовании.

Ил. 14. Табл. 1. Библиогр.: 10 назв.

УДК 004.382(075.8)
ББК 32.973.202я 73

Введение

Дисциплина «Вычислительные средства информационных систем (ИС)» – важная составляющая в подготовке специалистов по информационным системам и технологиям. Она знакомит их с аппаратными средствами ИС, в том числе с ядром ЭВМ, периферийными (внешними) устройствами и их интерфейсами. Методические указания содержат четыре лабораторных работы, в которых рассматриваются оценка производительности компьютеров, работа с прерываниями, видеосистема ПК и его диагностика.

При выполнении работ студенты изучают устройство исследуемых подсистем и разрабатывают программы работы с ними. Очередность работ соответствует лекционному курсу и может быть при необходимости изменена. При недостатке времени наиболее безболезненно может быть исключена работа № 4. В методических указаниях из-за ограничения объема дается лишь неполное описание рассматриваемых подсистем; подробное описание дается в электронном варианте, размещаемом на компьютере в лаборатории, в которой выполняются работы.

Лабораторная работа № 1

ОЦЕНКА ПРОИЗВОДИТЕЛЬНОСТИ ЭВМ, ЕЕ УСТРОЙСТВ И УЗЛОВ

Цель работы: изучение методов, методик и способов оценки производительности компьютера, его устройств и узлов.

Оценка производительности ЭВМ

При оценке производительности ЭВМ возникают следующие проблемы:

• достоверность оценки;
• адекватность оценки;
• интерпретация оценки.

Требования к тестам:

1. Тест должен получить признание широких кругов компьютерной общественности (производители, пользователи и эксперты).

2. Методика измерения производительности не должна допускать оптимизации, т.е. поддерживать объективные результаты.

3. Соответствие результатов тестирования представлению пользователей о типовых задачах.

4. Тест должен обеспечивать простоту запуска на компьютере любого типа.

5. Моделенезависимость.

Способы толкования оценки:

1. Средняя производительность компьютера.

2. Средняя производительность тестового пакета.

3. Производительность при выполнении определенной задачи, т.е. натуральная.

4. Пиковая производительность.

5. Производительность по сравнению с принятой за эталон производительностью некоторого компьютера.

Оценка производительности процессора:

1. Количество и длительность тактов синхронизации.

2. Частота синхронизации.

3. Среднее количество тактов синхронизации на одну команду процессора.

Способы и методы оценки производительности:

1. *MIPS* – количество миллионов команд (операций) в секунду.

Недостатки:

а) зависит от набора команд процессора;

б) зависит от набора команд в программе;

в) не всегда предсказуемо.

2. *MFLOPS* – миллионы операций с плавающей точкой в секунду.

3. *LINPACK* – ливерморские циклы.

4. *SPEC – standart performance evaluation corp*.

5. *TPC – A/B/C* (обработка и трансляция транзакций): A – количество транзакций в секунду; *B* – тест базы данных (БД); *C* – обработка заказов.

6. *Bench mark* – тестирование на многих пакетах. Пакеты включают

задачи по работе с БД, дисками, математические задач и обработку графики и звука и т.п.

Методические указания

1. Изучить методы, методики и способы оценки производительности компьютера и его устройств и узлов, используя литературу и материалы лекций.

2. Разработать методику оценки производительности компьютера или его устройства или узла по заданию преподавателя.

3. Разработать алгоритм и программу для оценки производительности компьютера или его устройства или узла.

4. Выполнить программу с несколькими наборами исходных данных, определяя каждый раз время выполнения. Количество повторений заданных операций должно быть большим и изменяться от 100 или 1000 до 1млрд с кратностью 10, т.е. в логарифмическом масштабе, количество экспериментов, т.е. точек на графике – не менее 5. Для измерения времени использовать процедуры или функции, имеющиеся в языках программирования. Например, в Паскале процедура

GetTime(h,m,s,d)

дает значение текущего времени, где *h* – часы; *m* – минуты; *s* – секунды; *d* – сотые доли секунды (все параметры – типа word).

Время выразить в сотых или тысячных долях секунды.

5. Повторить эксперименты на другом компьютере, имеющем другую конфигурацию и параметры производительности.

6. Вычислить (или получить на компьютере) некоторые оценки производительности компьютера или его устройства или узла.

7. Составить таблицу и построить графики полученных зависимостей. При построении графиков на формате А4 по горизонтальной оси абсцисс должно быть указано количество повторений в логарифмическом масштабе (не менее 5 точек), по вертикальной оси ординат на всю высоту листа – время в сотых или тысячных долях секунды. При малом разрешении по вертикали можно построить два графика – для малых и больших значений времени.

Примечания:

1. При выполнении теста компьютер не должен быть загружен другими задачами.

2. При программировании необходимо использовать язык более низкого уровня (Ассемблер, Паскаль, Си, а не Delphi, Java).

3. При выполнении операций результаты должны иметь тот же тип, что и операнды, чтобы не было затрат на преобразование типов.

4. При выполнении операций результаты не должны вызывать переполнение.

Содержание отчета

1. Цель работы.
2. Методика оценки.
3. Алгоритм.
4. Программа.
5. Условия экспериментов (детальное описание параметров и конфигурации вычислительной системы (ВС).
6. Таблицы и графики полученных зависимостей.
7. Выводы.

Варианты индивидуальных заданий

1. Производительность дисковых операций (чтение и запись).
2. Сравнительная производительность при выполнении операций с действительными (2 вида) и целыми числами.
3. Сравнительная оценка производительности двух различных ВС (машин) при выполнении различных операций, а именно:

а) короткие операции с действительными (3 вида) числами;

б) длинные операции с действительными (3 вида) числами;

в) короткие операции с целыми (3 вида) числами;

г) длинные операции с целыми (3 вида) числами.

4. Сравнительная оценка накладных расходов на организацию циклов различного типа (*for, while, repeat*) в различных языках программирования: а) Ассемблер; б) Паскаль; в) Си.
5. Сравнительная производительность при выполнении операций преобразования типов:

а) различные целые типы – в различные действительные;

б) различные действительные типы – в другие действительные.

6. Матричные операции с числами различных типов.
7. Векторные операции с числами различных типов.

Вопросы для самоконтроля

1. Способы оценки производительности ЭВМ (перечислить).

2. Способы оценки производительности ЭВМ по *MIPS*.

3. Способы оценки производительности ЭВМ по *MFLOPS*.

4. Способы оценки производительности ЭВМ по *World Banch*.

5. Способы оценки производительности ЭВМ по ливерморским циклам.

6. Проблемы при оценке производительности.

7. Составляющие производительности компьютера.

8. Виды времени, которые могут оцениваться.

9. Категории тестов.

10. Требования к тестам.

11. Способы толкования оценки производительности.

12. Способы оценки производительности процессора.

Лабораторная работа № 2

СИСТЕМА ПРЕРЫВАНИЙ ЭВМ

Цель работы: изучение системы прерываний ЭВМ и программ обработки прерываний в *DOS*.

Теоретические сведения

Прерывание – это кратковременная приостановка текущей процедуры программы, позволяющая выполнить другую процедуру. После завершения прерывания прерванная программа продолжает выполняться так, как будто бы ничего не происходило. Эти две процедуры могут быть несвязанными, и прерывание не окажет никакого воздействия на прерванную процедуру. Они могут быть взаимозависимы – прерванная программа может быть модифицирована процедурой обработки прерывания. Прерывание может быть вызвано внешним по отношению к выполняемой программе событием, или действием самой программы, или аппаратно, или командой из программы.

В компьютере *PC* существуют 256 различных прерываний с номерами от 0 до *FF (hex)*. При выполнении прерывания содержимое регистров признаков (флаги) сохраняются в стеке. После этого прерывания запрещаются, и выполняется программа с адреса, соответствующего предыдущему прерыванию. Запрет прерываний осуществляется функцией *“disable()”,* а разрешение – функцией *“enable()”*.

Эта программа должна сохранить используемые ею регистры, вы-

полнить свою задачу, восстановить значения регистров, выполнить команду возврата из прерывания, которая восстанавливает адрес прерванной программы и регистра признаков так, что прерванная программа продолжит исполнение с того места, где была прервана.

Аппаратные прерывания вызываются событиями, физически связанными в аппаратуре с соответствующими векторами прерываний. Например, клавиатура связана с прерыванием 09*h*, обращение к дисковым устройствам связано с прерыванием 13*h* и т. д.

Программные прерывания происходят при выполнении в текущей программе команды типа *int86()* с номером прерывания в качестве операнда. В остальном никакой разницы между программным и аппаратным прерываниями нет.

Программы обработки прерывания реагируют на прерывания от аппаратуры или от программ и обычно предназначены для поддержки различных устройств. Примером такой программы может быть обработчик прерывания от таймера или программа обработки аппаратных прерываний от устройства типа «мышь».

Захват прерывания. По прерыванию может выполняться ваша программа, если на неё указывает соответствующий вектор. Например, в языке Си это осуществляется с помощью функции *setvect (num_interrupt, address*), где *num_interrupt* – номер вектора прерывания (от 0 до *FFh*), а значение *address* – адрес программы обработки прерывания.

Замечание: полезно восстанавливать прерывания после их изменения, для этого нужно сохранить адрес обработчика до его замещения, это делается с помощью функции *void interrupt *getvect (num interrupt),* где *num interrupt* – номер вектора прерывания.

Совместное использование прерываний. Если необходимо использовать прерывания, которые уже используются другими программами, то необходимо сделать так, чтобы те программы не замечали вашего воздействия на них. Например, если осуществляется перехват таймера, то необходимо обеспечивать выполнение старой процедуры, ибо тогда часы, встроенные в ваш ПК, будут стоять. Или если вы перенаправите обработчик клавиатуры, то есть вероятность, что клавиатура не будет реагировать на все ваши дальнейшие усилия (если вы не выполните процедуру, которая выполнялась ранее по этому вектору).

Прерываниями (*interruption*) являются штатные ситуации, возникающие при поступлении соответствующих команд (программные преры-

вания) или внешних сигналов (аппаратные прерывания), а исключениями (*exeption*) – нештатные ситуации (ошибки), возникающие при работе процессора. При выявлении таких ошибок соответствующие блоки, контролирующие работу процессора, вырабатывают внутренние сигналы запроса, обеспечивающие вызов необходимой подпрограммы обслуживания.

Процессор способен обеспечить обслуживание 256 различных типов исключений и прерываний. Соответствующая обработка информации при возникновении таких ситуаций выполняется с помощью специальных подпрограмм обслуживания, начальные адреса (векторы) которых хранятся в таблице, размещаемой в памяти системы.

Запросы на выполнение аппаратных прерываний поступают от внешних устройств на входы *LINT0/INTR, LINT1/NMI* процессора. В мультипроцессорной системе, когда включен внутренний контроллер локальных прерываний *APIC*, сигналы *LINT1-0* на этих входах определяют номер запроса, поступающего от других устройств (процессоров) системы. В однопроцессорной системе, когда функционирование контроллера *APIC* запрещено, эти входы служат, соответственно, для подачи маскируемых *INTR* и немаскируемых *NMI* запросов прерывания от различных внешних устройств.

