ВВЕДЕНИЕ.

Компьютерная (или машинная) графика – интенсивно развивающаяся в последние десятилетия область применения средств вычислительной техники. Термин «компьютерная графика» обозначает обработку на ЭВМ графической информации, а так же ввод в ЭВМ исходных данных, первоначально представленных в графической форме, и вывод результатов в виде графических изображений.

Одной из основных областей приложения компьютерной графики являются системы автоматизированного проектирования (САПР). В них средства машинной графики используются для ввода исходной информации с целью описания структуры объекта проектирования или описания базового элемента при формировании библиотеки базовых элементов, а также для отображения (визуализации) промежуточных и окончательных результатов проектирования. Их можно использовать при проектировании зданий, самолетов, административных районов, радиоэлектронной аппаратуры. Наконец, при формировании и изготовлении твердых копий графической документации по спроектированному объекту используются методы и алгоритмы компьютерной графики.

Конечно же, САПР – это только одна из областей приложения возможностей машинной графики. Другими приложениями являются простые графические редакторы, компьютерные игры, мультипликация, реклама, видеоклипы. Развитие компьютерной графики создало новый изобразительный инструмент, привлекший внимание не только проектировщиков, но и дизайнеров, архитекторов, художников, модельеров.

Термин «компьютерная графика» предполагает, что изображение графических объектов и разнообразные действия с ними основываются на математических расчетах, реализуемых соответствующими алгоритмами компьютерной графики, базирующимися на фундаментальном математическом аппарате обыкновенной и проективной геометрии. Поэтому создание программной графической системы – чрезвычайно сложный и, одновременно,

увлекательный процесс, требующий от разработчика подобной системы значительных временных усилий и хорошей математической подготовки,

Средства машинной графики могут рассматриваться с нескольких позиций. Для создания графических изображений трехмерных объектов и анимационных сцен могут быть использованы широко известные высококачественные графические программные системы такие, как: 3DStudio MAX, CorelDraw, Fotoshop, Autocad, Компас. Пользователи таких систем не должны знать алгоритмов, используемых в этих системах при построения и преобразования графических объектов. Они должны владеть интерфейсом взаимодействия с соответствующей графической системой для того, чтобы ею правильно управлять в процессе создания изображения.

Есть другие люди, называемые, как известно, программистами, которые должны владеть основными принципами построения графического программного обеспечения, поскольку намерены сами его разрабатывать. Программисты, в свою очередь, могут либо использовать встроенные программные функции для вывода графических изображений на экран и выполнения разнообразных действий с ними, либо разрабатывать собственные программные функции на основе использования фундаментальных алгоритмов компьютерной графики.

Данное издание является первой частью учебного пособия «Компьютерная графика». В учебном пособии излагаются основные алгоритмы компьютерной графики, на основе которых могут быть созданы программные функции, обеспечивающие формирование статических и динамических изображений плоских и объемных тел в памяти компьютера и на экране дисплея. В книге приводятся форматы программных функций, встроенных в графическую библиотеку языка программирования СИ, которые реализуют рассматриваемые алгоритмы. Учебное пособие состоит из трех частей и содержит семь глав.

Первая часть «Видеосистема ПК. Геометрическое моделирование графических объектов и их преобразований» включает в себя первые три главы учебного пособия.

В главе 1 рассматривается организация видеосистемы персонального компьютера и принципы программного управления ею на различных уровнях управления системными ресурсами и для различного типа видеооборудования. Использованные литературные источники: [5, 10, 4, 14, 12, 9].

Математические основы выполнения геометрических операций над графическими объектами, в том числе для анимации, приводятся в главе 2 учебного пособия [14, 1, 11]. В этой главе описываются элементарные геометрические преобразования, а также способы реализации сложных геометрических преобразований на плоскости и в пространстве.

Глава 3 [11, 1, 7, 14] пособия разъясняет читателю, как строить геометрические модели трехмерных объектов, которые определяются координатами опорных точек и формализованным описанием связности этих точек. Приводятся методы построения геометрических моделей объектов, представленных в виде полигональных сеток, а также гладких кривых и поверхностей. В третьем разделе этой главы излагаются правила построения геометрических моделей типовых многогранников: конуса, пирамиды, цилиндра, призмы, сферы, правильных многогранников, тел вращения.

Вторая часть учебного пособия называется «Реалистическое изображение трехмерных объектов» и включает главы:

Глава 4. Изображение трехмерных объектов.

Глава 5. Удаление невидимых линий и поверхностей.

Третья часть учебного пособия, которая называется «Реалистическое изображение трехмерных объектов», содержит последние две главы:

Глава 6. Алгоритмы растровой графики.

Глава 7. Методы закраски.

ЛИТЕРАТУРА.

1. Аммерал А. «Машинная графика на языке Си. В 4-х томах». - М.: «Сол Систем», 1992.

2. Бронштейн И.П., Семендяев К.А. «Справочник по математике для инженеров и учащихся ВТУЗов». – Гостехиздат, 1948.

3. Гардан И. , Люка М. «Машинная графика и автоматизация конструирования» - М.: Мир, 1987г.

4. Григорьев В.П. «Видеосистемы ПК» - М.:Радио и связь, 1993г.

5. Касаткин А. И. «Профессиональное программирование на языке Си. Управление ресурсами: Справочное пособие». – Мн.: Высш. Шк., 1992.

6. Ньюмен У., Спрулл Р. «Основы интерактивной машинной графики». – М.: Мир, 1976г.

7. Пореев В. «Компьютерная графика».– СПб: БХВ–Петербург, 2002г.

8. Роджерс Д. «Алгоритмические основы машинной графики». – М.: Мир, 1989.

9. Справочник программиста и пользователя / Под ред. Шевчика А.Г., Демьянкова Т.В.. – М.: «Кварта», 1993.

10. Уилтон Р. «Видеосистемы персональных компьютеров IBM PC и PS/2. Руководство по программированию». - М.: Радио и связь, 1994

11. Фоли Дж., Дэм. А. вэн. «Основы интерактивной машинной графики: В 2-х книгах». – М.: Мир, 1985.

12. Фролов А.В. «Программирование видеоадаптеров CGA, EGA, SVGA» - М.: ДИАЛОГ- МИФИ, 1992г.

13. Шикин А.В., Боресков А.В. «Компьютерная графика. Полигональные модели». – М.: ДИАЛОГ-МИФИ, 2005г.

14. Шикин Е. В., Боресков А. В. «Компьютерная графика. Динамика, реалистические изображения». – М.: «ДИАЛОГ-МИФИ», 1995.

Глава 1.
ВИДЕОСИСТЕМА ПЕРСОНАЛЬНЫХ КОМПЬЮТЕРОВ.

Организация видеосистемы персональных компьютеров за относительно короткую историю своего развития претерпела серьезные видоизменения. Если первые модификации видеосистем обеспечивали формирование на экране черно-белого изображения, состоящего из ограниченного набора текстовых символов, то современное видеооборудование позволяет воспроизводить высококачественные разноцветные изображения, состоящие из очень маленьких графических элементов (picture element) - пикселов (pixel) или точек растра. По мере развития видеосистем увеличивалось количество воспроизводимых на экране цветов, уменьшались размеры пикселов, менялись способы доступа к программно-доступным компонентам видеооборудования и организация видеопамяти, появлялись новые, функционально более совершенные видеорежимы.

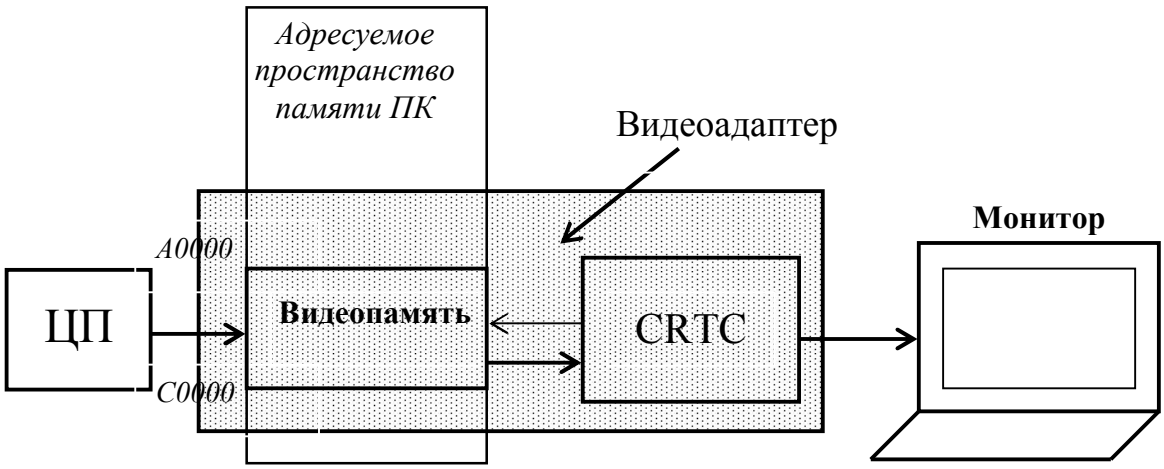
В этой главе рассматриваются логическая структура видеосистемы персонального компьютера и способы программного управления ее основными компонентами.

1.1. Организация видеосистемы.

Для формирования изображения на экране монитора современные персональные компьютеры (ПК) оборудованы видеосистемой.

Логическая структура видеосистемы ПК представлена на рисунке 1.1.

Из рисунка видно, что основу видеосистемы ПК составляют *монитор* и *видеоадаптер*, служащий для подключения монитора к ПК.



ЦП – центральный процессор,
CRTC – контроллер электронно-лучевой трубки.

Рис. 1.1. Логическая структура видеосистемы ПК.

Монитор.

Мониторы являются основным средством оперативного отображения вводимой в компьютер и выводимой из него информации.

В настоящее время применяются мониторы трех типов: композиционные, цифровые и аналоговые.

Композиционный монитор имеет одну входную линию, на которую от видеоадаптера поступает аналоговый сигнал, объединяющий в себе сигналы синхронизации, цвета и звука. Хотя видеоадаптеры CGA и могут генерировать такой составной сигнал, композиционные мониторы не нашли широкого применения из-за невысокого качества изображения.

Цифровой цветной RGB - монитор фирмы IBM имеет три входа для цвета (R - красный, G - зеленый, B - синий) и один вход (I) - для интенсивности (рисунок 1.2)., На эти четыре входные линии поступают цифровые сигналы, равные единице (цветовая компонента присутствует) или нулю (цветовая компонента отсутствует). Цифровой RGB –монитор воспроизводит $2^4 = 16$ цветов и поддерживается видеоадаптерами CGA и EGA.

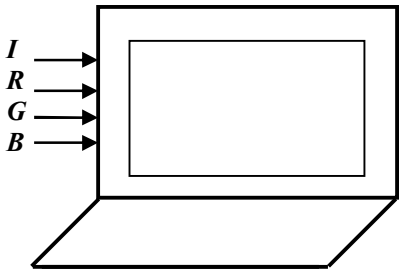


Рис. 1.2. Цифровой цветной RGB - монитор .

Улучшенный цифровой rgbRGB - монитор имеет шесть входов для цифрового видеосигнала (рисунок 1.3). Единичное значение сигнала, поступающего на первые три входа, обозначенные маленькими буквами (r, g, b), дает 1/3 интенсивности, а на вторые три входа (R, G, B) – 2/3 интенсивности, соответственно, красного, зеленого и синего цветов. Одновременное возбуждение сигнала на двух одноименных входах (например, r и R) даёт единицу интенсивности. Цифровой rgbRGB –монитор воспроизводит $2^6 = 64$ цвета и поддерживается видеоадаптером EGA.

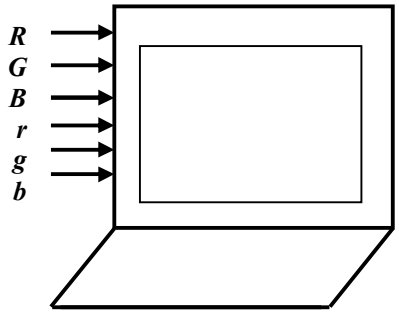


Рис. 1.3. **Цифровой rgbRGB монитор.**

Аналоговый монитор имеет три входные линии для аналоговых видеосигналов R, G и B (рисунок 1.4). Интенсивность луча в нем регулируется уровнем напряжения на соответствующей линии. Поэтому имеется возможность получения практически неограниченного числа цветов с плавными переходами между ними. Мониторы аналогового типа поддерживаются видеоадаптерами VGA, SVGA и более поздними модификациями.

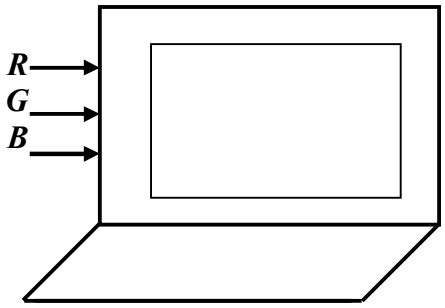


Рис. 1.4. Аналоговый монитор.

