VRIJE UNIVERSITEIT BRUSSEL

Computersystemen

WPO: Exercise Session 4

David Blinder
Raees K. Muhamad

Overview

- Video modes, palettes, how to draw to the screen
- Macros
- REP instruction sets
- Keyboard interrupts

Video modes

Video mode

When the 80386 PC boots in MS-DOS, it starts in text mode (3h).
In this mode, the screen cannot be addressed on a pixel per pixel basis, but rather displays complete ASCII symbols through hardware accelerated circuitry.

For pixel drawing, the video mode must be changed to a graphical mode.
We will mainly use video mode `13h`. It has:

- 320x200 pixels
- A palette of 256 colors
- 6 bits per RGB channel, totalling 2^{18} possible colors

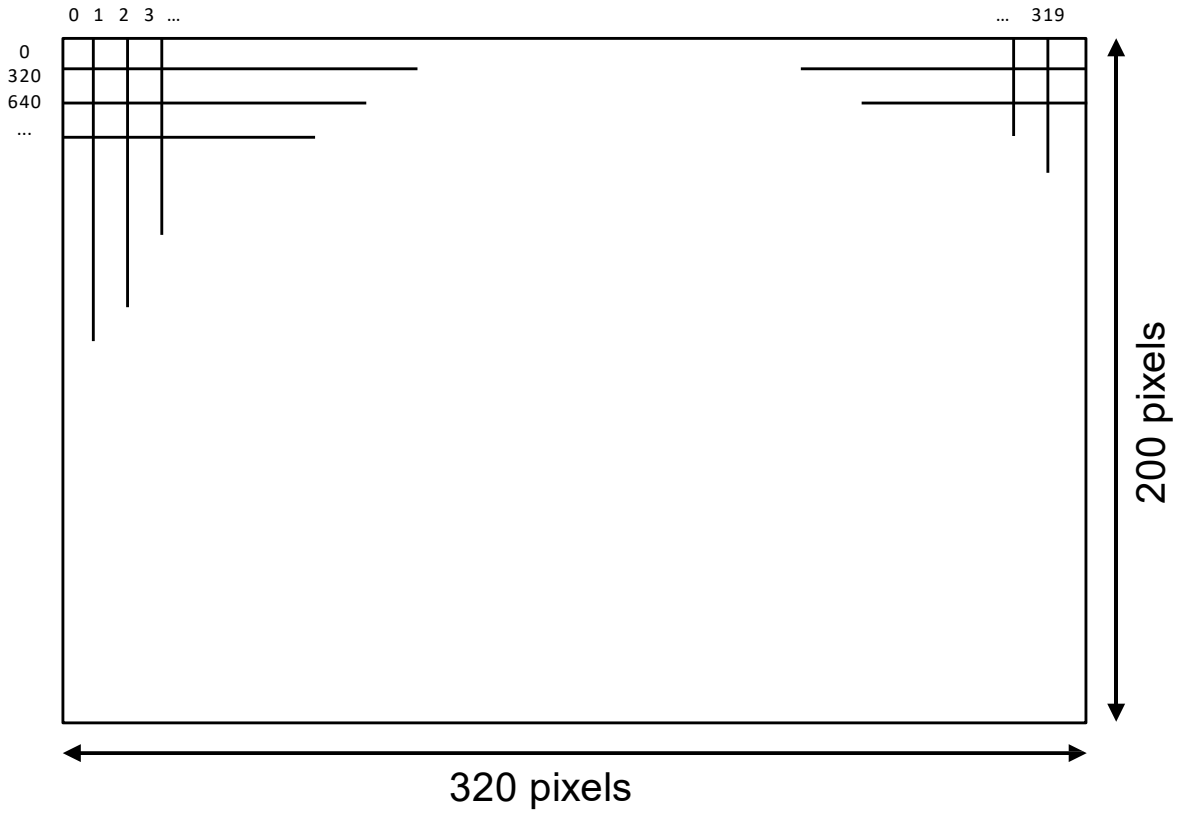
Set the video mode with `int 10h`, `AH=0`, `AL=`*mode.*

```
mov AX, 13h        ; video mode 13h
int 10h
```

Mode 13h: Video memory

Mode `13h` is very easy to program for, as it uses 8-bits, or 1 byte, per pixel in a linear frame buffer. This means that each pixel on the screen corresponds to exactly one byte in the video memory.

This video memory is always located at physical address `0A0000h` and it can be considered as a contiguous byte (or pixel) array.