На вход *INTR* поступают маскируемые запросы прерываний, обслуживание которых может быть запрещено (замаскировано) путем установки значения признака *IF* =1 в регистре состояний *EFLAGS*. Обычно такой запрос поступает от внешнего устройства через специальный контроллер прерываний, который собирает запросы от различных внешних устройств и передает их для обработки процессору, указав для них также номер *n*, определяющий вид прерывания (*INTR n*). На вход *NMI* поступает запрос на немаскируемое прерывание, процедура обслуживания которого имеет фиксированный номер $n = 2$. Значение признака *IF* в регистре *EFLAGS* не влияет на обслуживание процессором немаскируемого запроса прерывания *NMI*. При работе процессора в мультипроцессорной системе, когда функционирует контроллер локальных прерываний *APIC*, запросы аппаратных прерываний (немаскируемых и маскируемых) поступают по специальной *APIC*-шине.

Программные прерывания реализуются при поступлении команд *INT n, INT3, INTO, BOUND*. Эти команды вызывают переход к выполнению подпрограмм обслуживания, векторы которых можно найти в литературе [2, с. 481 – 483].

Для исключений зарезервированы первые 32 вектора в таблице прерываний, из которах в процессорах семейства *P*6 используются только 19. Каждый тип исключения имеет мнемоническое обозначение. Исключения делятся на ошибки *(faults)*, ловушки *(traps*) и отказы *(aborts)*.

Обработка прерываний

Прерывание – событие, при котором меняется нормальная последовательность команд, выполняемых процессом. Для обработки прерываний существуют специальные средства: аппаратные и программные. Частично это выполняет ядро операционной системы (ОС), но лишь в минимальной степени. Действия ЭВМ при обработке прерываний приведены на рис. 1, а временная диаграмма выполнения этих действий – на рис. 2.

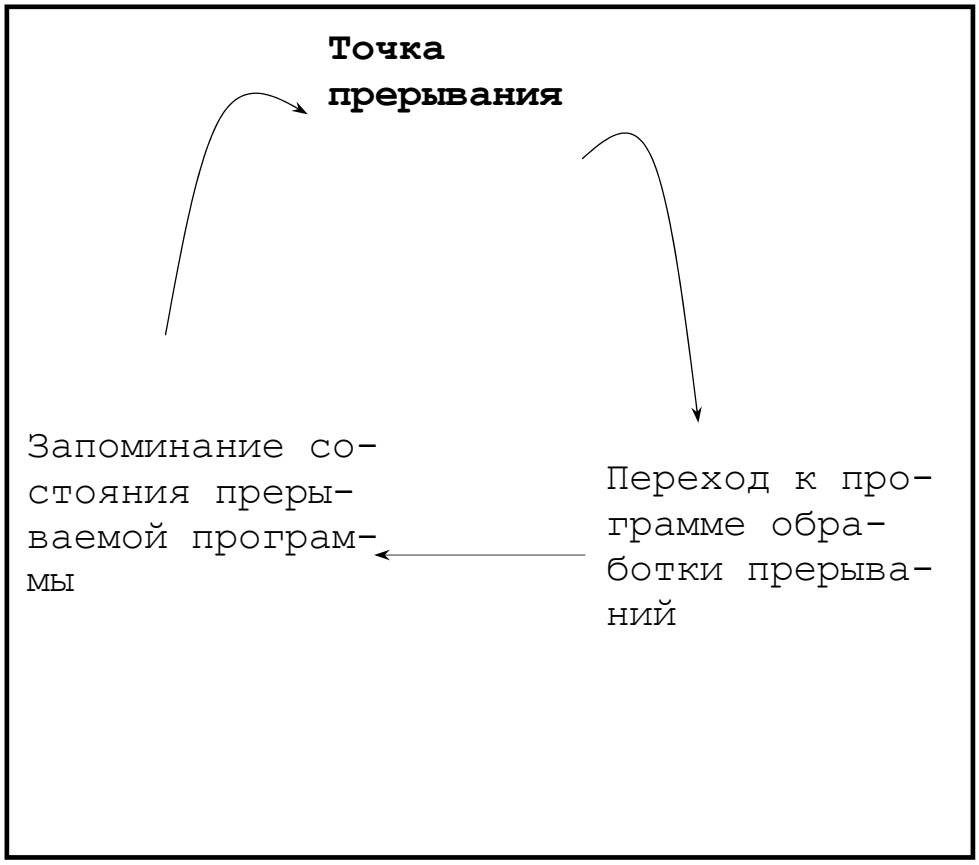


Рис. 1. Схема работы ЭВМ при прерывании

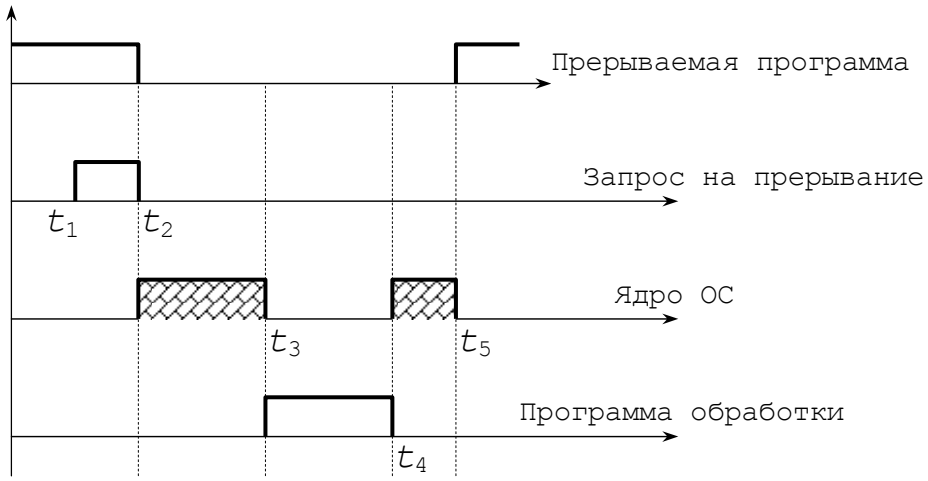


Рис. 2. Временная диаграмма обработки прерывания:

$t_2 - t_1 = t_р$ – время реакции; $t_3 - t_2 = t_з$ – время запуска программы обработки прерываний; $t_4 - t_3 = t_{оп}$ – время работы программы обработки прерываний; $t_5 - t_4 = t_в$ – время возобновления прерванной программы; $t_в + t_з = t_р$ – время накладных расходов

Прерывания IBM PC

Существуют прерывания *BIOS* (0 – 1*F*)и прерывания *DOS* (20 – *FF*).

1. Аппаратные прерывания от внешних устройств:
 - отказ питания;
 - таймер;
 - клавиатура;
 - адаптер связи;
 - накопитель на гибких магнитных дисках (НГМД); и т.д.
2. Исключения (логические прерывания от микропроцессора): 1 – пошаговый режим; 3 – достижение контрольной точки в процессе трассировки программы; 4 – переполнение.
3. Программные прерывания.

Схема движения информации при обработке прерываний

В *IBM PC* память делится на сегменты. Максимальный объем сегмента равен в ОС *MS-DOS* – 64 Кбайт. Адрес команды хранится в двух регистрах процессора: *CS* – начальный адрес сегмента кода; *IP* – смещение относительно начала сегмента. Реальный физический адрес команды в оперативной памяти вычисляется как сумма *CS* + *IP*. При поступлении сигнала прерывания ПК выполняет следующие действия.

а) завершение текущей команды;

б) сохранение адреса следующей команды в стеке;

в) обращение к таблице векторов прерываний;

г) получение в таблице адреса программы обработки прерываний;

д) выполнение программы обработки прерываний;

е) обращение к стеку за адресом прерванной программы (при достижении команды *IRET*);

ж) запись этого адреса из стека в соответствующие регистры *CS, IP*;

з) возвращение к выполнению прерванной программы.

Такая организация позволяет иметь разные программы для обработки одного и того же типа прерывания.

Структура данных при обработке прерываний в программе, написанной на PASCAL

Для обращения к содержимому регистров необходимо использовать структуру данных (запись), описанную в модуле *DOS Turbo-Pascal*:

```
type
  Registers = record
    case integer of
      0: (AX, BX, CX, DX, BP, SI, DS, DI, ES, Flags: Word);
      1: (AL, AH, BL, BH, CH, DL, DH, CL:Byte);
    end;
```

Модули DOS:

Intr (IntNo: Byte; var Regs: Registers) – выполняет прерывания с заданным номером IntNo;

MsDOS (var Regs: Registers);

SetIntVec (IntNo: Byte; Vector: Pointer) – адрес процедуры обработки прерывания;

GetInVec (IntNo: Byte; var Vector: Pointer) – получение и запись адреса из таблицы векторов.

Процедура для обработки прерывания должна иметь вид:

Procedure <имя> *(Flags, CS, IP, AX, BX, CX, DX, SI, DI, DS, ES, BP: Word)*; *interrupt;*

Программа обработки прерывания *Ctrl-Break*

```
Program EX
  uses crt, dos;
  var SaveAddrInt: Pointer;
            i: Integer;
{=========================================}
```

```
  {$F+} – указание на наличие межсегментных связей
   procedure Int1B: interrupt;
       Begin
       end;
  {$F-}
{=============================================}
Begin
  GetIntVec ($1B, SaveAddrInt); (сохранение адреса старого обработчика)
    For I:= 1 to 10 do
      delay (1000);
  SetIntVec ($1B, SaveAddrInt);
End.
```

Пример программы перехвата прерывания от таймера

```
program init1C;
uses dos,Crt;
var
 Save_1C: pointer;
 c: char;
{=========================================}
procedure pp1;interrupt;
const k:word = 0;
begin
 k:=k+1;
 k:=k mod 4;
 gotoxy(54,2);
 case k of
  0: Write('/');
  1: Write('-');
  2: Write('\');
  3: Write('|');
 end;
end;
{=========================================}
begin
 GetIntVec($1C,Save_1C);
 clrscr;
```

```
 Writeln('Press any key to continiue...');
 Writeln('Process is on....');
 SetIntVec($1C,@pp1);
 c:=readkey;
 SetIntVec($1C,Save_1C);
end.
```

Пример перехвата прерывания от клавиатуры

```
program sk;
uses crt,dos;
var
 savekbint:procedure;
 f:string;
{=========================================}
procedure beep;
 begin
  sound(1000);
  delay(100);
  nosound;
 end;
{=========================================}
procedure new_in;interrupt;
 const sign:boolean=true;
begin
 asm
  sti
 end;
if sign then beep;
sign:=not sign;
inline($9C);
savekbint;
asm
 cli;
end;
end;

{=========================================}
```

```
begin
 Writeln('Ready!');beep;write(':');
 GetIntVec($9,@savekbint);
 SetIntVec($9,addr(new_in));
 readln(f);
 SetIntVec($9,@savekbint);
 end.
```

Общее задание

1. Используя пример программного прерывания, вывести заданное количество заданных символов в заданной позиции экрана в режиме 80 символов на строку восстановить режим и снова вывести строку, (установка режима – функция 00H прерывания 10H, режим 80 колонок – 3, 40 колонок – 1, описание функции: в AL – номер режима).