Современные персональные компьютеры в большинстве своем оснащены мониторами, использующими растровый способ отображения информации.

При использовании растрового метода отображения информации экран представляется двумерной матрицей (растром) близко расположенных друг к другу точек (пикселей). Луч, управляемый сигналами от контроллера электронно – лучевой трубки, движется по зигзагообразной траектории (рисунок 1.5), начиная с левого края верхней горизонтальной строки растра. Это движение называется прямым ходом луча.

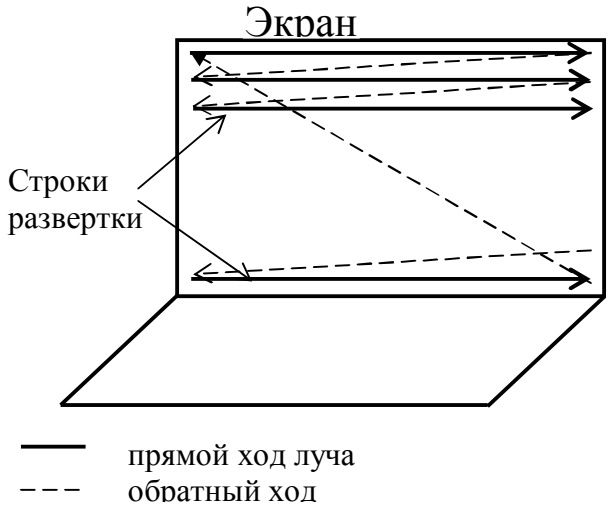


Рис. 1.5. Растровый способ формирования изображения.

Для отображения точки на экране (рисунок 1.6) в нужном месте интенсивность луча усиливается при его прямом ходе, что вызывает кратковременное свечение этой точки на покрытой фосфором поверхности экрана. После перемещения луча в крайнюю правую позицию он быстро

отклоняется вниз влево на начало следующей строки растра. Это движение называется горизонтальным обратным ходом луча, во время которого интенсивность луча гасится. Когда луч сформирует последнюю строку растра, он возвращается в верхний левый угол экрана (вертикальный обратный ход луча).

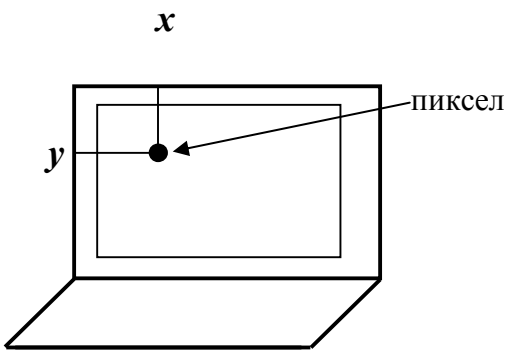


Рис. 1.6. Отображение точки на экране

За время одного кадра активизированные точки образуют отображаемую фигуру или текст. Для получения немерцающего изображения формирование растра повторяется с частотой 50-70 раз в секунду. Такое периодическое сканирование экрана, выполняемое контроллером ЭЛТ, называется регенерацией. Движением луча по горизонтали в течении прямого хода управляют сигналы строчной развертки, а по вертикали - сигналы кадровой развертки.

Мониторы не содержат программно-доступных элементов, а, следовательно, не могут напрямую управляться из программ. Программно-доступные элементы (регистры) располагаются в видеоадаптере. Содержимое этих регистров и определяет характеристики сигналов, управляющих монитором.

Видеоадаптер.

Видеоадаптер обеспечивает подключение монитора к компьютеру. Основным назначением видеоадаптера является преобразование содержимого

видеопамяти в сигналы, обеспечивающие формирование на экране изображения, адекватного содержимому видеопамяти.

Существует несколько различных типов видеоадаптеров, различающихся своими возможностями, аппаратным устройством и принципами работы с ними. Основные типы видеоадаптеров: MDA, CGA, EGA, VGA, SVGA.

Практически каждый видеоадаптер поддерживает несколько видеорежимов, отличающихся друг от друга размерами матрицы пикселов (разрешением) и размером палитры (количеством цветов, которые можно одновременно отображать на экране). Разные видеорежимы (даже одного видеоадаптера) имеют разную организацию видеопамяти и способы работы с ней.

Большинство видеоадаптеров строится по принципу совместимости со своими предшественниками. Так, видеоадаптеры EGA поддерживают все режимы видеоадаптера CGA. Поэтому любая программа, рассчитанная на работу с видеоадаптером CGA, будет также работать с видеоадаптером EGA. При этом, видеоадаптер EGA поддерживает еще и ряд своих собственных режимов. Аналогично, видеоадаптеры VGA поддерживает все режимы видеоадаптера EGA.

На печатной плате видеоадаптера располагаются следующие основные элементы, интересующие программиста видеосистемы:

- контроллер электронно – лучевой трубки (ЭЛТ);
- программно – доступные регистры;
- видеопамять (VideoRAM);
- ПЗУ знакогенератора.

Контроллер ЭЛТ (Cathode Ray Tube Controller - CRTC) осуществляет считывание содержимого видеопамяти, преобразование его в видеосигналы и передачу сформированных сигналов цвета и яркости вместе с сигналами синхронизации в монитор. Помимо этой основной функции CRTC реализует несколько дополнительных функций:

- определение формы и позиции аппаратного курсора;
- выбор выводимой на экран части видеобуфера;
- интерпретацию ASCII-кодов символов и поиск соответствующих им пиксельных матриц, определяющих разложение изображения символа по строкам растра, в таблице знакогенератора ПЗУ;
- задание аппаратного подчеркивания;
- восприятие сигналов светового пера (в персональных компьютерах световое перо, практически, не применяется: в адаптере MDA не использовалось, т.к. люминофор экрана такого монитора не даёт достаточно яркой вспышки при попадании на него электронного луча; в CRTC видеоадаптеров VGA и выше поддержка светового пера отсутствует, поскольку это устройство не нашло широкого применения);

Программное управление видеоадаптером и параметрами генерируемых им сигналов осуществляется через группу программно-доступных регистров. Управляющие *регистры видеоадаптера* делят на несколько групп, в число которых входят:

- регистры CRTC;
- регистры атрибутов: регистр управления режимом (порты: 3B8H – MDA, 3D8H – CGA и выше); регистр выбора цвета (порты: 3B9H – MDA, 3D9H – CGA и выше); регистр состояния (порты: 3BAH – MDA, 3DAH – CGA и выше).
- регистры секвенсера;
- регистры графического контроллера;
- регистры цифро-аналогового преобразователя (Digital to Analog Converter - DAC), регистры палитры;
- внутренние регистры.

Рассмотрим назначение программно-доступных *регистров контроллера ЭЛТ* видеоадаптеров разного типа.

Основные функциональные узлы контроллера ЭЛТ **видеоадаптера MDA** реализованы на микросхеме 6845 фирмы Motorola. Эта микросхема содержит 19 внутренних однобайтовых регистров. Порты доступа к этим регистрам:

3B4H – регистр адреса (индексный регистр), 3B5H – регистр данных. Для доступа к любому регистру контроллера ЭЛТ необходимо в регистр адреса (индексный регистр), доступный через порт 3B4H, записать номер этого регистра. После этого производиться обращение к нужному регистру через порт 3B5H. Назначение регистров CRTC приведено в табл.1.1.

Таблица 1.1.

Регистры контроллера электронно – лучевой трубки

Номер регистра	*Назначение*	*Доступ*
00H	Общее число символов по горизонтали	Запись
01H	Число отображаемых символов в строке	Запись
02H	Горизонтальная позиция синхронизации	Запись
03H	Ширина горизонтальной синхронизации в символах	Запись
04H	Общее число линий по вертикали	Запись
05H	Подстройка по вертикали (число линий сканирования)	Запись
06H	Количество отображаемых рядов	Запись
07H	Вертикальная позиция для синхронизации (число символьных рядов)	Запись
08H	Подстрочный режим («через строку»)	Запись
09H	Максимальный адрес линии сканирования для символа	Запись
0AH	Начало курсора (в линиях сканирования)	Запись
0BH	Конец курсора (в линиях сканирования)	Запись
0CH	Старший байт начального адреса	Запись
0DH	Младший байт начального адреса	Запись
0EH	Старший байт позиции курсора	Чтение/Запись
0FH	Младший байт позиции курсора	Чтение/Запись
10H	Старший байт светового пера	Чтение
11H	Младший байт светового пера	Чтение

Регистры 00H-09H содержат параметры сигналов горизонтальной и вертикальной развертки, управляющих формированием изображением на экране монитора. Содержимое регистров 00-09H инициализирует процедура начальной загрузки BIOS при включении питания или программная процедура установки видеорежима. Хотя эти регистры доступны для записи, экспериментировать с ними программисту не рекомендуется: случайная

ошибка при занесении в эти регистры может привести к повреждению монитора.

Регистры 0AH и 0BH определяют форму курсора (начальную и конечную строку развертки в символьной позиции).

Регистры 0CH и 0DH задают начальный адрес выводимого на экран блока видеопамяти. BIOS инициализирует эти регистры на ноль. Начальный адрес задаётся как относительный адрес (смещение) в видеобуфере. Регистры 0CH и 0DH играют важную роль при многостраничной организации видеопамяти.

Регистры 0EH и 0FH определяют относительный адрес символьной позиции видеобуфера, в которой находится курсор. Эти регистры допускают как считывание так и запись, благородя чему курсор можно установить в любую символьную позицию.

Регистры 10H и 11H, фиксируют позицию светового пера.

Контроллер CRTC **видеоадаптера CGA** также реализован на микросхеме 6845, поэтому способы программирования контроллера в адаптерах MDA и CGA одинаковые. Однако, в адаптере CGA индексный регистр отображен на порт 3D4H, а регистры данных – на порт 3D5H. Адрес порта доступа к индексному регистру хранится по адресу 0040:0063H области данных BIOS.

Для **видеоадаптера EGA** фирма IBM разработала БИС, которая выполняет функции CRTC. В ней ради совместимости сохранены регистры микросхемы 6845 и введены дополнительные регистры, обеспечивающие более гибкое программирование. Кроме этого, контроллер CRTC в адаптере EGA реализует больше управляющих функций. Например, он может сформировать сигнал прерывания по началу вертикального обратного хода луча, сообщая о возможности доступа к видеобуферу без помех.

Отметим, что в видеоадаптерах EGA, VGA и более поздних модификациях существенно упрощен доступ к регистрам данных CRTC. Так, команда вывода 16-битного слова *OUT DX, AX* позволяет записать в регистр

CRTC, номер которого указан в AL, байт данных, хранящийся в AH (DX - адрес порта доступа к индексному регистру CRTC). При выполнении таких команд OUT автоматически запрещаются все прерывания процессора.

Доступ к регистрам CRTC видеоадаптера EGA осуществляется через порты 3D4H (индексный регистр) и 3D5H (регистр данных).

Регистры CRTC **видеоадаптера VGA** образуют надмножество регистров CRTC адаптера EGA. Адресация регистров осуществляется через порты 3D4H (индексный регистр) и 3D5H (регистр данных). Все спецификации регистров адаптера EGA перенесены и в адаптер VGA, что обеспечивает их полную программную совместимость. Для образования растров из 400 и 480 строк в регистры добавлено несколько полей. Поддержка светового пера в VGA отсутствует.

Все регистры данных CRTC VGA допускают как считывание, так и запись. Регистры 00-07H, определяющие характеристики сигналов развертки, можно защитить от записи, установив в “1” бит 7 регистра конца обратного хода луча.

Форматы *регистров атрибутов* приведены в таблицах 1.2, 1.3, 1.4.

Таблица 1.2.

Формат регистра управления режимом.

Номер бита	Значение бита	Описание
0	1 0	80 x 25 текстовый режим 40 x 25 текстовый режим
1	1 0	320 x 200 графический режим текстовый режим
2	1 0	запрещение цвета разрешение цвета
3	1 0	Разрешение видеосигнала Запрещение видеосигнала (гашение экрана)
4	1 0	640 x 200 графический режим все остальные режимы
5	1 0	Мерцание Нет мерцания
6, 7	0	Не используются.

Таблица 1.3.

Формат регистра выбора цвета.

Номер бита	Описание
0 – 2	RGB для фона
3	интенсивность
5	*=1* – палитра «1» (синий, красный, белый). *=0* – палитра «0» (зеленый, красный, коричневый)

Таблица 1.4.

Формат регистра состояния.

Номер бита	Значение бита	Описание
0	1	Отображение разрешено (видеопамять доступна)
1	1	Установлен триггер светового пера
2	1	Световое пера подключено
3	1	Обратный вертикальный ход луча (доступ к видеопамяти возможен в течение последующего интервала длительностью 1,25 мс)

Секвенсер генерирует внутренние сигналы для адресации видеопамяти. Этот узел содержит в своем составе один индексный регистр и 5 регистров данных. Доступ к регистрам секвенсера осуществляется через порты 3C4H (индексный регистр) и 3C5H (регистр данных). Формат регистров данных секвенсера приведен в табл. 1.5.