Mode 13h: Video memory

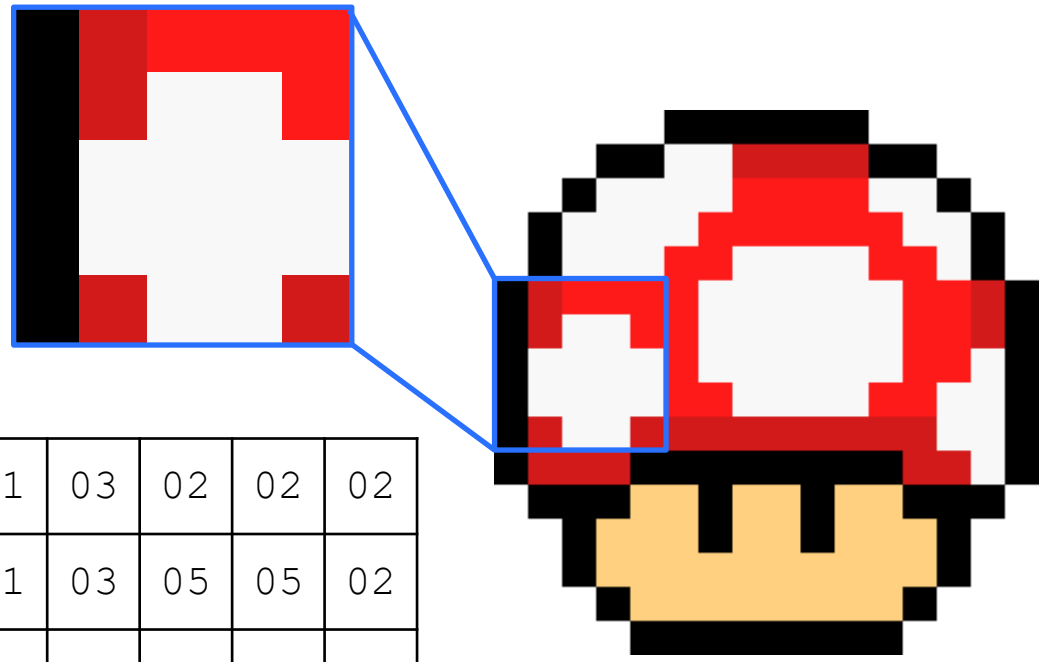
Drawing to the screen in mode `13h` is as simple as writing to video memory:

```
mov EDI, 0A0000h      ; start of video memory
mov AL, 15            ; color index 15 (white by default)
mov [EDI], AL         ; write pixel (0,0) (top-left)
add EDI, 2*320+10     ; go 2 lines down, 10 to the right
mov [EDI], AL         ; write pixel (10,2)
```

Mode 13h: Palette (example)

Palette	RGB		
00h	63	63	63
01h	00	00	00
02h	63	06	06
03h	53	06	06
04h	63	52	32
05h	63	63	63

e.g. background color



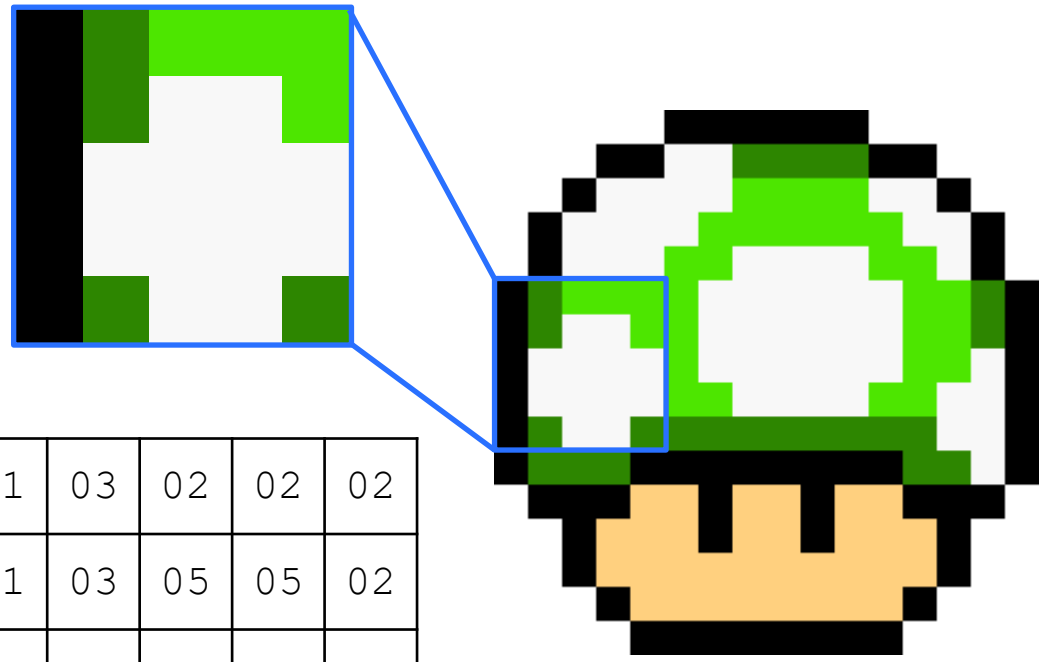
01	03	02	02	02
01	03	05	05	02
01	05	05	05	05
01	05	05	05	05
01	03	05	05	03

Video memory

Mode 13h: Palette (example)