2. Используя прерывание таймера, написать собственный обработчик прерывания, который выводит на экран чередование букв Вашей фамилии (можно латинскими буквами).

Варианты индивидуальных заданий

Написать программу, выполняющую указанные действия с заданным устройством (системой), используя указанные прерывания, возможные для обработки (номер главы дан по [6]).

1. Клавиатура – 09h, 16h (гл. 3):
   а) управление клавиатурой;
   б) доступ к отдельным клавишам;
   в) сводка скэн-кодов клавиш.
2. Диски – 13h (гл. 5):
   а) управление распределением диска;
   б) работа с каталогами;
   в) подготовка к работе с файлами;
   г) чтение и запись файла;
   д) подсчитать количество обращений к дисковым накопителям, используя прерывание.
3. Таймеры 1*Ah*,15*h* (гл. 2):
   а) установка и чтение таймера;
   б) создание звука;
   в) провести эксперимент с прерыванием таймера:

1) переопределить обработчик без выполнения внутри себя старого обработчика;

2) то же, но с выполнением старого обработчика.

Зафиксировать отставание встроенных часов в первом случае и его отсутствие во втором случае.

4. Принтер – 17 *h* (гл. 6):

а) управление работой принтера;

б) установка спецификаций печати;

в) посылка данных на принтер.

5. COM-порт – 14 *h* (гл. 7):

а) доступ к последовательному порту;

б) создание драйвера устройства;

в) использование устройств ввода/вывода.

Пример выполнения общего задания 1

```
program z1;
 uses dos,crt;
 var
  ch:char;
  regs:registers;
{=========================================}
procedure setr80;
begin
 with regs do
  begin
   al:=3;
   ah:=$00;
   intr($10,regs);
  end;
end;
{=========================================}
procedure setr40;
begin
 with regs do
  begin
   al:=1;
   ah:=$00;
   intr($10,regs);
```

```
  end;
end;
{=========================================}
procedure out5z;
begin
 with regs do
  begin
   ah:=$A;
   al:=$5A; {прописная Z}
   bh:=0;   {страница}
   cx:=5;
   intr($10,regs);
  end;
end;
{=========================================}
 begin
  setr80;
  gotoxy(10,10);
  out5z;
  ch:=readkey;
  setr40;
  gotoxy(10,10);
  out5z;
  ch:=readkey;
  setr80;
  gotoxy(10,10);
  out5z;
  ch:=readkey;
  clrscr;
 end.
```

Вопросы для самоконтроля

1. Описать механизм обработки прерывания.
2. Временная диаграмма при обработке прерывания.
3. Виды прерываний и их особенности.
4. Почему используются два контроллера прерываний?

5. Как соединены контроллеры прерываний?

6. Какова методика изменения обработчика прерываний и программные средства?

7. Назначение прерываний.

8. Назначение контроллера прерываний.

Лабораторная работа № 3

ВИДЕОСИСТЕМА ПК

Цель работы: изучение видеосистемы ПК (системы отображения информации), ее элементов, устройств и узлов. Приобретение навыков программирования видеосистемы ПК с использованием прерываний.

Теоретические сведения

Видеосистема ПК включает следующие основные блоки (рис. 3):

- северный мост, обеспечивающий связь видеоадаптера (видеокарты) с процессором и операционной памятью (ОП);
- интерфейс северный мост – видеокарта типа *PCI – Express*×16 или *AGP*×8 (в старых системах);
- видеоадаптер (видеокарта);
- интерфейс видеокарта – монитор (аналоговый или цифровой);
- монитор на ЭЛТ-трубке или жидкокристаллический.

Более детально видеосистема и ее интерфейсы рассмотрены в [5].

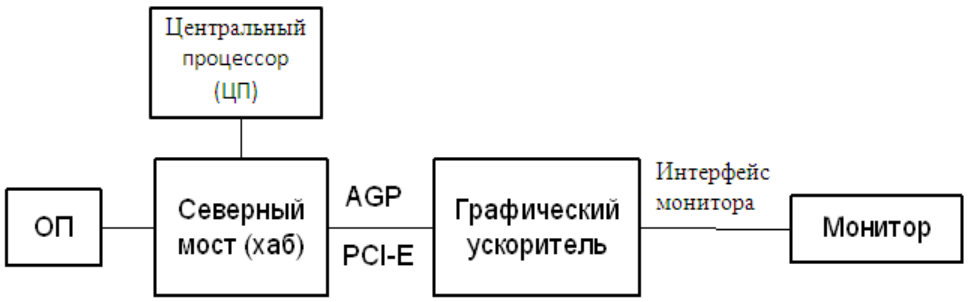


Рис. 3. Структурная схема видеосистемы ПК

Мониторы

ЭЛТ-мониторы

Сегодня самый распространенный тип мониторов - это ЭЛТ или

CRT (*Cathode Ray Tube*) мониторы. Как видно из названия, в основе всех подобных мониторов лежит катодно-лучевая трубка, но это дословный перевод, технически правильно говорить электронно-лучевая трубка (ЭЛТ). Иногда *CRT* расшифровывается и как *Cathode Ray Terminal*, что соответствует уже не самой трубке, а устройству, на ней основанному (рис. 4).

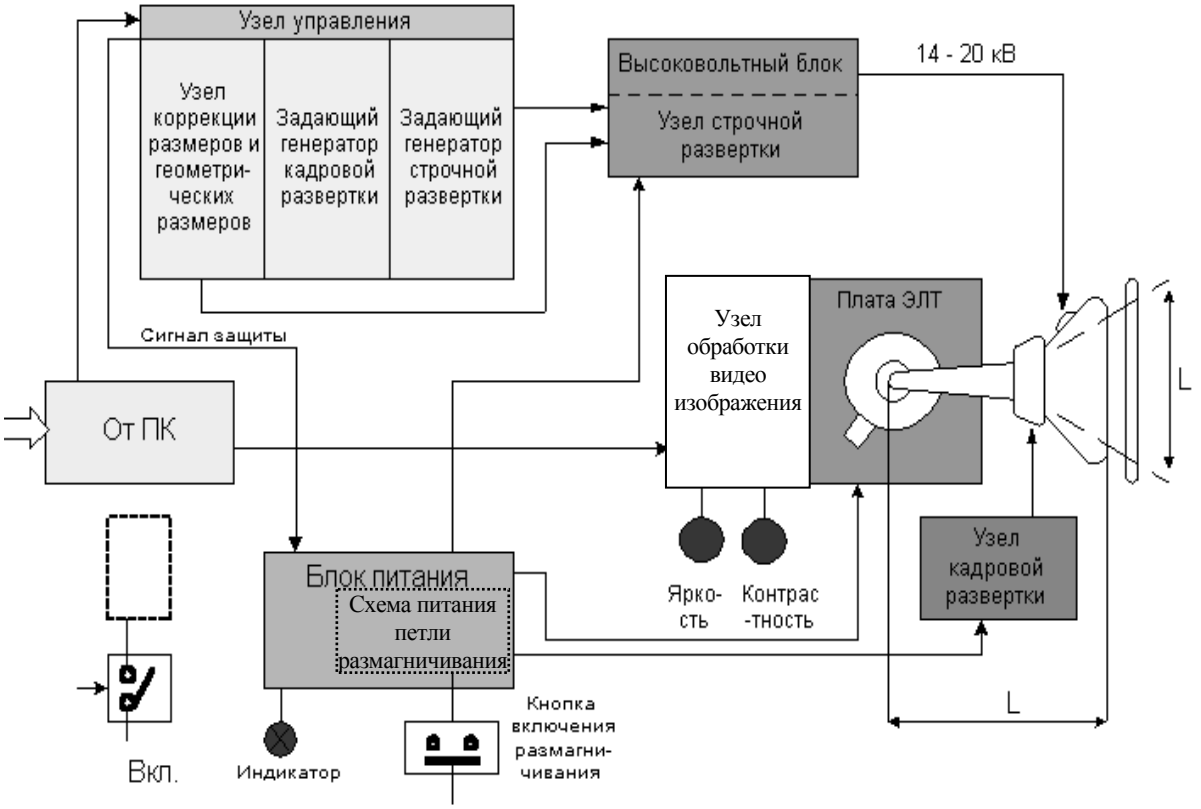


Рис. 4. Обобщенная структурная схема видеомонитора на ЭЛТ

Используемая в этом типе мониторов технология была разработана немецким ученым Фердинандом Брауном в 1897 г. и первоначально создавалась в качестве специального инструмента для измерения переменного тока, то есть для осциллографа.

Самый важный элемент монитора – кинескоп, называемый также электронно-лучевой трубкой (основные конструкционные узлы кинескопа показаны на рис. 5). Кинескоп состоит из герметичной стеклянной трубки, внутри которой находится вакуум, то есть весь воздух удален. Один из концов трубки (узкий и длинный) – это горловина, а другой – широкий и достаточно плоский – это экран. С фронтальной стороны внутренняя часть стекла трубки покрыта люминофором

(luminophor). Для создания изображения в ЭЛТ-мониторе используется электронная пушка, откуда под действием сильного электростатического поля исходит поток электронов. Сквозь металлическую маску они попадают на внутреннюю поверхность стеклянного экрана монитора, которая покрыта разноцветными люминофорными точками. Поток электронов (луч) может отклоняться в вертикальной и горизонтальной плоскостях, что обеспечивает последовательное попадание его на все поле экрана. Отклонение луча происходит посредством отклоняющей системы (рис. 5). Отклоняющая система состоит из нескольких катушек индуктивности, размещенных у горловины кинескопа. С помощью переменного магнитного поля две катушки создают отклонение пучка электронов в горизонтальной плоскости, а другие две – в вертикальной.

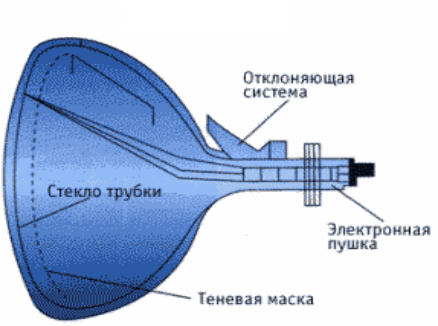


Рис. 5. Конструкция ЭЛТ-монитора

Как правило, в цветном *CRT*-мониторе используются три электронные пушки, в отличие от одной пушки, применяемой в монохромных мониторах, которые сейчас практически не производятся. Наши глаза реагируют на основные цвета: красный (*Red*), зеленый (*Green*) и синий (*Blue*) и на их комбинации, которые создают бесконечное число цветов. Люминофорный слой, покрывающий фронтальную часть электронно-лучевой трубки, состоит из очень маленьких элементов (настолько маленьких, что человеческий глаз их не всегда может различить). Эти элементы воспроизводят основные цвета палитры *RGB*. Электронный луч, предназначенный для красных люминофорных элементов, не должен влиять на люминофор зеленого или синего цвета. Для того чтобы добиться такого действия, используется специальная маска, чья структура зависит от типа кинескопов от разных производителей.