Таблица 1.5.

Регистры данных секвенсера

Номер регистра данных секвенсера	Назначение
0	Регистр установки секвенсера
1	Регистр режима тактирования
2	Регистр маски двоичных плоскостей видеопамяти
3	Регистр выбора карты символов
4	Регистр режима памяти

Графический контроллер управляет обменом данными между видеопямятью и процессором. Этот узел содержит в своем составе один индексный регистр и 5 регистров данных. Доступ к регистрам графического контроллера осуществляется через порты 3CEH (индексный регистр) и 3CFH (регистр данных). Формат регистров данных графического контроллера приведен в табл. 1.6.

Таблица 1.6.

Регистры данных графического контроллера.

№ регистра	Назначение регистра	Начальное значение
00h	Установки/сброса	00h
01h	Разрешения установки/сброса	00h
02h	Сравнения цвета	00h
03h	Сдвига данных/выбора функции	00h
04h	Выбора считываемого банка	00h
05h	Режима	00h
06h	Вспомогательный	00h
07h	Безразличного цвета	0Fh (16-ти цветный режим) 01h (2-х цветный режим)
08h	Двоичной маски	FFh

Третьей главной составляющей видеоадаптера компьютера является *видеопамять.*

Видеопамять предназначена для хранения отображаемой на экране информации. Эта память, допускающая операции чтения и записи, носит название VideoRAM. Каждый бит или группа битов в видеобуфере определяет цвет и яркость конкретного пиксела на экране. Для образования на экране немерцающего изображения контроллер ЭЛТ периодически (с частотой 50-70Гц) считывает содержимое видеобуфера и преобразует его в видеосигналы, управляющие формированием изображения на экране.

Видеобуфер является частью адресуемого пространства памяти персонального компьютера. Такое местоположение видеопамяти обеспечивает два важных преимущества.

Во-первых, считывание данных из видеопамяти эквивалентно чтению информации на экране.

Во-вторых, высокое быстродействие отображения информации на экране монитора. В этом случае процессор может записывать коды цветов пикселей в видеопамять с такой скоростью, с какой процессор может записывать данные в RAM (оперативную память компьютера), т.е. сотни тысяч байт в секунду. Благодаря регенерации экрана с частотой 50-70Гц любое изменение программой содержимого видеобуфера вызывает почти немедленное изменение изображения на экране. Для сравнения: терминалы, ранее используемые в качестве устройств отображения, принимали информацию от вычислительной машины со скоростью всего около 1000 байт в секунду. Высокая производительность вывода информации на экран особенно необходима в графическом режиме при формировании движущихся изображений.

Емкость видеобуфера в различных видеосистемах варьируется от 4Кбайт (MDA) до 256Кбайт (VGA). Очевидно, чем больше емкость видеобуфера, тем более сложные изображения можно сформировать на экране монитора. Кроме того, в большинстве видеосистем емкость видеобуфера достаточна для того, чтобы хранить больше данных, чем требуется для заполнения экрана. Следовательно, в любой момент времени на экране наблюдается только часть информации из видеобуфера. Это обстоятельство позволяет реализовать множество интересных и удобных приемов управления выводом на экран, основанных на многостраничной организации видеопамяти.

Видеосистема (за исключением адаптера MDA) может работать в двух основных режимах: текстовом и графическом.

В текстовом режиме экран разделяется на отдельные позиции, в каждую из которых выводится один символ. В видеобуфере каждой символьной позиции соответствует 2 байта. Байт с четным адресом содержит ASCII-код

символа, байт с большим нечетным адресом - атрибуты символа (цвет, яркость, мерцание). В текстовом режиме контроллер ЭЛТ преобразует ASCII-коды символов в пиксельную форму с использованием встроенного в видеосистему знакогенератора. Знакогенератор содержит набор пиксельных матриц, отображающих разложение изображения символов по строкам развертки.

В текстовом режиме имеются ограниченные средства псевдографики. Для этого в расширенный код ASCII включены символы, представляемые отрезками горизонтальных и вертикальных прямых, угловыми элементами и т.п. Они считаются текстом и обрабатываются как обычные литеры. Путем комбинирования располагаемых встык символов псевдографики, можно создавать несложные графические формы (псевдографический режим). Пример использования псевдографического режима: текстовый редактор LEXICON.

Для формирования высококачественных графических изображений на экране монитора видеосистема ПК предусматривает наличие графического режима.

В графическом режиме цветовое значение каждого пиксела хранится в видеопамяти как один или несколько бит. При этом, для каждого типа видеоадаптера и каждого режима его работы имеется взаимно однозначное соответствие между битами в видеопамяти и пикселами на экране.

Если в видеобуфере пиксел кодируется n битами, одновременно на экране можно наблюдать 2^n цветов. Чем больше пикселов на экране и чем больше бит в видеобуфере отведено на каждый пиксел, тем более сложные и качественные изображения можно сформировать на экране.

Формат видеобуфера в графическом режиме совершенно отличается от формата текстового режима. Более того, разные графические режимы даже одного видеоадаптера имеют разную организацию видеопамяти и способы работы с ней. Рассмотрим организацию видеопамяти в различных графических режимах работы видеоадаптеров.

Адаптер CGA.

Этот адаптер поддерживает два графических режима с различной разрешающей способностью (табл. 1.7).

Таблица 1.7.

Графические режимы видеоадаптера CGA.

Номер режима	*Разрешение экрана*	*Бит на пиксел*	*Количество цветов*
4, 5	320x200	2	4
6	640x200	1	2

В двухцветном режиме 640 x 200 цветовое значение пиксела в видеопамяти кодируется одним битом. Если бит равен нулю, соответствующий ему пиксел выводится на экран, как черный, а если бит равен единице - цвет пиксела определяется значением разрядов 0 - 3 регистра выбора цвета (порт 3D9H). По умолчанию принимает белый цвет. Каждый байт видеопамяти в этом режиме содержит информацию о 8 пикселах.

В четырехцветном режиме 320 x 200 цветовое значение пиксела кодируется в видеобуфере двумя битами. Цвет пиксела в этом режиме определяется:

а) значением соответствующих двух бит в видеобуфере;

б) значением разрядов 0 – 3 регистра выбора цвета (порт 3D9H), если двухбитовый код пиксела в видеопамяти равен «*00»*;

в) значением бита 2 регистра управления режимом (порт 3D8H) и значением разряда 5 регистра выбора цвета (порт 3D9H), если двухбитовый код пиксела в видеопамяти не равен *«00»* (табл. 1.8).

Каждый байт видеопамяти в этом режиме содержит информацию о четырех пикселах.

Таблица 1.8.

Палитра цветов видеоадаптера CGA.

Значение управляющих разрядов регистров видеоадаптера	*Цвет*
Бит 2 регистра управления режимом = 0, Бит 5 регистра выбора цвета = 0, Двухбитовый код пиксела в видеопамяти:	
01 (1)	Зеленый
10 (2)	Красный
11 (3)	Желтый
Бит 2 регистра управления режимом = 0, Бит 5 регистра выбора цвета = 1, Двухбитовый код пиксела в видеопамяти:	
01 (1)	Голубой
10 (2)	Фиолетовый
11 (3)	Белый
Бит 2 регистра управления режимом = 1, Двухбитовый код пиксела в видеопамяти:	
01 (1)	Голубой
10 (2)	Красный
11 (3)	Белый

В видеобуфере емкостью 16 Кбайт данные хранятся в виде двух блоков: информация 100 четных строк развертки занимает 8000 байт с начальным адресом B800:0000, а 100 нечетных строк развертки занимают 8000 байт с начальным адресом B800:2000H (рисунок 1.7). Для нумерации четных и нечетных строк развертки требуется 7 бит.

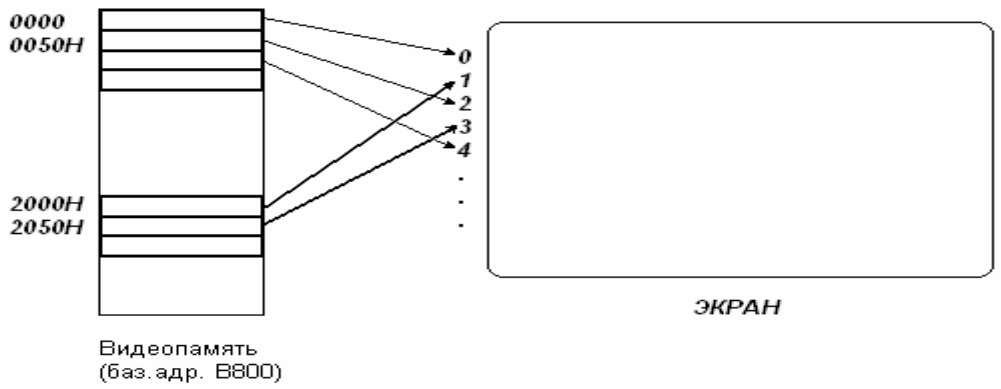


Рис. 1.7. Организация видеопамяти адаптера CGA.

Адаптер EGA

Когда этот адаптер программируется на эмуляцию графических режимов адаптера CGA, пикселы в видеобуфере отображаются точно так же, как в адаптере CGA.

Видеоадаптеры EGA поддерживают четыре графических режима. Список основных графических режимов видеоадаптера EGA приведен в табл. 1.9.

Таблица 1.9.

Графические режимы видеоадаптера EGA.

Номер режима	*Разрешение экрана*	*Бит на пиксел*	*Количество цветов*
0DH	320x200	1	16
OEH	640x200	1	16
0FH	640x350	1	2
10H	640x350	1	16

В естественных видеорежимах адаптера EGA строки пикселов отображаются в видеопамяти линейно аналогично линейному отображению текстовых строк в текстовом режиме.

В видеоадаптере EGA применяется наиболее распространенная в видеоустройствах схема образования цвета: любой цвет может быть представлен как смешение трех основных цветовых компонент – красной R (Red), зеленой G (Green) и синей B (Blue) с заданными интенсивностями. Для образования палитры, состоящей из $16 = 2^4$ различных цветов, требуется 4 бита видеопамяти.

В этих режимах видеобуфер емкостью до 256 Кбайт состоит из четырех параллельных битовых плоскостей (рисунок 1.8). Плоскости считаются параллельными в том смысле, что они занимают один и тот же диапазон адресного пространства. Любому пикселу на экране соответствует один бит на каждой из битовых плоскостей. Обращение к битовым плоскостям осуществляется через регистры секвенсера и графического контроллера.

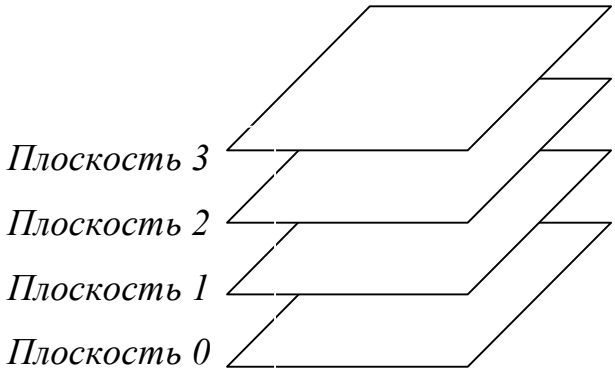


Рис. 1.8. Организация видеопамяти адаптера EGA

Код цвета пиксела определяют адрес байта видеопамяти и позиция бита внутри байта. Цветовое значение пиксела дают 4 бита в одной и той же позиции байтов видеопамяти с одинаковым адресом из четырех двоичных плоскостей. Эти четыре байта в совокупности кодируют цвет 8 соседних пикселов на экране. Поэтому битовые плоскости видеопамяти называют цветовыми плоскостями. Такая кодировка позволяет получить $2^4 = 16$ цветов, отображаемых на экране дисплея. Таким образом, в естественных режимах работы адаптера EGA цветовое значение пиксела в видеопамяти кодируется одним битом. То есть один байт видеопамяти содержит информацию о восьми пикселах.

Четырехбитовый код цвета пиксела в видеопамяти EGA-адаптера – это номер одного из 16 специальных внутренних регистров адаптера, называемых *регистрами палитры*. Шестиразрядные регистры палитры EGA-адаптера имеют формат: r g b R G B,

где маленькие буквы определяют половинную интенсивность цветовой компоненты (r - красный, g – зеленый, b - синий), а большие буквы – нормальную интенсивность соответствующей цветовой компоненты. Таким образом, для каждой цветовой компоненты возможны четыре градации яркости (rR, gG, bB):

00 – цветовая компонента отсутствует,

01 – слабая яркость,

10 – нормальная интенсивность,

11 – яркий цвет.