Электроника должна оптимизировать усиление сигнала и управлять работой электронных пушек, которые инициируют свечение люминофора, создающего изображение на экране. Оно воспроизводится в результате процесса, в ходе которого свечение люминофорных элементов инициируется электронным лучом, проходящим последовательно по строкам в следующем порядке: слева направо и сверху вниз на экране монитора. Чем быстрее электронный луч проходит по всему экрану, тем меньше будет заметно мерцание картинки. Считается,

что такое мерцание становится практически незаметным при частоте повторения кадров (проходов луча по всем элементам изображения) примерно 75 в секунду.

LCD-мониторы

LCD (*Liquid Crystal Display*) – жидкокристаллические мониторы) – сделаны из веществ, которые находятся в жидком состоянии, но при этом обладают некоторыми свойствами, присущими кристаллическим телам – это производные бензола, дефинила, стероидов, т.е. сложные органические соединения. Фактически это жидкости, обладающие анизотропией свойств (в частности, оптических), связанных с упорядоченностью в ориентации молекул. Это свойство характерно для твердых тел, а среди жидких им обладает лишь ограниченный класс веществ. Существуют три вида жидких кристаллов в зависимости от способа ориентации молекул: смектические, нематические, холестерические. Молекулы жидких кристаллов под воздействием электрического поля могут изменять свою ориентацию и вследствие этого изменять поляризацию светового луча, проходящего сквозь них.

Виды электрических явлений в жидких кристаллах, используемые для изменения их светопропускания: эффект динамического рассеивания; твист-эффект; эффект «гость-хозяин».

ЖК-монитор состоит из ячеек, показанных на рис. 6.

Для того чтобы ЖК-индикатор был хорошо виден, используют подсветку. Для предотвращения осаждения примесей на полупрозрачный электрод используют переменное питающее напряжение (фазовый метод питания).

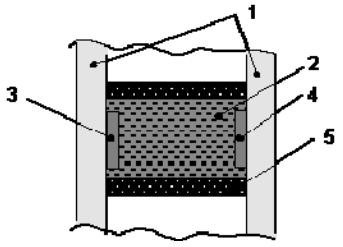


Рис. 6. Конструкция ЖК-ячейки: 1 – стеклянные пластины; 2 – жидкокристаллическое вещество; 3 – непрозрачный электрод; 4 – прозрачный электрод; 5 – склеивающие пластины

Параметры ЖК-индикаторов:

Контрастность……..80 – 100 %

Напряжение……..2 – 20 В

Ток……1 – 100 мкА

Частота управляющего напряжения…10 Гц – 1000 кГц.

Достоинства: простота конструкции, низкое энергопотребление, хороший контраст, совместимость с интегральными схемами по напряжению и току.

Недостатки: необходимость подсветки, узкий диапазон темпера-

тур от – 15 до +55 °С, изменение параметров с течением времени.

Экран *LCD*-монитора (рис. 7) представляет собой массив маленьких сегментов – ячеек (называемых пикселями), которыми можно манипулировать для отображения информации. *LCD*-монитор имеет несколько слоев, где ключевую роль играют две панели, сделанные из свободного от натрия и очень чистого стеклянного материала, называемого субстрат, или подложка. Между ними располагается тонкий слой жидких кристаллов. На панелях имеются бороздки, которые направляют кристаллы, сообщая им специальную ориентацию. Две панели расположены очень близко друг к другу.

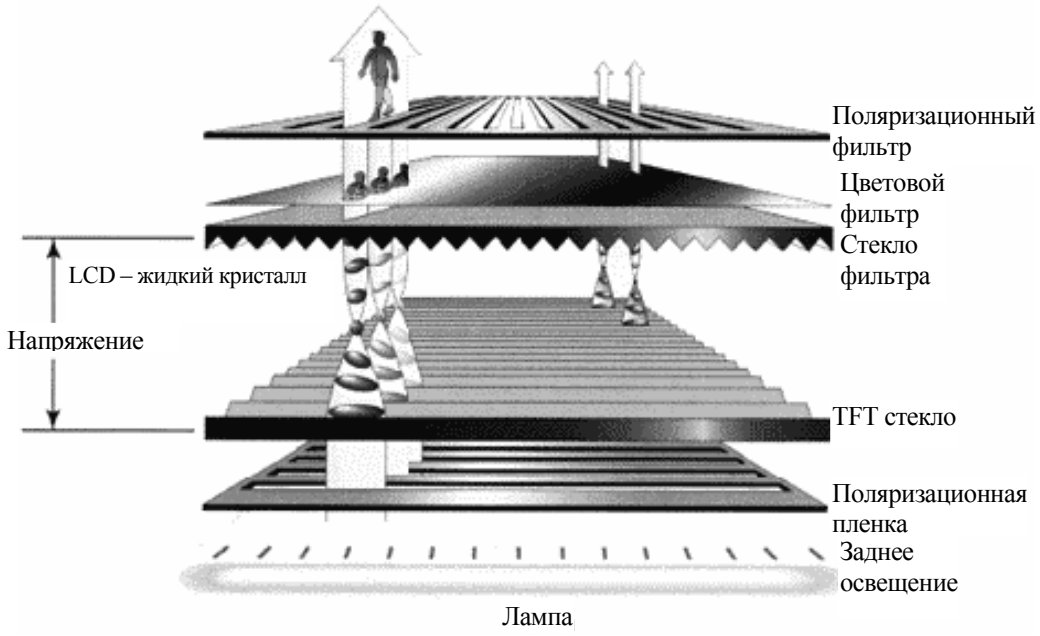
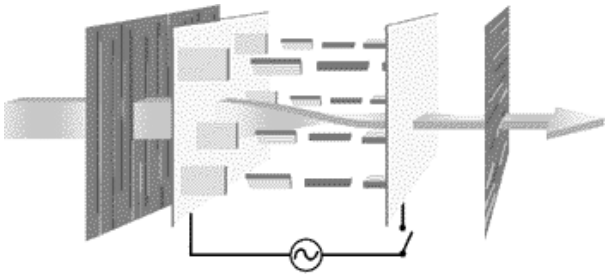


Рис. 7. Структурная схема экрана LCD-монитора

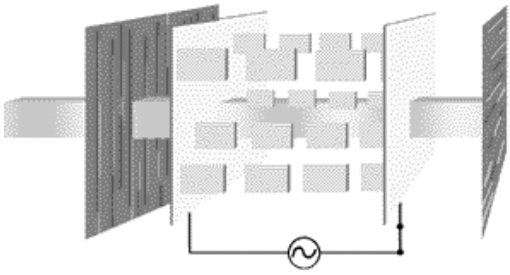
Плоскость поляризации светового луча поворачивается на 90° при прохождении одной панели. При появлении электрического поля молекулы жидких кристаллов частично выстраиваются вдоль поля и изменяют угол поворота плоскости поляризации света.

Поворот плоскости поляризации светового луча незаметен для глаза, поэтому возникла необходимость добавить к стеклянным панелям два других слоя, представляющих собой поляризационные фильтры. Они пропускают только ту компоненту светового пучка, у которой ось поляризации соответствует заданному, поэтому при прохождении поляризатора пучок света будет ослаблен в зависимости от угла между его плоскостью поляризации и осью поляризатора. Под действием электрического поля поворот вектора поляризации в жидком

кристалле происходит на меньший угол, поэтому второй поляризатор становится только частично прозрачным для излучения (рис. 8).



а)



б)

Рис. 8. Прохождение поляризованного луча через экран LCD-монитора: а – напряжение есть; б – напряжения нет

Если расположить большое число электродов, которые создают разные электрические поля в отдельных местах экрана (ячейки), то появится возможность, при правильном управлении потенциалами этих электродов, отображать на экране элементы изображения. Для вывода изображения необходима подсветка монитора сзади, для того чтобы можно было наблюдать изображение хорошего качества, даже если окружающая среда не является светлой.

Цвет получается в результате использования трех фильтров, которые выделяют из излучения источника белого света три основные компоненты. Комбинация трех основных цветов для каждой точки, или пикселя, экрана дает возможность воспроизвести любой цвет: можно создать несколько фильтров, расположенных друг за другом (что приводит к малой доле проходящего излучения), можно воспользоваться свойством жидкокристаллической ячейки – при изменении напряженности электрического поля угол поворота плоскости поляризации излучения изменяется по-разному для компонент света с разной длиной волны. Эту особенность можно использовать для того, чтобы отражать (или поглощать) излучение заданной длины волны

(возникает проблема необходимости точно и быстро изменять напряжение). Какой именно механизм используется, зависит от конкретного производителя: первый метод проще, второй эффективнее.

Матрицы ЖК-мониторов могут быть пассивными и активными. Термин "пассивная матрица" (*passive matrix*) появился в результате разделения монитора на точки, каждая из которых благодаря электродам может задавать ориентацию плоскости поляризации луча независимо от остальных. Изображение формируется (строка за строкой) путем последовательного подвода управляющего напряжения на отдельные ячейки. Из-за довольно большой электрической емкости ячеек напряжение на них не может изменяться достаточно быстро, поэтому обновление картинки происходит медленно. Дисплей с пассивной матрицей имеет много недостатков с точки зрения качества, потому что изображение не отображается плавно и дрожит на экране. Маленькая скорость изменения прозрачности кристаллов не позволяет правильно отображать движущиеся изображения; между соседними электродами возникает некоторое взаимное влияние, которое может проявляться в виде колец на экране; ограничен угол обзора экрана.

Существенно лучших результатов с точки зрения стабильности, качества, разрешения, гладкости и яркости изображения можно добиться, используя экраны с активной матрицей, которые, впрочем, стоят значительно дороже. В активной матрице используются отдельные усилительные элементы для каждой ячейки экрана, компенсирующие влияние емкости ячеек и позволяющие значительно уменьшить время изменения их прозрачности. Активная матрица (*active matrix*) имеет массу преимуществ по сравнению с пассивной: это лучшая яркость и возможность смотреть на экран даже с отклонением до 45° и более (т.е. при угле обзора 120° – 140°) без ущерба качества изображения, а дорогие модели *LCD*-мониторов с активной матрицей обеспечивают угол обзора в 160°, и есть все основания предполагать, что технология будет и дальше совершенствоваться. В случае с активной матрицей вы можете отображать движущиеся изображения без видимого дрожания, так как время реакции дисплея с активной матрицей – около 50 миллисекунд против 300 для пассивной матрицы, а качество контрастности лучше, чем у *CRT*-мониторов. Следует отметить, что яркость отдельного элемента экрана остается неизменной в течение времени между обновлениями картинки, а не представляет собой короткий импульс света, излучаемый

элементом люминофора *CRT*-монитора сразу после прохождения по этому элементу электронного луча. Именно поэтому для *LCD*-мониторов достаточной является частота регенерации 60 Гц. Благодаря лучшему качеству изображений эта технология также используется и в мониторах для настольных компьютеров, что позволяет создавать компактные мониторы, менее опасные для нашего здоровья.