Следовательно, EGA-адаптер позволяет выбрать 16 цветов из 64 (2^6) возможных.

Палитра EGA-адаптера, устанавливаемая по умолчанию, приведена в табл. 1.10.

Таблица 1.10.

Палитра EGA-адаптера.

Номер регистра палитры	*Содержимое регистра палитры (rgbRGB)*	*Цвет*
0	000000	Черный
1	000001	Синий
2	000010	Зеленый
3	000011	Голубой
4	000100	Красный
5	000101	Вишневый
6	000110	Коричневый
7	000111	Белый
8	111000	Серый
9	001001	Ярко-синий
10	010010	Ярко-зеленый
11	011011	Ярко-голубой
12	100100	Ярко-красный
13	101101	Ярко-вишневый
14	110110	Желтый
15	111111	Ярко-белый

Обобщенная схема определения цвета пиксела в EGA-видеоадаптере представлена на рисунке 1.9.

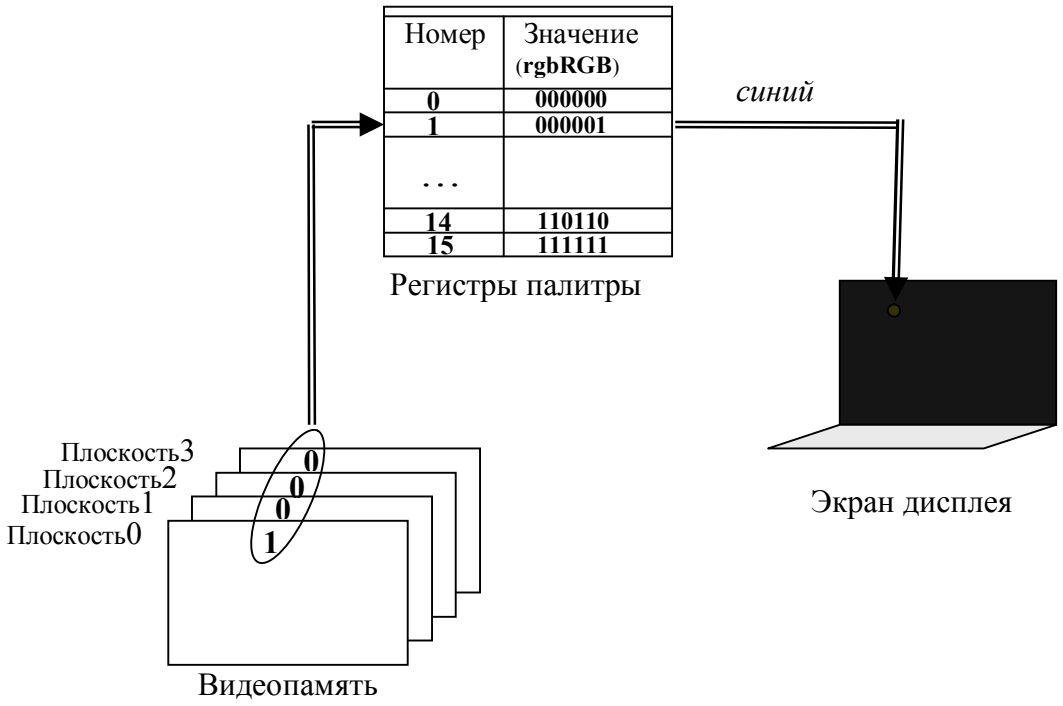


Рис. 1.9. Обобщенная схема определения цвета пиксела в EGA-адаптере.

В регистры палитры можно загружать любые значения из 64 возможных. Для изменения содержимого регистров палитры в библиотеку графики языка программирования СИ включены функции *setallpalette()* и *setpalette ()*.

Функция *setallpalette()* позволяет изменить содержимое всех 16 регистров палитры. Формат функции:

#include <graphics.h>

*void far setallpalette (struct palettetype far *palette);*

Новые значения регистров палитры являются элементами массива *colors[MAXCOLORS+1]*, входящего в состав структурной переменной *palettetype.* Шаблон структуры *palettetype* определен в <*graphics.h*> так:

```
struct palettetype
    {
      unsigned char size;
      signed char colors[MAXCOLORS + 1];
    }
```

Константа *MAXCOLORS* в настоящий момент определена равной 15. Элементы массива *colors* – это любые целые числа в диапазоне от 0 до 63. Внутренние регистры палитры заполняются из массива *colors* последовательно. Если элемент массива *colors* равен *-1*, значение соответствующего регистра палитры не переопределяется.

Функция *setpalette()* позволяет изменить содержимое одного из регистров палитры. Формат функции:

#include <graphics.h>

void far setpalette (int colornum, int color);

где *colornum* – номер регистра палитры,

color - *rgbRGB* код цвета.

Для шестнадцатицветных видеорежимов *colornum* лежит в диапазоне от 0 до 15. Значение *color* может быть любым числом от 0 до 63.

Адаптер VGA

Поддерживает три дополнительных графических режима (табл. 1.11). Два из них (двухцветный 640x480 и шестнадцатицветный 640x480) имеют линейную организацию видеопамяти в виде четырех параллельных битовых плоскостей с начальным адресом *A000:0000*, аналогичную видеоадаптеру EGA.

Таблица 1.11.

Графические режимы видеоадаптера VGA.

Номер режима	*Разрешение экрана*	*Бит на пиксел*	*Количество цветов*
11H	640x480	1	2
12H	640x480	1	16
13H	320x200	8	256

В 256-цветном режиме 320x200 также используется линейная организация видеопамяти без разделения на битовые плоскости. Каждому пикселу на экране соответствует один байт в видеобуфере. Таким образом, в этом графическом режиме на экране можно одновременно наблюдать $2^8 = 256$

цветов, выбираемых из регистров цифро-аналогового преобразователя (Digital to Analog Converter - DAC). Эти регистры имеют название DAC-регистры.

VGA–адаптер также использует "косвенную" схему определения цвета пиксела. В этом адаптере содержащийся в видеопамяти код определяет номер DAC-регистра. Цифро – аналоговый преобразователь имеет в своем составе 256 трехбайтовых внутренних регистров (по одному байту для красного, синего и зеленого цветов). Шесть бит каждого байта кодируют уровень интенсивности одной из цветовых компонент. Тем самым обеспечивается возможность воспроизведения на экране $2^6 = 64$ оттенка каждого цвета, а полное число цветов, которое выдает такой видеоадаптер, равно 64x64x64 = 262 144.

При работе VGA–адаптера в EGA-режимах используются только первые 64 DAC-регистра, задающие 64 возможных цвета палитры. В 256 – цветном графическом режиме адаптера VGA восьмибитовый код в видеопамяти задает номер одного из 256 DAC-регистров. Изменение значений в DAC-регистрах изменяет код цвета пиксела на экране. Для изменения содержимого DAC-регистров в библиотеку графики языка программирования СИ включена функция *setrgbpalette()*. Формат функции:

#include <graphics.h>

void far setrgbpalette (int colornum, int red, int green, int blue);

где *colornum* – номер DAC-регистра,

red, green, blue – уровень интенсивности, соответственно, красной, зеленой и синей компоненты цвета (значимыми являются шесть младших бит).

Например, для получения 16 оттенков серого цвета (от светло – серого до темно-серого) необходимо перезагрузить шестиразрядные *RGB*-регистры значениями:

$$\frac{63}{15} * i \qquad \text{или} \qquad 4 * i.$$

где – i номер регистра палитры $(i = 1 \quad 16)$.

Эти значения могут быть занесены в первые шестнадцать регистров палитры *(i = 1 16)* с помощью функции:

*Setrgbpalette(I, 63*i/15, 63*i/15, 63*i/15).*

Аналогично, 16 оттенков красного могут быть заданы, если первые 16 DAC-регистров перезагрузить значениями:

*G=0; B=0; R=63*i/15.*

Адаптеры SVGA

Видеоадаптеры SVGA поддерживают расширенный набор видеорежимов, стандартизированных ассоциацией стандартов в области видеоэлектроники VESA (Video Electronic Standarts Assotiation). Список основных VESA-режимов приведен в табл. 1.12.

SVGA –видеоадаптеры наряду с палитровыми (100H – 107H) поддерживают непалитровые (10DH – 11BH) режимы работы видеосистемы. В непалитровых режимах для каждого пиксела задается не номер регистра палитры (или DAC - регистра), а непосредственно RGB-код цвета. Непалитровыми режимами являются режимы HiColor (15 или 16 бит на пиксед) и TrueColor (24 бита на пиксел).

В режимах HiColor и TrueColor видеопамять имеет линейную организацию. Под каждый пиксел отводится целое количество байт видеопамяти (2 байта для HiColor и 3 байта для TrueColor).

Видеорежим HiColor имеет два подрежима:

- под каждую цветовую компоненту (красную, зеленую, синюю) отводится по 5 бит двухбайтового кода цвета в видеопамяти, последний бит не используется. Таким образом, код цвета пиксела в этом режиме состоит из 15 бит, что обеспечивает воспроизведение на экране 2^{15} = 32 тысячи цветов (видеорежимы 10DH, 110H, 113H, 116H, 119H).

- под красную и синюю компоненты двухбайтового кода цвета в видеопамяти отводится по 5 бит, под зеленую – 6 бит. Таким образом, код цвета

пиксела в этом режиме состоит из 16 бит, что обеспечивает воспроизведение на экране 2^{16} = 64 тысяч цветов (видеорежимы 10EH, 111H, 114H, 117H, 11AH).

В режиме TrueColor. под каждую цветовую компоненту (R, G, B) отводится по одному байту. Таким образом, код цвета пиксела состоит из 24 бит, что обеспечивает воспроизведение на экране 2^{24} = 16 миллионов цветов (видеорежимы 10FH, 112H, 115H, 118H, 11BH).

Таблица 1.12.

Графические режимы видеоадаптера SVGA (VESA-режимы).

Номер режима	*Разрешение экрана*	*Бит на пиксел*	*Количество цветов*
100H	640x480	8	256
101H	640x480	8	256
102H	800x600	4	16
103H	800x600	8	256
104H	1024x768	4	16
105H	1024x768	8	256
106H	1280x1024	4	16
107H	1280x1024	8	256
10DH	320x200	15	32 000
10EH	320x200	16	64 000
10FH	320x200	24	16 000 000
110H	640x480	15	32 000
111H	640x480	16	64 000
112H	640x480	24	16 000 000
113H	800x600	15	32 000
114H	800x600	16	64 000
115H	800x600	24	16 000 000
116H	1024x768	15	32 000
117H	1024x768	16	64 000
118H	1024x768	24	16 000 000
119H	1280x1024	15	32 000
11AH	1280x1024	16	64 000
11BH	1280x1024	24	16 000 000

1.2. Программное управление видеосистемой.

Программное управление видеосистемой в графическом режиме может выполняться на одном из трех уровней:

- уровне языка программирования (Си, Паскаль);
- уровне BIOS;
- физическом уровне.

Графические библиотеки языков программирования содержат в своем составе программные функции вывода на экран графических примитивов (точек, линий) и графических объектов (окружности, эллипса, параллелепипеда и др.), а также программные функции установки атрибутов вывода графических изображений (определение стиля рисования, заполнение замкнутой области, закраска графических объектов). Поэтому использование встроенных графических функций намного сокращает объем программирования при разработке графических программ. Библиотечные функции языка «маскируют» многие технические детали управления видеооборудованием, о которых пользователь должен быть осведомлен при работе с видеоадаптером через функции BIOS и, особенно, через порты ввода/вывода видеоадаптера. Платой за эти удобства является значительное увеличение размера EXE-файла.

Базовая система ввода/вывода (BIOS) содержит в своем составе несколько десятков функций (программных процедур) управления дисплеем, в том числе и в графическом режиме. Эти функции, называемые видеоBIOS и доступные через прерывание 10H, реализуют все возможные действия управления видеосистемой, начиная от установки/чтения режима работы дисплея и заканчивая записью/считыванием кода цвета пиксела, расположенного в определенной позиции экрана дисплея. Функции вывода основных графических примитивов и объектов (линии, окружности) в состав BIOS не входят. Управление видеосистемой через вызовы процедур BIOS упрощает транспортабельность программ между видеооборудованием различных моделей и облегчает процесс программирования за счет освобождения

программиста от детального знания назначения многочисленных программно – доступных регистров видеоадаптера и особенностей организации видеобуфера в различных режимах работы видеосистемы. Однако, наряду с универсальностью, процедуры BIOS имеют существенный недостаток – сравнительно невысокое быстродействие. Каждая из процедур производит много служебных операций, связанных с сохранением и восстановлением значений регистров процессора, определением конкретной аппаратной конфигурации, текущего видеорежима и т.п. Конечно, эти операции не существенны для редко выполняемых графических операций (например, смена режима дисплея или загрузка символьного набора). Однако, при выводе сложных графических изображений или формировании движущихся объектов применение BIOS может стать ограничивающим фактором.

В персональных компьютерах фирмы IBM предусмотрено также прямое управление видеосистемой. Оно обеспечивает наивысшую производительность формирования графического изображения, но при этом теряется возможность транспортабельности программ (работы программ на другом типе видеоадаптера или в другом режиме). Кроме того, от программиста требуется детальное знание многочисленных программно-доступных регистров и особенностей организации видеобуфера в различных режимах работы видеосистемы.

Базовыми операциями, выполняемыми в графическом режиме работы видеосистемы являются операции, связанные с выводом пиксела специфицированного цвета в заданную позицию экрана и определением кода цвета пиксела, расположенного в заданной позиции экрана.

Для того, чтобы выполнять базовые графические операции необходимо установить, в каком режиме работает видеосистема. Видеорежим определяет логику работы аппаратуры видеосистемы и принципы ее программирования. Поведение видеоадаптера в том или ином режиме является фактическим стандартом и полностью характеризует доступные для программиста возможности и средства управления видеосистемой:

а) разрешающую способность экрана;

б) количество цветов, которые можно одновременно отобразить на экране и механизм формирования этих цветов;

в) организацию видеопамяти и способы доступа к ней;

г) назначение и порты доступа к программно-доступным регистрам видеоадаптера.

Режимы нумеруются от нуля (графическая библиотека языка Си содержит набор зарезервированных констант, идентифицирующих номер режима работы дисплея). Список основных графических режимов для видеоадаптеров разных типов приведен в табл. 1.7, 1.9, 1.11, 1.12.

Установка видеорежима может быть выполнена на всех трех уровнях программного управления видеосистемой.

В языке Си установку графического режима выполняет функция *initgraph().* Функция *initgraph()* инициализирует графическую систему, загружает для заданного видеоадаптера BGI-драйвер (BGI - Borland Graphics Interface) и устанавливает видеоадаптер в определенный графический режим.

Формат этой функции:

 # include <graphics.h>

*void far initgraph (int far*graphdriver, int far*graphmode, char far*pathdriver);*

Здесь:

graphdriver – указатель на ASCII-строку, хранящую спецификацию файла BGI-драйвера. Графическая библиотека языка Си поддерживает фиксированное число драйверов, определяемых типом видеоадаптера.

graphmode – указатель на графический режим;

pathdriver - маршрут поиска файла, содержащего BGI-драйвер.

В случае, когда при вызове функции *initgraph()* первый параметр *(graphdriver)* указывает на встроенную символическую константу *DETECT*, функция *initgraph()* вызывает библиотечную функцию *detectgraph (),* определяющую тип активного видеоадаптера системы. Значение подходящего для этого адаптера BGI-драйвера и видеорежим, обеспечивающий

максимальное разрешение, возвращаются в ячейках памяти, на которые указывают параметры *graphdriver* и *graphmode* функции *initgraph()*.

В состав videoBIOS входят функции установки видеорежима и определения текущего видеорежима.

Формат функции BIOS установки видеорежима:

На входе: *AH = 00h*

AL = номер видеорежима

Возвращаемое значение: *нет.*

Формат функции BIOS определения видеорежима:

На входе: *AH = 0Fh*

Возвращаемое значение: *AH – число символьных столбцов на экране*

AL – номер видеорежима

BH – активная видеостраница

Операции установки/определения видеорежима могут быть выполнены и на физическом уровне. Реализация этих операций связана с записью/считыванием содержимого внутренних регистров видеоадаптера. При выполнении операций установки/определения видеорежима должно быть записано/считано содержимое битовых полей атрибутных регистров видеоадаптера (регистра режима, регистра цвета, статусного регистра), а также регистров 00H – 09H контроллера электронно – лучевой трубки. Эти операции достаточно кропотливые, требуют большой точности, поскольку любая случайная ошибка при их выполнении может привести даже к выводу из строя монитора. Поэтому операции установки/определения видеорежима рекомендуется выполнять с использованием встроенных программных процедур BIOS или языка программирования.

Инициализацию содержимого регистров видеоадаптера, связанных с установкой режима его работы, выполняет процедура начальной загрузки BIOS при включении компьютера. Эта же процедура инициализирует значения параметров видеосистемы в области данных BIOS (например, ячейка

0040:0065H повторяет содержимое регистра видеоадаптера 3x8H, а ячейка 0040:0049H содержит номер текущего видеорежима).

После того, как режим, в котором работает видеосистема, определен, можно выполнять основные графические операции.

Как уже было сказано ранее, базовыми графическими операциями являются операции чтения/записи кода цвета пиксела. Это связано с тем, что изображение любого графического объекта формируется в результате подсвечивания нужным цветом определенной совокупности точек (пикселов) на экране.

Операции чтения/записи кода цвета пиксела в видеопамять могут быть выполнены на всех трех уровнях программного управления видеосистемой.

Графическая библиотека языка Си содержит в своем составе две функции, связанные с манипуляцией отдельными пикселами на экране:

getpixel() – функция считывания из видеопамяти кода цвета пиксела, расположенного в позиции с координатами *(x, y)* на экране;

putpixel() – функция записи в видеопамять кода цвета пиксела расположенного в позиции с координатами *(x, y)* на экране.

Формат функции считывания кода цвета пиксела из видеопамяти:

 # include <graphics.h>

unsigned far getpixel(int x, int y);

Функция возвращает хранящийся в видеопамяти код цвета пиксела, расположенного в позиции с координатами *(x, y)* на экране.

Формат функции записи кода цвета пиксела в видеопамять:

 # include <graphics.h>

void far putpixel(int x, int y, int color);

Функция записывает в видеопамять код цвета *color* для пиксела, расположенного в позиции с координатами *(x, y)* на экране.

Базовая система ввода/вывода для реализации двух основных графических операций содержит в своем составе функции 0CH (запись пиксела) и 0DH (считывание пиксела) прерывания 10H.

Формат функции BIOS записи кода цвета пиксела в видеопамять:

На входе: *AH = 0CH*

AL - код цвета пиксела

BH - страница видеопамяти

CX - координата x (столбец)

DX - координата y (строка)

Возвращаемое значение: *нет.*

Формат функции BIOS считывания кода цвета пиксела из видеопамяти

На входе: *AH = 0Dh*

BH - страница видеопамяти

CX - координата x

DX - координата y

Возвращаемое значение: *AL - код цвета пиксела*

Вывод пиксела через BIOS даже для высокопроизводительных компьютеров выполняется недостаточно быстро. Функция записи кода цвета пиксела является основной при формировании на экране графических изображений, поэтому к ее производительности предъявляются особые требования. Максимальная скорость выполнения базовых графических операций достигается при непосредственном доступе к аппаратуре видеоадаптера.

Реализация операции записи кода цвета пиксела в видеопамять на физическом уровне включают в себя пять последовательных шагов:

Шаг 1. Вычисление линейного адреса байта видеопамяти, содержащего код цвета пиксела, по заданным координатам (*x, y*) на экране;

Шаг 2. Считывание адресуемого байта из видеопамяти;

Шаг 3. Выделение в считанном байте значения нужного пиксела;

Шаг 4. Модификация байта новым значением кода цвета пиксела;

Шаг 5. Запись модифицированного байта в видеопамять.

Операция считывания кода цвета пиксела из видеопамяти на физическом уровне предполагает выполнение первых трех шагов приведенной выше последовательности.

Рассмотрим реализацию этих шагов для различных типов видеоадаптеров на физическом уровне более детально.

Шаг 1. Вычисление линейного адреса байта видеопамяти, содержащего код цвета пиксела, по заданным координатам (x, y) на экране

Позиция пиксела на экране определяется прямоугольными декартовыми координатами *(x,y),* где *x* – номер столбца, *y* – номер строки экрана. Начало экранной системы координат *(0,0)* располагается в верхнем левом углу экрана. В графическом режиме цветовое значение каждого пиксела хранится в видеопамяти как один или несколько бит. При этом, для каждого типа видеоадаптера и каждого режима его работы имеется взаимно однозначное соответствие между позициями пикселов на экране и расположением их цветовых кодов в видеопамяти (рисунок 1.10).

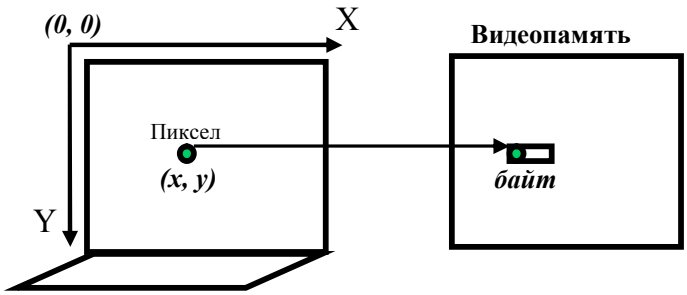


Рис. 1.10. Положение цветового кода пиксела в видеопамяти.

Таким образом, любому пикселу с координатами *(x,y)* на экране однозначно соответствуют строго определенные биты строго определенного байта в видеопамяти. Следовательно, по заданным экранным координатам можно вычислить, где в видеобуфере находится цветовое значение конкретного пиксела.

Адаптер CGA.

Начальный адрес видеопамяти в адаптере *CGA* равен *b800:0000.* Сегментная часть адреса у всех ячеек видеопамяти адаптера *CGA* одинаковая и равняется *b800*. Требуется определить величину смещения полного адреса байта видеопамяти, содержащего код цвета пиксела, расположенного в позиции *(x,y*) на экране.

Прежде всего необходимо выяснить, в какой половине видеобуфера располагается код цвета пиксела, заданного координатами *(x, y*). Если младший бит координаты "*y"* равен *«0»* (*(y & 1) = 0*), то пиксел располагается в четной строке экрана, а информация о нем – в первой половине видеопамяти, начинающейся со смещения 0H. Если младший бит координаты "*y"* равен *«1»* (*(y & 1) ≠ 0*), то пиксел располагается в нечетной строке, а информация о нем – во второй половине видеопамяти, начинающейся со смещения 2000H (80 байт x 100 четных строк = 8000_{10} = 2000h).

В графических режимах адаптера *CGA* на одной строке экрана располагается 640 пикселов для двухцветного режима и 320 пикселов для четырехцветного режима. В двухцветном режиме адаптера *CGA* один байт видеопамяти кодирует цвета 8 пикселов, в четырехцветном режиме - 4 пикселов. Таким образом, для хранения информации о цветах пикселов, расположенных в одной строке экрана, требуется 80 байт (640/8 – для двухцветного, 320/4 – для четырехцветного режима). Следовательно, при вычислении начального адреса (смещение внутри выбранной половины видеобуфера) участка видеопамяти, в котором располагаются коды всех пикселов строки с номером *"y",* значение координаты *"y"* (без младшего бита) необходимо умножить на 80:

*80 * (y >>1).*

Внутри этого участка видеопамяти каждым 8 (для двухцветного режима) или 4 (для четырёхцветного режима) последовательным пикселам, начинающимся с позиции, кратной 8 (или 4), соответствует один байт видеопамяти. Таким образом, для определения величины смещения нужного

байта, внутри участка видеопамяти, соответствующего строке, в которой располагается пиксел, значение его координаты *"x"* требуется разделить нацело:

- на восемь для двухцветного режима:

x >> 3:

- на четыре для четырёхцветного режима:

x >> 2.

Окончательно, адрес байта видеопамяти, в котором содержится цветовое значение пиксела, заданного координатами *(x, y)*, для *CGA* – адаптера рассчитывается по формулам:

Для двухцветного режима

byte = b800 : [*80 * (y>>1) + (x>>3)*], если строка четная ((y&1) = 0);

byte = b800 : [*2000H + 80 * (y>>1) + (x>>3)*], если строка нечетная ((y&1) ≠ 0).

Для четырёхцветного режима

byte= b800 : [*80*(y>>1) + (x>>2)*], если строка четная *((y&1) = 0);*

byte= b800 : [*2000H + 80*(y>>1) + (x>>2)*], если строка нечетная *((y&1) ≠ 0).*

Адаптеры EGA,VGA.

В естественных режимах этих видеоадаптеров, за исключением 256-цветного режима адаптера *VGA*, организация видеопамяти характеризуется следующими особенностями:

а) начальный адрес видеопамяти равен A000:0000;

б) весь видеобуфер имеет линейную организацию, т.е. не разделяется на 2 половины;

в) байт с конкретным адресом содержит значения 8 пикселов, а строка развёртки соответствует 80 байтам видеопамяти ($\frac{640\_пикселов\_в\_строке}{8}$).

Таким образом, для 16-цветных режимов *EGA (VGA)* – адаптера адрес байта видеопамяти, в котором содержится цветовое значение пиксела с координатами *(x,y)*, рассчитывается по формуле:

byte = A000 : [*80 * y + (x>>3)*].