В активной матрице к каждому электроду добавлен запоминающий транзистор, который может хранить цифровую информацию (двоичные значения 0 или 1), и в результате изображение сохраняется до тех пор, пока не поступит другой сигнал. Запоминающие транзисторы должны производиться из прозрачных материалов, что позволит световому лучу проходить сквозь них, а значит, транзисторы можно располагать на тыльной части дисплея на стеклянной панели, которая содержит жидкие кристаллы. Для этих целей используют пластиковые пленки, называемые "*Thin Film Transistor"* (*TFT*) т.е. тонкопленочный транзистор, толщина которого 1/10 – 1/100 микрона. Технология создания *TFT* очень сложна, при этом имеются трудности при достижении приемлемого процента годных изделий из-за того, что число используемых транзисторов очень велико (более миллиона).

Разрешение *LCD*-мониторов имеет фиксированное значение, называемое *native*, оно соответствует максимальному физическому разрешению *CRT*-мониторов.

Видеоадаптеры

Видеоадаптер, называемый также видеокартой, видеоконтроллером, обеспечивает обработку изображения и преобразование его к виду, необходимому для отображения на мониторе. Основные узлы и блоки видеоадаптера показаны на рис. 9.

Один из важных узлов видеоадаптера – *RAMDAC* (рис. 10). *RAMDAC* отвечает за формирование окончательного изображения на мониторе, то есть преобразует результирующий цифровой поток данных, поступающих от других элементов видеоадаптера, в уровни интенсивности, подаваемые на соответствующую электронную пушку (красную, зеленую, синюю) трубки монитора. *RAMDAC (Random Access Memory*) – это память с произвольной выборкой, а *DAC (Digital to Analog Converter)* – цифро-аналоговый преобразователь. Память в модулях *RAMDAC* построена на статических элементах, поэтому по быст-

родействию примерно соответствует кэш-памяти процессоров. *DAC* объединяет три параллельных канала, по одному на каждый цвет.

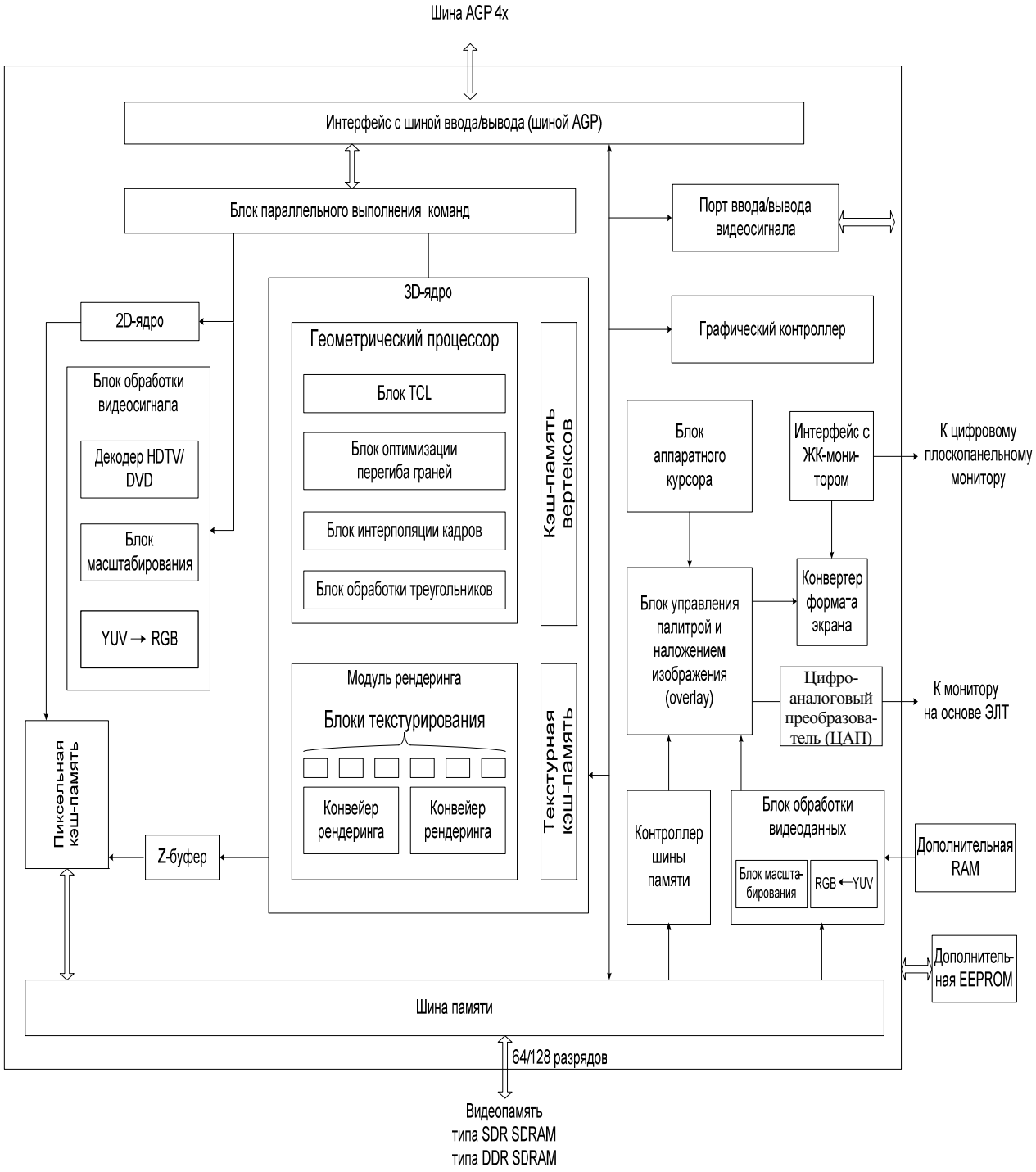


Рис. 9. Структурная схема видеоадаптера

Схемотехника *RAMDAC* быстро развивалась, и сегодня стандартным считается *RAMDAC,* обеспечивающий разрешение 1600×1200 точек при 32-битном цвете на частоте 75 – 85 Гц. Качество получае-

мого изображения в большой степени зависит от таких характеристик *RAMDAC*, как его частота, разрядность, время переключения с черного сигнала на белый и обратно, варианта исполнения (внешний или внутренний). Частота *RAMDAC* говорит о том, какое максимальное разрешение при какой частоте кадровой развертки сможет поддерживать видеоадаптер. Например, при разрешении 1024×768 точек и частоте кадровой развертки 70 Гц выводить *единичный* пиксел (с учетом времени на обратный ход луча по горизонтали и вертикали) необходимо примерно за 13 нс. Следовательно, в этом режиме *RAMDAC* должен поддерживать собственную частоту около 75 МГц. Современными можно считать *RAMDAC* с частотой не ниже 170 МГц. Разрядность *RAMDAC* говорит о том, какое цветовое пространство способен охватывать видеоадаптер. Большинство микросхем поддерживает представление 8 бит на каждый канал цвета, что обеспечивает отображение около 16,7 миллиона цветов.

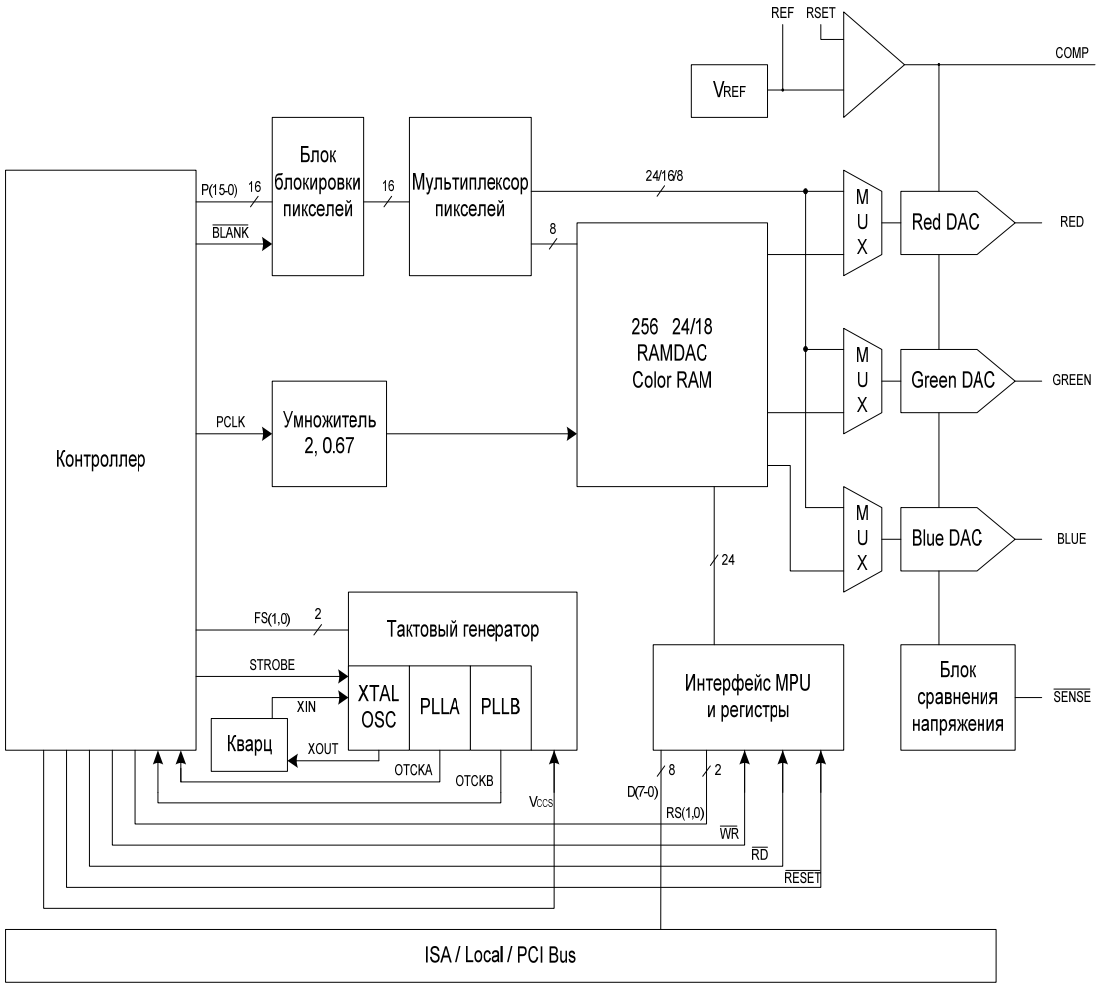


Рис. 10. Функциональная схема RAMDAC

Видеопамять служит для хранения изображения. От ее объема зависит максимально возможное полное разрешение видеоадаптера. На графических адаптерах массового применения использовать дорогую память типа *3D RAM*, *DR DRAM*, *CD RAM* невозможно по экономическим соображениям, поэтому большинство производителей используют модули с памятью типа *SDRAM* и *DDRDRAM*. При их совместной работе с мощными графическими чипсетами возникает ряд проблем.