В отличие адаптера CGA, где применяется черезстрочная организация видеобуфера, в естественных видеорежимах адаптера EGA строки пикселов отображаются в видеопамяти линейно аналогично линейному отображению текстовых строк в текстовом режиме.

В 256 – цветном режиме 320 x 200 в связи с тем, что один байт видеопамяти кодирует один пиксел, каждой строке экрана соответствует 320 байт видеопамяти. И, следовательно, для 256-цветного режима *VGA* – адаптера адрес байта видеопамяти, в котором содержится код цвета пиксела с координатами *(x,y*), рассчитывается по формуле:

$byte = A000 : [320 * y + x]$.

После того, как адрес байта видеопамяти определен, выполняется операция считывания содержимого ячейки видеопамяти с адресом *byte* (Шаг 2). Реализация этой операции для различных типов видеоадаптеров будет рассмотрена позднее.

Шаг 3. Выделение в считанном байте значения нужного пиксела.

При выполнении этой операции требуется знать номер бита (для двухцветного и шестнадцатицветного режимов) или пары битов (для четырехцветного режима адаптера *CGA*) внутри считанного из видеопамяти байта, определяющих код цвета пиксела с координатами *(x, y*). Поскольку в двухцветном и шестнадцатицветном режимах каждый байт видеопамяти хранит информацию о восьми пикселах, номер бита, определяющего код цвета заданного пиксела, является остатком от деления координаты «*x*» пиксела на 8:

$$Nb = x \ \& \ 7.$$

В четырехцветном режиме адаптера *CGA* номер пары бит, определяющей код цвета заданного пиксела, рассчитывается, как остаток от деления координаты «*x*» пиксела на 4 (4 - количество закодированных в одном байте видеопамяти пикселов):

$$Npb = x \ \& \ 3.$$

Прим. Нумерация пикселов в байте видеопамяти производится слева направо:

Двухцветный и шестнадцатицветный режимы.

0 1 2 3 4 5 6 7 - нумерация пикселов в байте видеопамяти.

X	***X***	***X***	***X***	***X***	***X***	***X***	***X***

- байт видеопамяти

Четырехцветный режим адаптера *CGA.*

0 1 2 3 - нумерация пикселов в байте видеопамяти.

X	***X***	***X***	***X***	***X***	***X***	***X***	***X***

- байт видеопамяти

Для выделения соответствующих заданному пикселу бит необходимо значение считанного байта видеопамяти (**byte)* логически умножить на число *CH*, у которого установлены в единицу биты, соответствующие заданному пикселу:

**byte& CH.*

Для двухцветных и шестнадцатицветных режимов:

CH = 0x80 >> Nb = 0x80>>(x&7)

Для четырехцветного режима адаптера *CGA*:

CH = 0xC0 >> (Npb<<1) = 0xC0 >> ((x&3) << 1)

Шаги 2,4,5. Считывание байта видеопамяти, модификация байта новым значением кода цвета пиксела, запись модифицированного байта в видеопамять.

Во всех графических режимах адаптера *CGA*, 256-цветном режиме адаптера *VGA* и во всех непалитровых режимах видеоадаптера *SVGA* можно обратиться к пикселам, осуществляя прямое считывание или запись байта в видеобуфер по указанному адресу.

Однако, в естественных графических режимах адаптеров *EGA* и *VGA* прямой доступ к видеобуферу, состоящему из четырех битовых плоскостей, не возможен. Двоичные плоскости в графических режимах адаптеров EGA и VGA адресуются параллельно, т.е. при производстве процессором считывания или записи по конкретному адресу видеопамяти этот адрес относится не к одному, а к четырём одинаково размещённым байтам (по одному на каждой битовой

плоскости). Доступом к значениям пикселов, хранящимся в двоичных плоскостях, управляют специальные регистры, входящие в состав графического контроллера и секвенсера.

Для обеспечения взаимодействия с видеопамятью, состоящей из четырех параллельных битовых плоскостей, в состав видеоадаптеров *EGA* и *VGA* входят четыре (по одному на каждую битовую плоскость) однобайтовые регистра - защелки (latch-registers), как это показано на рисунке 1.11. При выполнении операции чтения из видеопамяти четыре байта, расположенные по одному адресу в разных битовых плоскостях видеопамяти, переносятся в регистры-защёлки (РЗ). Затем эти данные могут быть переданы в процессор. При выполнении операции записи в видеопамять информация из регистров-защелки, скомбинированная с данными, переданными от процессора, переносится по указанному адресу в каждую из четырех битовых плоскостей видеопамяти. Перед записью информации в видеопамять содержимое регистров-защелки должно быть обновлено. Это обычно достигается с помощью операции чтения из видеопамяти.

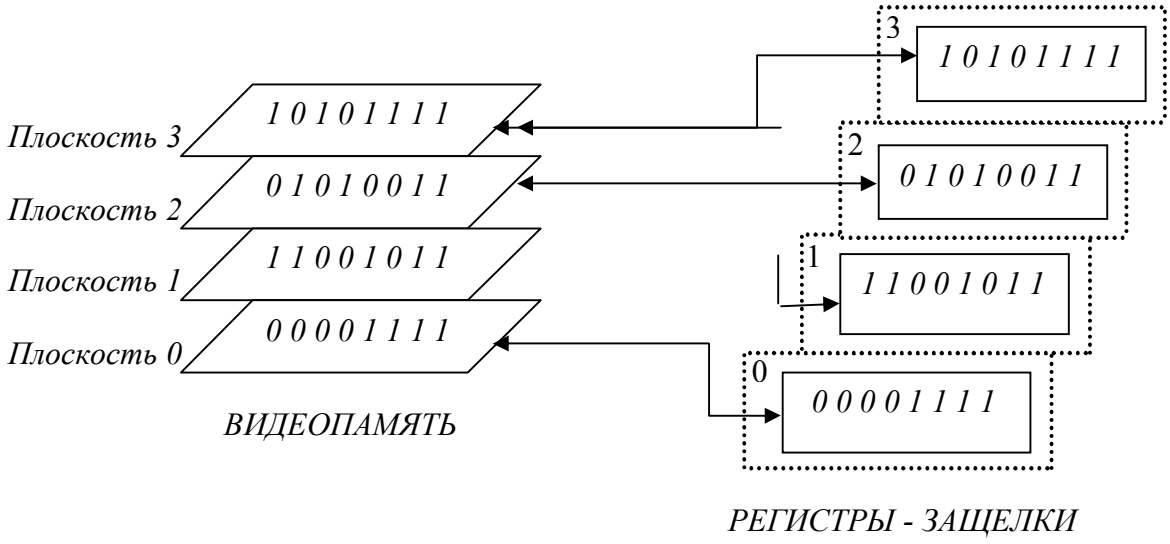


Рис.1.11. Регистры – защелки.

Обменом данными между процессором, регистрами защелки и видеопамятью управляет графический контроллер (ГК). Графический контроллер содержит 9 регистров.

Доступ к регистрам графического контроллера осуществляется через порты:

3CEH – адрес порта доступа к индексному регистру графического контроллера. В индексный регистр заносится номер регистра данных графического контроллера;

3CFH – адрес порта доступа к регистру данных, номер которого был указан в индексном регистре.

Содержимое регистров данных графического контроллера определяет, как будут обрабатываться данные при выполнении операций считывания из видеопамяти и записи в видеопамять. Эти регистры обеспечивают гибкие средства маскирования, т.е. селективного выбора участвующих в операциях чтения/записи пикселов и плоскостей.

Назначение регистров данных графического контроллера и их содержимое, устанавливаемое процедурой начальной загрузки ROM-BIOS по умолчанию приведено в табл. 1.6.

В случае изменения содержимого регистров данных рекомендуется после завершения выполняемой операции приводить их в начальное состояние.

В адаптере EGA встроены два режима чтения (0 и 1) и три режима записи (0, 1, 2), а в адаптере VGA добавлен ещё один режим записи (3). Каждый из этих режимов предоставляет значительные удобства при программировании конкретных графических задач. Номер режима чтения или записи задается в регистре режима 05h графического контроллера: бит 3 определяет номер режима чтения, а биты 1 - 0 задают номер режима записи. Остальные биты этого регистра обычно содержат нули. Формат регистра режима графического контроллера:

№ бита	***7***	***6***	***5***	***4***	***3***	***2***	***1***	***0***
Состояние бита	0	0	0	0	Номер режима считывания	0	Номер режима записи	

Режимы чтения.

Режим чтения 0.

В этом режиме при инициировании процессором операции чтения из видеопамяти графический контроллер передаёт в процессор содержимое одного из четырех регистров-защелки (рисунок 1.12).

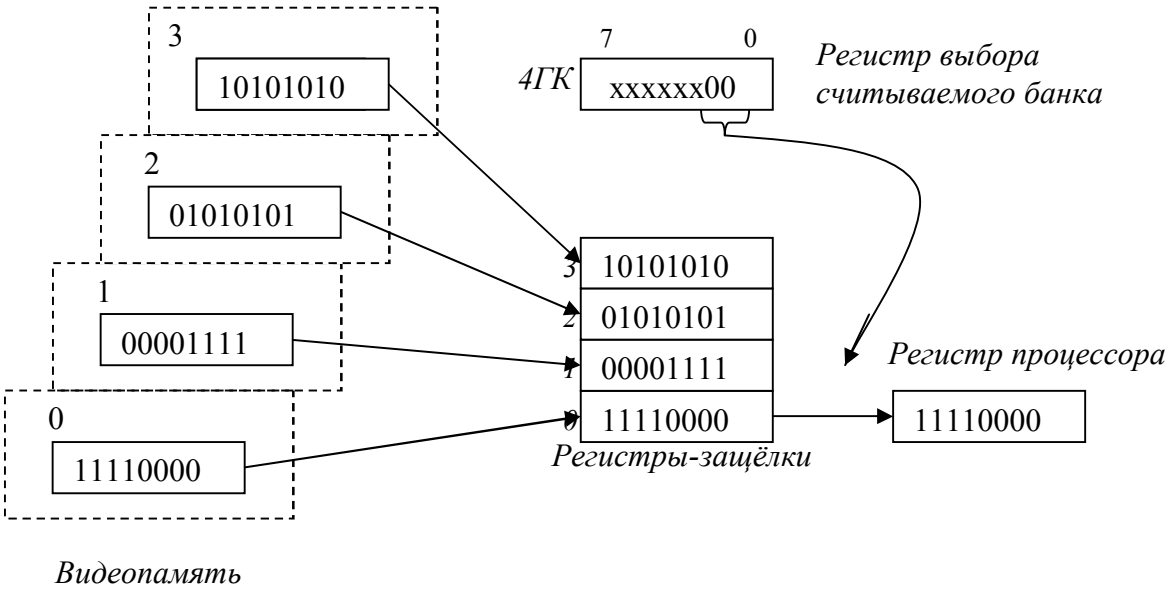


Рис.1.12. Режим чтения 0

Режим чтения 0 позволяет передать в процессор байт из определенной двоичной плоскости видеопамяти. Номер двоичной плоскости, содержимое которой будет передано в процессор, задается в битах 0 и 1 регистра 4 (регистра выбора считываемого банка) графического контроллера

Этот режим чтения может быть использован для реализации функции *getpixel()* чтения кода цвета пиксела на физическом уровне. Для этого потребуется выполнить операции чтения байта из каждой двоичной плоскости, поскольку четырехбитовый код цвета пиксела распределён между этими плоскостями.

Например, программная реализация на физическом уровне функции *getpixel()* считывания из видеопамяти кода цвета пиксела, расположенного на экране в позиции *(x, y)*, может выглядеть так:

```
int getpixel (int x, int y)
 {
    int i, color=0;
   short int a;
    short int far * byte = (short int far*) 0xA0000000;
    outpb (0x3CE,0x05);
    outpb (0x3CF, 00);              /*задание в регистре 5ГК режима чтения 0*/
    FP_OFF (byte) = 80*y + (x>>3);    /* Шаг1. Вычисление адреса байта*/
    for (i=3; i>=0; i- -)                  /*i – номер двоичной плоскости */
      {
        color << =1;                  /*color – считываемы код цвета пиксела*/
        outpb (0x3CE, 0x4);
        outpb (0x3CF, i);      /*задание  в регистре 4ГК номера двоичной плоскости*/
        a = (*byte & (0x80 >> (x 8 7)));          /*Шаги 2,3 */
        if (a != 0)
             color |= 1;
      }
    return color;
}
```

Режим чтения 1.

В этом режиме коды цвета восьми пикселов, переданных из видеобуфера в регистры-защелки в результате выполнения процессором считывания из ячейки видеопамяти, сравниваются с содержимом младших четырех бит регистра 02h сравнения цвета графического контроллера (2ГК). Результат сравнения передается в процессор в виде байта считанных данных (рисунок 1.13). Когда значение пиксела равно содержимому регистра сравнения цвета, соответствующий бит в возвращаемом байте устанавливается в *«1»,* а при неравенстве – в *«0»*.

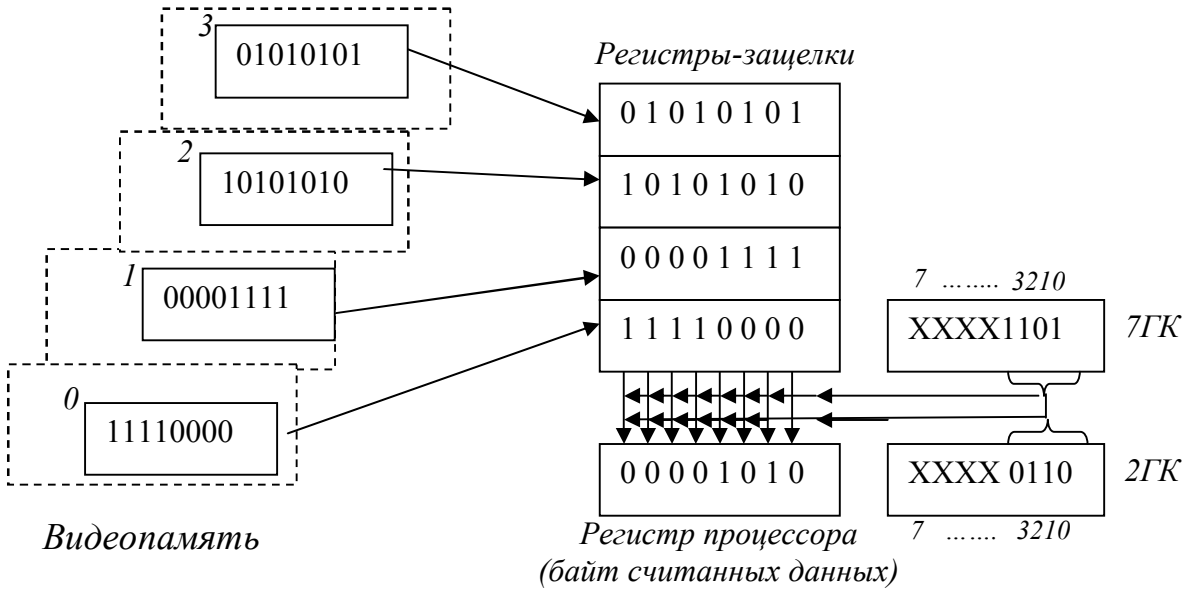


Рис.1.13. Режим чтения 1

В процессе сравнения участвует регистр 07h безразличного цвета графического контроллера (7ГК). Четыре младших бита этого регистра определяют, значения каких двоичных плоскостей участвуют в операции сравнения: если бит *k* регистра 7ГК = 0, то *k*-ая плоскость участвует в операции сравнения.

Режим чтения 0 позволяет быстро находить пикселы, имеющие конкретный цвет. В частности, он может быть использован для быстрого нахождения всех нефоновых пикселов независимо от их конкретного цвета. Для этого в регистр сравнение цвета (2ГК) загружается код цвета фона. В возвращаемом в процессор байте считанных данных соответствующие нефоновым пикселам биты содержат нули.

Этот режим может быть использован для реализации функции *getpixel()* чтения кода цвета пиксела на физическом уровне. При этом, может потребоваться до 16 (по числу цветов палитры) считываний из видеопамяти, прежде чем для пиксела будет возвращена 1, показывающая, что пиксел имеет конкретное значение..

Режимы записи.

В реализации всех режимов записи участвуют следующие регистры графического контроллера:

Регистр 00h установки /сброса (0ГК);

Регистр 01h разрешения установки/сброса (1ГК);

Регистр 03h сдвига данных/ выбора функции (3ГК);

Регистр 08h двоичной маски (8ГК);

Регистр двоичной маски (08h) определяет, значение каких пикселов подлежит модификации. Соответствующие модифицируемым пикселам разряды регистра двоичной маски установлены в единицу. Если же конкретный бит в этом регистре равен нулю, цветовое значение соответствующего пиксела просто копируется в видеобуфер из регистров-защелки.

Регистр сдвига данных/выбора функции (03h) имеет два поля:

1) биты 4-3 задают тип операции, которая будет реализована между содержимым регистров защелки и новым цветовым значением для пикселов, подлежащих модификации. Эти биты могут принимать следующие значения:

00_2 – простая замена новыми значениями цвета;

01_2 – логическая операция «И» (AND);

10_2 – логическая операция «ИЛИ» (OR);

11_2 – логическая операция «исключающее ИЛИ» (XOR)

2) биты 2-0 – определяют, на сколько разрядов должен быть циклически сдвинут вправо байт данных от процессора до его использования. Такая возможность применяется редко при графическом программировании. Обычно проще осуществить сдвиги данных в процессоре до записи их в видеобуфер, чем программировать на сдвиги графический контроллер. Поэтому поле сдвига (биты 2 - 0) в регистре 3ГК обычно содержит *000*.

Режим записи 0.

Это наиболее сложный изо всех режимов чтения и записи. В этом режиме может выбираться один из двух способов записи информации в видеопамять: попиксельная запись (подрежим А). или побайтовая запись (подрежим Б). Таким своеобразным двухмерным воздействием управляет регистр 01h разрешения установки/сброса графического контроллера (1ГК). Если разряды с 0 по 3 этого регистра равны *1111*, то будет реализован подрежим А, т.е. модификация отдельных пикселов всех плоскостей. Если четыре младшие бита регистра 1ГК равны *0000*, то будет реализован подрежим Б, т.е. модификация двоичных плоскостей целыми байтами.

В подрежиме А (рисунок 1.14) для всех разрешенных маской пикселов в видеобуфер заносятся значения пикселов, получаемые путем объединения (замены) цветовых кодов соответствующих пикселов из регистров-защелки с содержимым младших четырех бит регистра 00h установки/сброса графического контроллера (0ГК).

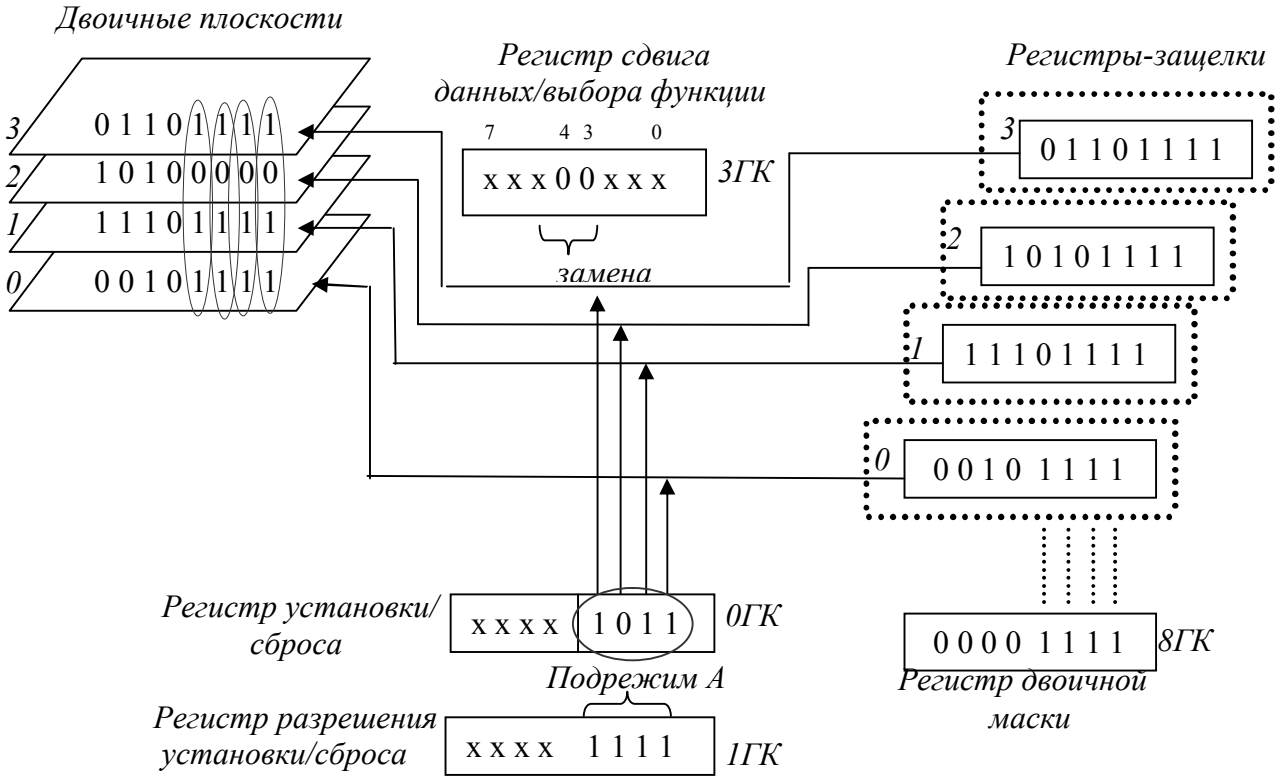


Рис. 1.14. Подрежим А режима записи 0 (попиксельная запись в видеопамять).

В подрежиме Б (рисунок 1.15) в каждую двоичную плоскость видеопамяти заносится байт данных, получаемый путем объединения (замены) сдвинутого байта данных от процессора с содержимым соответствующего регистра-защелки. Операция выполняется только для разрешенных регистром маски (8ГК) пикселов.

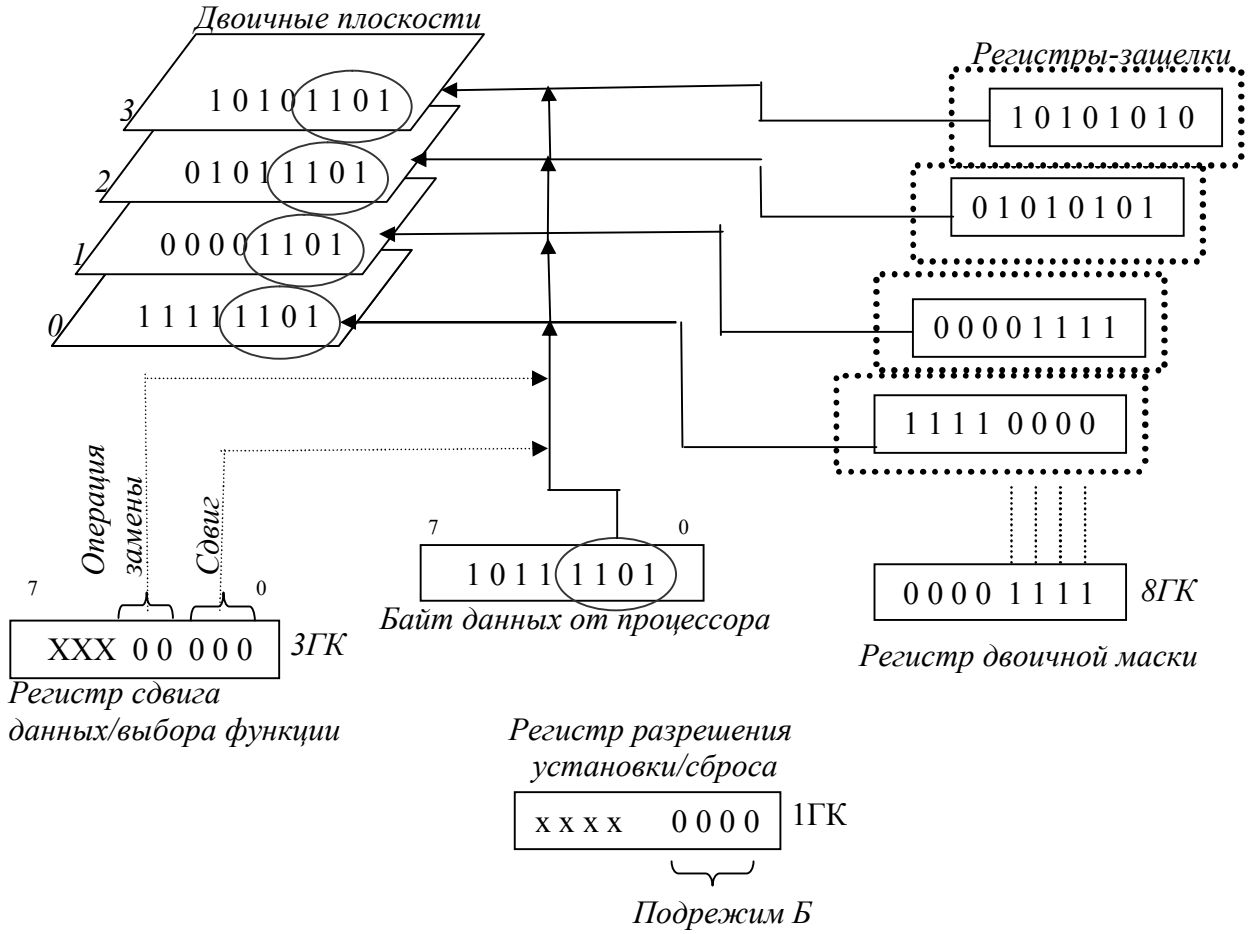


Рис 1.15. Подрежим Б режима записи 0 (побайтовая запись в видеопамять).