Для кодирования графической и видеоинформации используются форматы *JPEG* и *MPEG*. *JPEG* – стандарт сжатия изображения с потерями, однако весьма эффективный.

Интерфейс AGP

Стандарт на *AGP* был разработан фирмой *Intel* для того чтобы, не меняя сложившийся стандарт на шину *PCI*, ускорить ввод/вывод данных в видеокарту, и, кроме этого, увеличить производительность компьютера при обработке трехмерных изображений без установки дорогостоящих двухпроцессорных видеокарт с большими объемами как видеопамяти, так и памяти под текстуры, z-буфер и т.п. Скорость передачи данных интерфейса AGP – до 532 Мбайт, которая обусловлена частотой шины – 66 МГц, возможностью отмены механизма мультиплексирования шины адреса и данных (на *PCI* по одним и тем же физическим линиям сначала выдается адрес, а потом данные). Интерфейс может пропустить 66 000 000,4 байта = 266 Мбайт/с. В режиме "×1" в качестве строба используется сам сигнал тактовой частоты. Для повышения пропускной способности шины в стандарт заложена возможность передавать данные с помощью дополнительных специальных сигналов, используемых как стробы, вместо сигнала *CLK* в обычном режиме (это режимы "×2" и "×4"). В режиме "×2" пропускная способность становится 66 000 000 × 2 × 4 байта = 532 Мбайт/с. В режиме "×4" (введен в спецификации 2.0) пропускная способность возрастает соответственно до 1064 Мбайт/с; "×8" – следующий вариант шины. Основная идея введения режимов – увеличение полосы пропускания до 8×4 = 32 байт за один такт системной шины. Это означает, что скорость передачи данных на шине возрастет до 2 Гбайт/с.

Интерфейс PCI-Express

Двадцать второго июля две тысячи второго года опубликована базовая спецификация протокола и сигнального уровня, а также базовая спецификация на форм-фактор и энергопотребление карт и разъемы. Стандарт *PCI-Express (PCI-E)* впервые был представлен на форуме *Intel* в сентябре 2003 года и, по словам разработчиков, должен заменить *AGP*, т.к. имеет значительные преимущества перед ним. Развитием стандарта *PCI-Express* занимается организация *PCI-Special Interest Group*.

Ни для кого не секрет, что в наше время идеальный внешний интерфейс является последовательным. Переход происходит медленно, но верно: сначала клавиатура и мышь, затем модем, через годы – сканеры и принтеры, видеокамеры, цифровые фотоаппараты – все они используют последовательные интерфейсы *USB, IEE1394, USB 2*. В настоящее время большинство новых интерфейсов переходит на последовательные соединения. Не за горами и беспроводные решения. Очевиден тот факт, что в наше время выгоднее заложить максимум функциональности в чип, нежели иметь дело с избыточными объемами контактов, кабелями с сотней проводов внутри, недешевыми пайкой, экранированием, разводкой и медью. Более подробно преимущества последовательных шин и интерфейсов рассмотрены в [5].

Преимущества PCI-Express.

• Высокая производительность: повышение пропускной способности версии "×1" как минимум вдвое по сравнению с *PCI*, возможность линейного наращивания производительности путем линейного расширения шины. Помимо этого, *PCI-Express* – это реально дуплексная шина;

• упрощение разводки периферии: стандартизация там, где ранее использовались всевозможные варианты *PCI , PCI-X* и др.; снижение комплексных затрат на разработку и внедрение систем;

• уровневая архитектура: основные затраты на развитие *PCI-Express* в дальнейшем ложатся лишь на разработку соответствующей обвязки, что дает возможность экономить на работе с прежним программным обеспечением;

• следующее поколение периферии: *PCI-Express* позволяет реализовать новые возможности обмена данными и мультимедийным контентом за счет изохронной природы передачи (т.е. разнесения отдельных частей сигнала по времени);

• простота использования: производить апгрейд и доработку систем устройствами *PCI-Express* станет значительно легче. Теперь появится возможность использовать *PCI-Express*-карты с "горячим" подключением.

Ключевые отличия PCI-Express от PCI

• новая шина последовательная, а не параллельная. Основные преимущества – снижение стоимости, миниатюризация, лучшее масштабирование, более выгодные электрические и частотные параметры (нет необходимости синхронизировать все сигнальные линии);

• спецификация разделена на целый стек протоколов, каждый уровень которого может быть усовершенствован, упрощен или заменен, не оказывая влияния на остальные. Это позволяет быстро и удобно разрабатывать адаптированные варианты специального назначения;

• в изначальной спецификации заложены возможности “горячей” замены карт;

• заложены возможности создания виртуальных каналов, гарантирования пропускной полосы и времени отклика, сбора статистики *QoS (Quality of Service* – качество обслуживания);

• заложены возможности контроля целостности передаваемых данных (*CRC*);

• заложены возможности управления питанием.

Архитектуру PCI-Express можно рассматривать послойно, в сравнении с адресной моделью *PCI.* Конфигурация *PCI-Express* – стандартна для устройств, определенных *plug-and-play* спецификациями *PCI*: программный уровень генерирует запросы чтения/записи, уровень транзакций транспортирует эти запросы к периферийным устройствам с помощью разделенного пакетного протокола. Для поддержания высокой производительности шины соединительный *(link)* уровень добавляет пакетам очередность и *CRC.* Базовый физический уровень состоит из двойного симплексного канала, осуществляющего функции приемной и передающей пары. *PCI-Express* использует программную модель шины *PCI.*

Физический протокол основан на последовательной передаче данных. На электрическом уровне каждое соединение использует низковольтную дифференциальную передачу сигнала (*LVDS*), приём и передача информации производятся каждым устройством *PCI-Express* по отдельной витой паре. Каждый канал состоит из двух дифференциаль-

ных сигнальных пар (необходимы только четыре контакта), сигнальный уровень составляет 0,8 вольт (рис. 11).

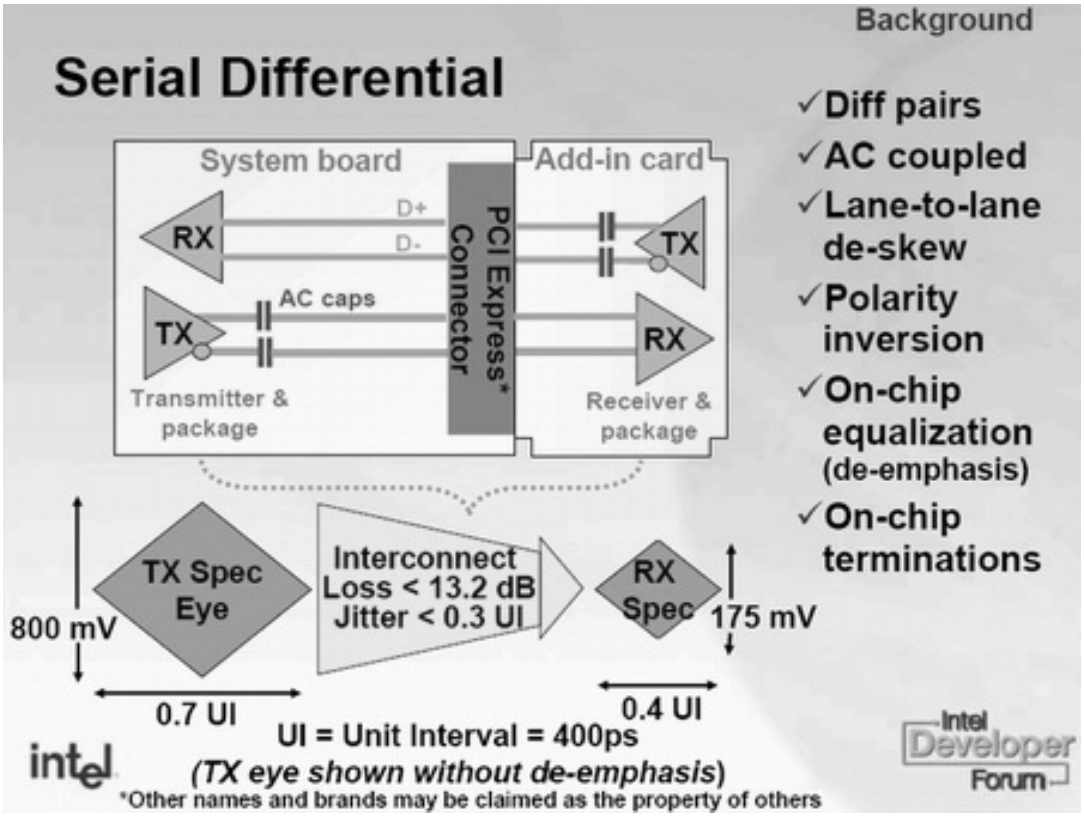


Рис. 11. Физический уровень интерфейса PCI-Express

Во всех высокоскоростных последовательных протоколах (например, *GigabitEthernet*) информация о синхронизации должна быть встроена в передаваемый сигнал. На физическом уровне *PCI-Express* использует ставший общепринятым метод кодирования 8 байт/10 байт, который позволяет поднять уровень помехозащищенности. Спецификация *PCI-Express* также предусматривает другой метод помехозащищенного кодирования – скремблинг (*scrambling*).

Пропускная способность интерфейса 2,5 Гбита (250 Мбайт) в секунду для одного канала в каждом направлении одновременно (полный дуплекс), однако, следует учесть, что эффективная скорость передачи данных за вычетом избыточного кодирования составляет 2 Гбита (200 Мбайт) ровно. Стандартизированы 1-, 2-, 4-, 8-, 16- и 32-канальные варианты (до 6,4 эффективных Гбайт в секунду при передаче в одну сторону и вдвое больше при передаче в обоих направлениях). Осуществля-

ется передача данных параллельно (но не синхронно) по всем доступным каналам: пропускная способность, с учетом двунаправленной передачи, для шин *PCI-Express* с разным количеством связей указана в таблице.