Режим записи 1.

В этом наиболее простом режиме при выполнении процессором операции записи текущее содержимое регистров-защелки помещается в двоичные плоскости видеопамяти (рис.1.16).

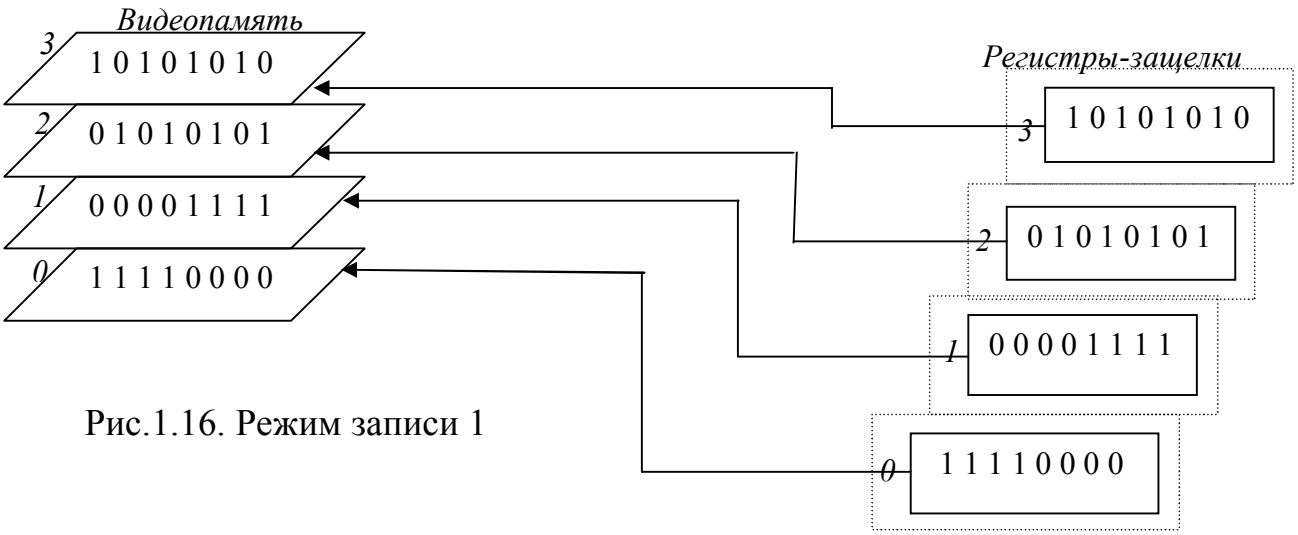


Рис.1.16. Режим записи 1

В этом режиме не привлекаются никакие регистры графического контроллера. Обычно, данный режим применяется с целью перемещения изображения из одной области экрана в другую. В этом случае за одну операцию чтения-записи перемещаются значения сразу восьми пикселов.

Режим записи 2.

В этом режиме (рисунок 1.17) младшие четыре бита байта данных от процессора играют такую же роль, как содержимое регистра установки/сброса (0ГК) в подрежиме А режима записи 0.

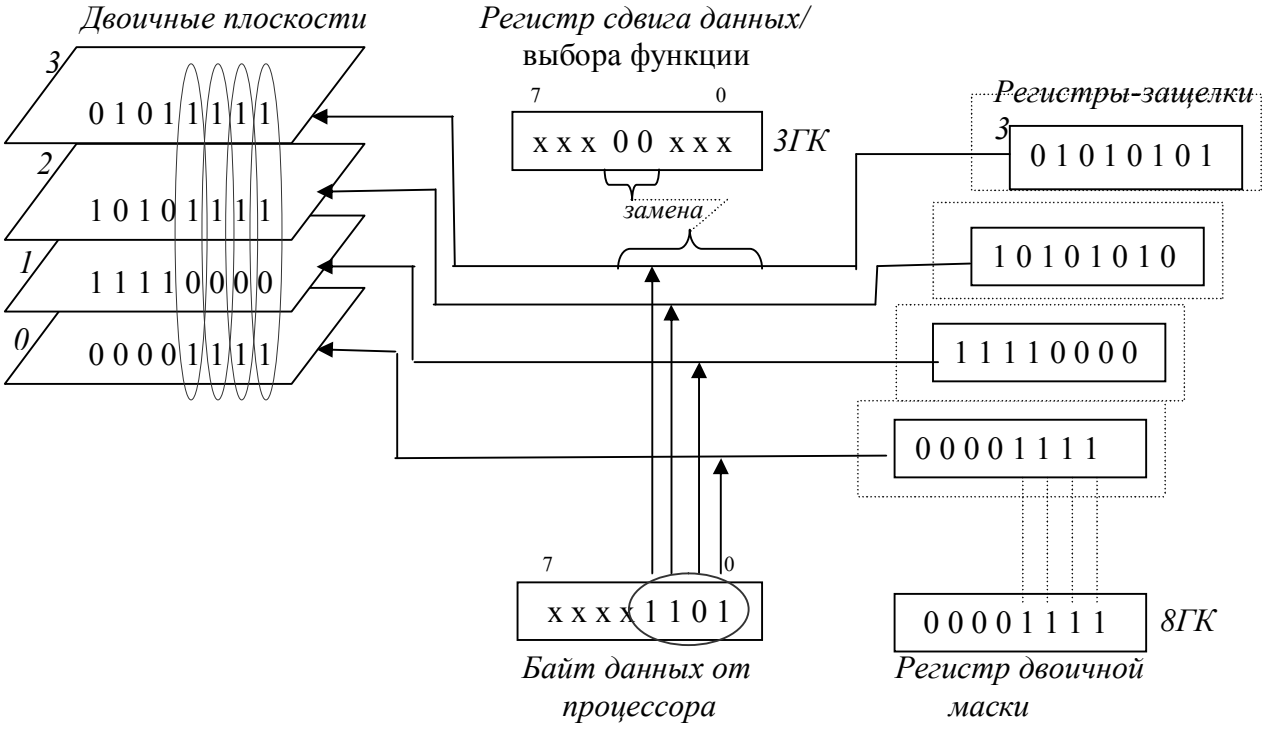


Рис. 1.17. Режим записи 2

Другими словами, для всех разрешенных маской пикселов в видеобуфер заносятся четырехбитовые коды цвета, получаемые путем объединения (замены) цветовых значений соответствующих пикселов из регистров-защелки с содержимым младших четырех бит байта данных от процессора. Тип используемой логической операции определяют биты 3 и 4 регистра 03h сдвига данных/выбора функции графического контроллера (3ГК).

Этот режим наиболее удобен для записи в видеобуфер кодов цвета отдельных пикселов. Например, программная реализация на физическом уровне функции *putpixel()* записи в видеопамять кода цвета *color* для пиксела, расположенного на экране в позиции (*x, y*), может выглядеть так:

```
void putpixel (int x, int y, int color)
    {
     int mask;
     short int byte;
     short int far *ptr = (short int far*) 0xA0000000;
   FP_OFF (ptr) = 80 * y + (x>>3);   /* Шаг1. Вычисление адреса байта*/
   byte = *ptr;     /* Шаг2. Считывание адресуемого байта из видеопамяти */
     outportb (0x3CE, 0x5);          /*задание в регистре 5ГК режима записи 2*/
     outportb (0x3CF, 0x2);
     mask = 0x80 >> (x&7);       /*Шаг3: выделение в байте нужного пиксела*/
     outportb (0x3CE, 0x8);         /*установка регистра маски ГК*/
     outportb(0x3CF, mask);
     outportb (0x3CE, 3);   /*задание операции замены в регистре 3ГК*/
     outportb (0x3CF, 0);
     *ptr = color;              /*запись кода цвета color в видеопамять*/
    }
```

Режим записи 3.

В данном режиме (рисунок 1.18), имеющемся только в адаптере VGA, пикселы модифицируются с помощью объединения значений пикселов из

регистров защелки с содержимым четырех младших бит регистра 00h установки/сброса графического контроллера (0ГК).

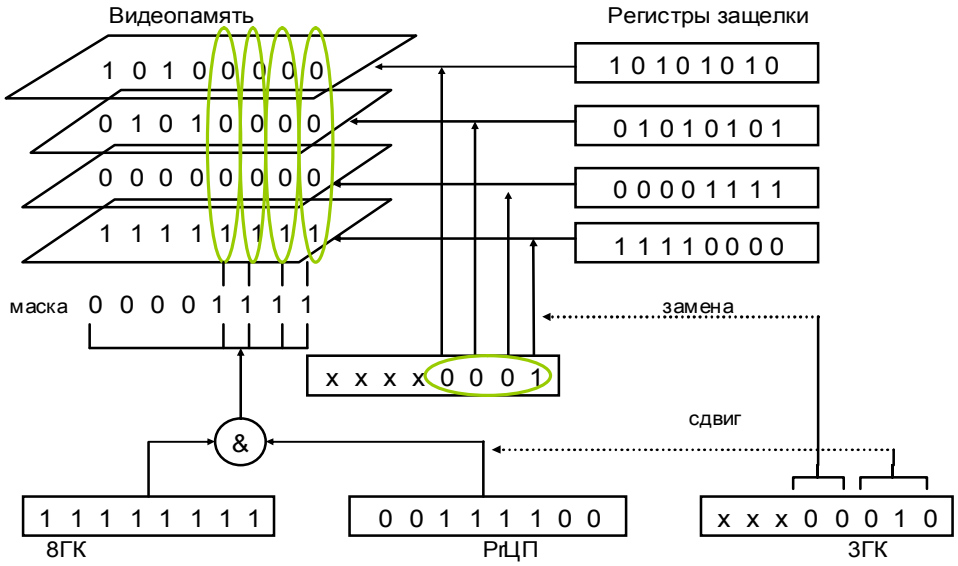


Рис.1.18. Режим записи 3.

По-прежнему, регистр 03h сдвига данных/ выбора функции графического контроллера (3ГК) определяет тип логической операции, которая используется при записи значений пикселов в видеопамять.

А вот двоичная маска, играющая такую же роль как и в остальных режимах записи, образуется путем поразрядного логического умножения (&) байта данных от процессора (циклически сдвинутого вправо на число бит, определяемое значением разрядов 0 - 2 регистра сдвига данных/ выбора функции графического контроллера) на содержимое регистра двоичной маски.

Контрольные задачи к главе 1.

1. Рассчитать адрес байта видеопамяти, содержащего код цвета пиксела с координатами:

а). (*65, 20*) в 16-цветном режиме видеоадаптера EGA с разрешением 640 x 350;

б). (100, 120) в 2-цветном режиме видеоадаптера CGA с разрешением 640 x 200;

в). (79, 125) в 4-цветном режиме видеоадаптера CGA с разрешением 320 x 200;

г). (57, 123) в 256-цветном режиме видеоадаптера VGA с разрешением 320 x 200;

д). (100, 200) в режиме HiColor (32 тысячи цветов) видеоадаптера SVGA с разрешением 640 x 480:

е). (33, 330) в режиме HiColor (64 тысячи цветов) видеоадаптера SVGA с разрешением 640 x 480:

ж). (50, 125) в режиме TrueColor (16 милионов цветов) видеоадаптера SVGA с разрешением 640 x 480.

2. Определить номер бита в байте видеопамяти, содержащего код цвета пиксела с координатами (*250, 180*) в 16-цветном режиме видеоадаптера EGA с разрешением 640 x 350.

3. Определить номер пары бит в байте видеопамяти, содержащей код цвета пиксела с координатами (*250, 180*) в 4-цветном режиме видеоадаптера CGA с разрешением 320 x 200.

4. Чему должно быть равно содержимое регистра маски графического контроллера в 16-цветном режиме видеоадаптера EGA с разрешением 640 x 350 для выполнения операции записи в видеопамять кода цвета пиксела, расположенного на экране в позиции (*300, 125*).

5. Коды цветов скольких пикселов хранятся в шести байтах видеопамяти:

а). в 16-цветном режиме видеоадаптера EGA с разрешением 640 x 350;

б). в 4-цветном режиме видеоадаптера CGA с разрешением 320 x 200;

в). 2-цветном режиме видеоадаптера CGA с разрешением 640 x 200;

г). в режиме HiColor (32 тыс. цветов) видеоадаптера SVGA с разрешением 640x480;

д). в режиме HiColor (64 тыс. цветов) видеоадаптера SVGA с разрешением 640x480;

е). в режиме TrueColor (16 млн. цветов) видеоадаптера SVGA с разрешением 640x480;

ж). в 256-цветном режиме видеоадаптера VGA с разрешением 320 x 200.