Число линий *PCI-Express*	Пропускная способность в одном направлении	Суммарная пропускная способность
1	250 Мб/с	500 Мб/с
2	500 Мб/с	1 Гб/с
4	1 Гб/с	2 Гб/с
8	2 Гб/с	4 Гб/с
16	4 Гб/с	8 Гб/с
32	8 Гб/с	16 Гб/с

Стандарт предусматривает и альтернативные носители сигнала, такие как оптические волноводы, и возможность распознавания и использования альтернативных (улучшенных) протоколов обмена, существует возможность динамического подключения и конфигурации устройств.

Самый простой вариант перехода на *PCI-Express* для стандартных по архитектуре настольных систем выглядит так, как показано на рис. 12. Однако в будущем логично ожидать появление некоего разветвителя *PCI-Express*. Тогда вполне оправданным станет и объединение северного и южного мостов.

Как правило, в настольных системах (на чипсетах *Intel* 915 и 925X) присутствуют один слот *PCI-Express* "×16" (предназначен для установки видеокарты; заменяет разъем) и до четырех слотов *PCI-Express* "×1". Серверные платы и платы, предназначенные для рабочих станций, имеют, кроме того, слоты *PCI-Express* "×4" и "×8". При этом в многоканальные разъёмы *PCI-Express* можно вставлять платы расширения, рассчитанные на меньшее число каналов.

Разъёмы, платы и карты. Разъём *PCI-Express* делится ключом на две части. Первая часть (та, что ближе к задней стенке корпуса) одинакова для всех разъёмов и предназначена для питания карты. Сюда подводятся напряжения 3,3 и 12 В. Спецификацией предусматривается подводка мощности 60 Вт.

Использование новых разъемов и других конструктивных возможностей, оговоренных спецификациями нового стандарта, позволяет говорить об увеличении энергопотребления конечных контроллеров до 75 Вт при токе до 5,5 А. Такие мощные контроллеры потребуют дополнительных мер по отводу тепла из корпуса, зато отпадет нужда в подводке разъемов дополнительного питания, которые так характерны для нынешнего поколения видеокарт "×8".

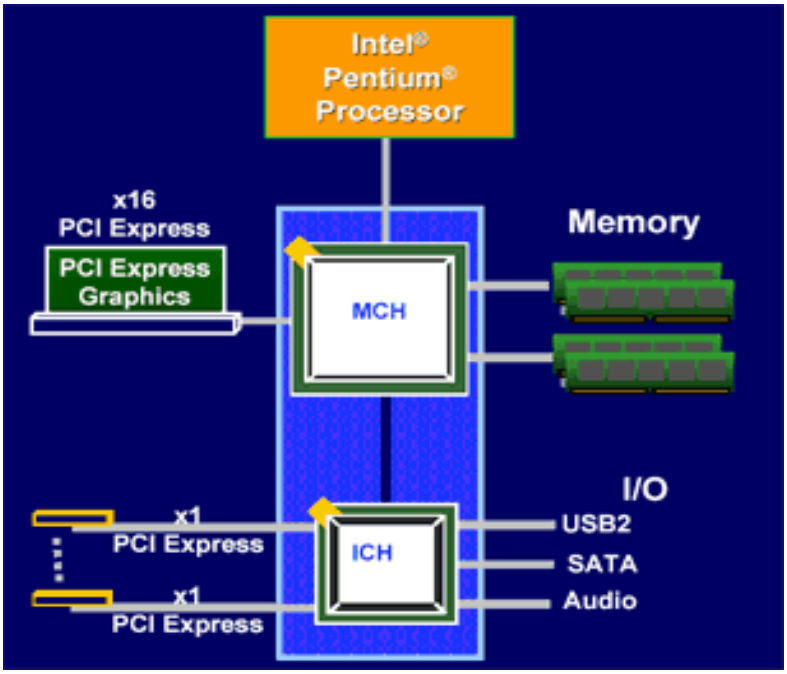


Рис. 12. Использование интерфейса PCI-Express в настольном ПК

Одними из первых устройств, которые стали массово выпускаться для шины *PCI-Express*, стали видеоадаптеры. Специалисты посчитали, что для видеоадаптера даже с запасом на будущее, подойдет 164-контактный разъём *PCI-Express* "×16". Эффективная пропускная способность *PCI-Express* "×16" заметно выше таковой у интерфейса AGP "×18": 3200 Мбайт/с против примерно 2000 Мбайт/с соответственно.

Интерфейсы графических адаптеров и мониторов

Первые графические интерфейсы были дискретные (*MDA, CGA, EGA*), интерфейсы второго поколения были аналоговыми (*VGA,*

SVGA), интерфейсы третьего поколения, предназначенные для жидкокристаллических (*LCD*) мониторов, - цифровые (*DVI, P&D, DFP*).

Дискретные интерфейсы имели уровень *TTL*, девятиконтактный разъем: шесть линий информационных контактов, одна линия – земля, две – синхронизация; они обеспечивали разное количество цветов: *MDA* – черно-белый, *CGA* – 16 цветов, *EGA* – 64 цвета.

Два стандарта – *VGA* и *SVGA* – формируют цветное изображение из трех основных цветов – *RGB* (красный, зеленый, синий) – с различным количеством оттенков (от 256 до 16 млн). Разрешение экрана определяется в зависимости от стандартов, которые ранее использовали девятиконтактный разъем *DB*-9*S*, а сейчас – пятнадцатиконтактный *VGA*-15. Введены дополнительные параметры логического сигнала идентификации типа монитора *ID*0…*ID*3, которые затем заменили двумя *SCL*, *SDA* (последовательная передача). Разъем *DB*-15 работает на $f \le 150$ МГц. Специалисты перешли к другому разъему *BNC*, где передача осуществляется не по витой паре, а по коаксиальным кабелям; количество кабелей 3 – 5, волновое сопротивление $p = 75$ Ом, l(длина) = 10 – 15 м.

Разъем *EVC* (затем разъем стал называться *P&D-A*) имеет четыре коаксиальных кабеля с частотой до 2 ГГц каждый; $p = 75$ Ом. Низкочастотный тридцатиконтактный разъем имеет три уровня реализации: полый, видео и *DDS*, мультимедиа и аудио-интерфейс; существует вариант разъема *EVS* для портативных ПК.

Цифровые интерфейсы

Интерфейс *Panel-Link* определяет протокол и структуру, используемые во всех цифровых интерфейсах, рассматриваемых ниже. В 1996 г. его спецификация (*FPDI*-2) была утверждена *VESA*.

Цифровой интерфейс имеет три канала передачи данных *Data*[0:3] и канал синхронизации *Clock*. Схема интерфейса приведена на рис. 13. В каналах используется дифференциальная передача сигналов с минимизацией переходов — протокол *TMDS (Transition Minimazed Differetial Signaling)*.

Каждый канал данных образован кодером, расположенным на видеокарте, линией связи и декодером, расположенным в дисплее. На вход кодера каждого канала поступают 8 бит кода яркости базисного цвета текущего пиксела. На вход кодера канала 0 поступают сигналы

строчной и кадровой синхронизаций, а на остальные каналы – дополнительные управляющие сигналы *CTL*[0:3], по паре на каждый канал.

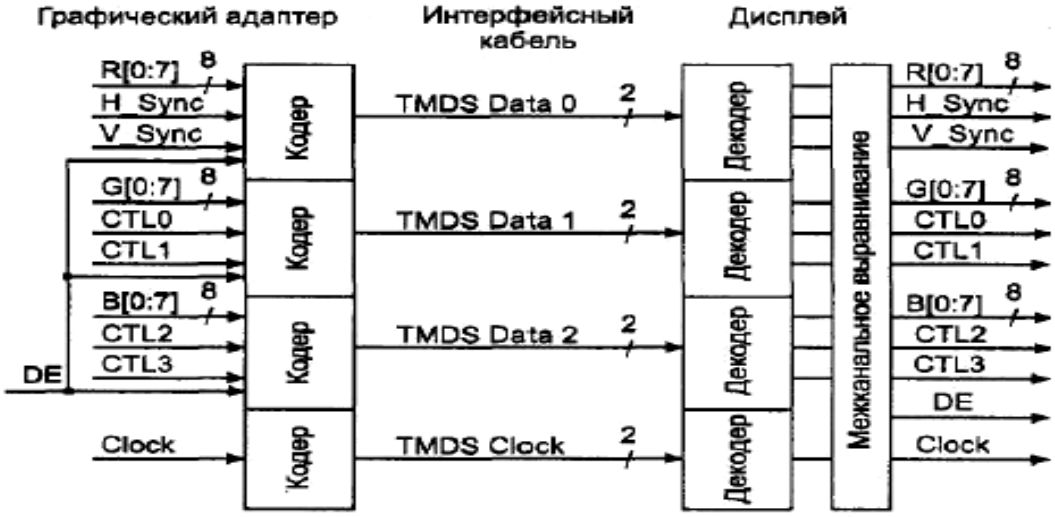


Рис. 13. Интерфейс Panel-Link

Кодеры преобразуют данные в последовательный код, для минимизации переключений 8 входных бит кодируются десяти битным символом, передаваемым по каналу последовательно. В зависимости от входного сигнала разрешения данных *DE* кодеры передают либо данные цветовых каналов, либо синхросигналы и управляющие биты. На приемной стороне сигналы декодируются и восстанавливаются в том же виде, в котором они поступали на входы кодеров.

Частота пикселов может достигать 165 МГц, интерфейс обеспечивает максимальное разрешение 1280×1024 (24 бита на пиксел).

Интерфейс *DVI (Digital Visual Interface)* – самый распространенный, мощный и универсальный. Он разработан группой *DDWG (Digital Display Working Group* – рабочая группа по цифровым дисплеям) в 1999 г. и предназначен для подключения дисплеев любого типа (ЭЛТ и матричных) к компьютеру.

Интерфейс имеет два варианта разъема:

1) чисто цифровой, который содержит канал синхронизации и три канала данных (*Data0*-2) и предусматривает способ повышения пропускной способности за счет обратных переходов;

2) цифровой с традиционными аналоговыми сигналами.

Дополнительно в интерфейс *DVI* входят сигналы интерфейса

VESA DDC2: DDC Data; DDC Clock; линия питания +5 В; сигнал *HPD* (*Hot Plug Detect*, отслеживание подключения/отключения дисплея). «Горячее» подключение обеспечивается также и механическими особенностями разъемов, поддерживающих требуемую последовательность соединения/рассоединения разных групп контактов.

Интерфейс и дисплеи с *DVI* должны обеспечивать стандартные (*VESA*) графические режимы, начиная от 640×480/60 Гц (частота пикселов 22,175 МГц). Его предел – 2048×1536 пикселов (частота 330 МГц). Интерфейс поддерживает сигнализацию управления энергопотреблением (*DPMS*). Вариант с аналоговыми сигналами – *VESA DD*2 – позволяет подключить аналоговые мониторы. Этот интерфейс является самым универсальным, мощным и наиболее распространенным среди цифровых интерфейсов.

Интерфейс *P&D* имеет пропускную способность 3,6 Гбит/с (450 Мбайт/с) при частоте пикселов 150 МГц и 24-битным числом (*True Color*) и разрешение 1280×1024 (24 бита на пиксел). Физические линии реализованы экранированными витыми парами. Передатчики являются дифференциальными коммутируемыми источниками тока (12 мА), входы дифференциальных приемников подтянуты нагрузочными резисторами 50 Ом к уровню питания +3,3 В, амплитуда сигнала 500 мВ, разъем у *P&D* такой же, как и у *EVC*. На контакты выведены цифровые сигналы. Интерфейсы можно рассматривать как комбинацию усеченных интерфейсов *DVI* и *EVC*. Интерфейс *P&D* дорогой и используется редко.

Интерфейс плоских дисплеев *DFP* (*Digital Flat Panel*, 1999 г.) использует дешевый разъем (рис. 14) типа *MDR (mini-D ribbon)* с ленточными контактами.

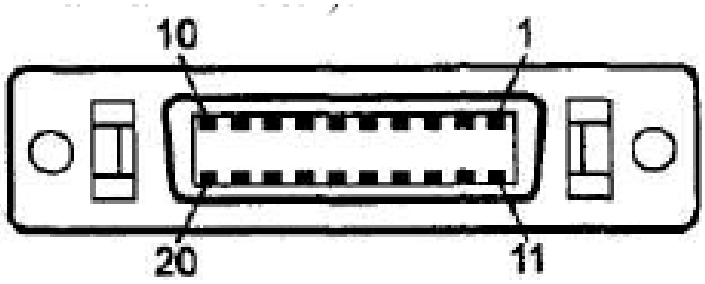


Рис. 14. Разъем интерфейса DFP: 1,10,11,20 – номера контактов

Интерфейс *DFP* имеет следующие сигналы:

- три пары сигналов для цифровых каналов данных;

- одну пару для цифрового канала синхронизации;
- сигнал обнаружения «горячего» подключения (*HPD*);
- питание (+5 В);
- канал *DDC2*.

Частота пикселов может достигать 85 МГц (для плоских панелей не требуется слишком высокая частота развертки). Интерфейс предназначен для режимов с разрешением до 1280×1024 (24 бита на пиксел). Достоинство интерфейса – низкая стоимость, недостаток – невозможность подключения мониторов с аналоговым входом.

Общее задание

Изучить работу видеосистемы компьютера и ее устройств и блоков (узлов), в том числе видео-сервис:

а) управление выводом на терминал (управление цветом);

б) управление курсором;

в) вывод символов;

г) вывод точек графики;

д) сдвиг экрана и страницы.

Варианты индивидуальных заданий

Используя прерывание 10h, написать программу вывода на экран заданной картинки [6, гл. 4]. Элементы картинки должны иметь разные цвета.

1. Четыре вертикальные полосы.
2. Пять диагональных полос из верхнего левого угла (главная диагональ).
3. Шесть диагональных полос из верхнего правого угла (побочная диагональ).
4. Шахматная доска, содержащая 16 квадратов или прямоугольников, т. е. по четыре вертикальных и горизонтальных полосы.
5. Семь горизонтальных полос.
6. Семь вертикальных полос.
7. Восемь диагональных полос из верхнего левого угла (главная диагональ).
8. Семь диагональных полос из верхнего правого угла (побочная диагональ).
9. Шахматная доска, содержащая 36 квадратов или прямоугольников, т. е. по шесть вертикальных и горизонтальных полос.
10. Пять диагональных полос из верхнего левого угла (главная диагональ).

11. Шесть диагональных полос из верхнего правого угла (побочная диагональ).

12. Шахматная доска, содержащая 25 параллелограммов и сформированная из пяти диагональных полос двух направлений (главная и побочная диагонали).

13. Девять диагональных полос из верхнего левого угла (главная диагональ).

14. Девять диагональных полос из верхнего правого угла (побочная диагональ).

15. Шахматная доска, содержащая 25 квадратов или прямоугольников, т. е. по пять вертикальных и горизонтальных полос.

16. Шахматная доска, содержащая 16 параллелограммов и сформированная из четырех диагональных полос двух направлений (главная и побочная диагонали).

Пример выполнения индивидуального задания

Разработать программу вывода на экран четырех разноцветных полос, используя прерывание 10h.

Текст программы:

```
program lab2;
uses dos, crt;
var i:integer; regs: registers;
{================================================}
procedure set_display_mode;
begin
  with regs do
  begin
    ah:=0;
    al:=$D;
    intr($10, regs);
  end;
end;
{================================================}
procedure show_horizontal_line(x: integer; y: integer; len: integer; width: integer; color: byte);
var x_tmp, len_tmp : integer;
begin
  with regs do
```

```
    while width <> 0 do
    begin
      x_tmp := x;  { save global x }
      len_tmp := len; {save global len }
      while len_tmp <> 0 do
      begin
        { set point }
        ah := $C; { function set point }
        al := color; { point color }
        cx := x_tmp; { row }
        dx := y; { column }
        intr($10, regs); { draw point }
        inc(x_tmp);
        dec(len_tmp);
      end;
      inc(y);
      dec(width);
    end;
  end;
  {=========================================}
  begin
    set_display_mode;
    for i:=0 to 2 do begin
      show_horizontal_line(0, 0, 320, 50, 15);
      show_horizontal_line(0, 50, 320, 50, 1);
      show_horizontal_line(0, 100, 320, 50, 4);
      show_horizontal_line(0, 150, 320, 50, 8)
    end;
    readkey;
  end.
```

Результат работы программы представляет четырех горизонтальные полосы равной высоты соответственно белого, синего, красного и зеленого цветов.

Вопросы для самоконтроля

1. Назначение дисплея.
2. Классификация дисплеев.

3. Параметры дисплеев на ЭЛТ.
4. Параметры дисплеев на ЖК.
5. Дисплей на ЭЛТ: структурная схема.
6. Дисплей на ЖК: способы запитки матрицы.
7. Цифровые интерфейсы дисплеев и протоколы.
8. Дисплей на ЖК: преимущества и недостатки.
9. Архитектура графической системы *IBM PC.*
10. Параметры видеоадаптера.
11. Способы представления цвета.
12. Интерфейс *AGP*: назначение, общие характеристики, разновидности и параметры.
13. Интерфейс *PCI-Express*: общие характеристики, разновидности и параметры.
14. Интерфейс *PCI-Express*: особенности функционирования и способы повышения производительности.
15. Структурная схема видеоадаптера.

Содержание отчета

1. Цель работы.
2. Краткие теоретические сведения, в том числе все схемы и ответы на вопросы для самоконтроля.
3. Индивидуальное задание.
4. Алгоритм.
5. Программа, содержащая комментарии.
6. Результаты работы программы (хотя бы в черно-белом варианте).

Лабораторная работа № 4

ДИАГНОСТИКА И ТЕСТИРОВАНИЕ ЭВМ, ЕЕ УСТРОЙСТВ И УЗЛОВ

Цель работы: изучение методов, методик и способов диагностики и тестирования компьютера, его устройств и узлов.

Методические указания

1. Изучить методики, способы и имеющиеся программы диагностики и тестирования компьютера, его устройств и узлов, используя литературу и Интернет.

2. Разработать методику диагностики и тестирования компьютера, его устройств и узлов.

3. Выбрать в Интернете программу для диагностики и тестирования компьютера, его устройств и узлов.

4. Выполнить программу диагностики и тестирования на первом компьютере.

5. Повторить эксперименты на другом компьютере, имеющем другую конфигурацию и параметры производительности.

6. Составить таблицу полученных результатов диагностики и тестирования для нескольких компьютеров (не менее двух).

Примечания

1. Желательно провести диагностику и тестирование с использованием другой диагностической программы.

2. В случае диагностики лишь одного компьютера необходимо провести диагностику и тестирование с использованием двух или более диагностических программ.

Содержание отчета

1. Цель работы.
2. Краткий обзор существующих программ диагностики и тестирования.
3. Детальное описание выбранной программы диагностики и тестирования.
4. Методика диагностики и тестирования.
5. Таблица полученных результатов.
6. Выводы.

Заключение

Выполнение представленных лабораторных работ позволяет студентам познакомиться с различными подсистемами ЭВМ, изучить их устройство и способы работы с ними, в том числе алгоритмы и программы, и увидеть проблемы и трудности, возникающие при этом.

Библиографический список

1. *Танненбаум, Э.* Архитектура компьютера / Э. Танненбаум. – 4-е изд. – СПб.: Питер, 2006. – 698 с. – ISBN 5-318-00298-6.

2. *Гук, М.* Аппаратные интерфейсы ПК : энциклопедия / М. Гук. – СПб.: Питер, 2002. – 528 с. – ISBN 5-94723-180-8.

3. *Он же*. Аппаратные средства IBM PC : энциклопедия / М. Гук. – СПб.: Питер, 2003. – 928 с. – ISBN 5-318-00047-9.

4. *Мюллер, С*. Модернизация и ремонт ПК / С. Мюллер. – 12-е изд. – М. : Вильямс, 2001. – 1184 с. – ISBN 5-8459-0167-7.

5. *Быков, В.И.* Интерфейсы периферийных устройств : учеб. пособие [электронный ресурс] / В.И. Быков. – Владимир: ВлГУ, 2007. – 126 с.

6. *Джордейн, Р*. Справочник программиста персональных компьютеров типа *IBM PC, AT* и *XT* / Р. Джордейн. – М.: Финансы и статистика, 1992. – 543 с.

7. *Хамахер, К.* Организация ЭВМ / К. Хамахер, З. Вранешич, С. Заки. – 5-е изд. – Спб.: Питер; Киев: BHV, 2003. – 848 с. – ISBN 5-8046-0162-8.

8. *Мелехин, В.С.* Вычислительные машины, системы и сети / В.С. Мелехин, Е.Г. Павловский. – М.: Академкнига, 2007. – 555 с. – ISBN 978-5-7695-4485-9.

9. *Максимов, Н.В.* Архитектура ЭВМ и вычислительные системы / Н.В. Максимов, Т.А. Патыка, И.И. Попов. – М.: Форум: Инфра-М, 2007. – 511 с. – ISBN 978-5-91134-105-3 (Форум). – ISBN 978-5-16-002970-2 (Инфра-М).

10. Вычислительные системы, сети и телекоммуникации: учеб. для вузов / А. П. Пятибратов [и др.]. – М.: Финансы и статистика, 2004. – 509 с. – ISBN 5-279 01804-X.

Оглавление

ВЫЧИСЛИТЕЛЬНЫЕ СРЕДСТВА
ИНФОРМАЦИОННЫХ СИСТЕМ

Методические указания к лабораторным работам

Составитель
БЫКОВ Валерий Ильич
Ответственный за выпуск – зав. кафедрой профессор В.Н. Ланцов

Подписано в печать 29.10.10.
Формат 60x84/16. Усл. печ. л. 2,56. Тираж 100 экз.
Заказ
Издательство
Владимирского государственного университета.
600000, Владимир, ул. Горького, 87.