Palette	
	00h
	01h
	02h
	03h
	04h
	05h

RGB		
63	63	63
00	00	00
19	**58**	**00**
11	**34**	**00**
63	52	32
63	63	63



01	03	02	02	02
01	03	05	05	02
01	05	05	05	05
01	05	05	05	05
01	03	05	05	03

Video memory

Changing the palette will affect drawn colors without changing video memory

Mode 13h: Palette

By default, the VGA card provides a standard color palette. However, it is possible to reprogram the color palette with customized colors. This can be done by communicating directly with the VGA card via IO ports (`in` and `out` instructions on the Intel 80386).

```
mov DX, 03C8h      ; DAC write port
mov AL, 0          ; index of first color to change
out DX, AL         ; write to IO
mov DX, 03C9h      ; DAC data port
mov AL, [r]        ; load red value (6-bit)
out DX, AL         ; write red value
mov AL, [g]        ; load green value (6-bit)
out DX, AL         ; write green value
mov AL, [b]        ; load blue value (6-bit)
out DX, AL         ; write blue value
```

Other video modes

Many video modes are available, but can be much more complicated to address. Some other modes to be aware of:

Mode 11h: 640x480 monochrome (black or white), one bit per pixel (every byte encodes 8 pixels). It's used in EXAMPLES/BADAPPLE

Mode 12h: 640x480 16 colors. Pixels are encoded in 4 bitplanes, which are not directly addressable. More complicated to work with than modes 11h or 13h.

VESA BIOS extensions: more "recent" video modes from the mid-90's. Supports resolutions of 640x480 and higher up to 24-bit color. Only recommended for image editor projects and such.

Macros

X86 MACROS

Macros allow for evaluating constant expressions, or even define parametrized sets of instructions.

For this exercise session, we will only use them for compile-time constants. We'll see more uses later.

```
; compile-time constants (with macros)
VMEMADR EQU 0A0000h        ; video memory address
SCRWIDTH EQU 320           ; screen width (for mode 13h)
SCRHEIGHT EQU 200          ; screen height
…
mov EDI, VMEMADR
mov ECX, SCRWIDTH*SCRHEIGHT
```

Simple keyboard interrupts

Keyboard BIOS Interrupt (int 16h)

Use `AH=0` if the program should block until a key is pressed, `AH=1` if not. As a result, you get the scancode in `AH` and the ASCII code in `AL`.

```
mov   AH, 01h     ; function 01h (test key pressed)
int   16h         ; call keyboard BIOS
jz    @@no_key_pressed    ; jump if no key was pressed
…
mov   AH, 00h     ; function 00h (get key from buffer)
int   16h         ; call keyboard BIOS
; process key code here
;(scancode in AH, ascii code in AL)
```

<u>Do not use this for interactive gameplay!</u> (see EXAMPLES/KEYB)

REP instruction sets

x86 REP instructions

When you want to replace the color palette, or more generally, move entire arrays, this can be done efficiently with REP instructions. *(Repeat String Operation Prefix)*

They generally involve ESI as the source address, EDI as the destination address and ECX as the amount of elements that should be moved.

Some examples: (https://kernfunny.org/x86/html/file_module_x86_id_279.html)

```
REP INSx                          ; input data from port DX
REP MOVSx                         ; move array contents
REP OUTSx                         ; output data to port DX
REP STOSx                         ; fill target array with AL/AX/EAX
REPE CMPSx / REPNE CMPSx          ; find (non)matching symbols
REPE SCASx / REPNE SCASx          ; scan for (non)matching AL/AX/EAX
```

x86 REP instructions

Specify whether you want to copy bytes/words/doublewords by appending b/w/d to the REP-instructions (e.g. STOSB/STOSW/STOSD, OUTSB/OUTSW/OUTSD, …)

```
DATASEG
  palette db 0, 0, 0, 63, 63, 63 ; defines black and white

CODESEG
cld
mov ESI, offset palette     ; set the palette (DAC) address
mov ECX, 2*3                ; set color 0 and 1
                            ; (2 indexes in total, 2 * 3 bytes)
mov DX, 03C8h               ; VGA DAC set port
mov AL, 0                   ; set start color index
out DX, AL                  ; signal index 0
inc DX                      ; DAC data port (03C9h = 03C8h + 1)
rep outsb
```

Exercises

Exercises

Note: **download the template** for these exercises

1. Change the video output to 320x200 resolution with 8-bit indexed colors (mode 13h) via the procedure `setVideoMode (input argument: mode`. Exit the program only when ESC is pressed. Restore to the default text mode 03h after exiting the program.

2. Create procedure `fillBackground` to fill the screen in a single color, provided as an argument. Use `rep stosb/stosw/stosd`.

3. Reprogram the palette to have 64 grayscale values, going from black (0,0,0) to white (63,63,63). Use this to draw a horizontal gradient on the screen, going from black (left) to white (right).

4. Implement the procedure `drawRectangle`, drawing a rectangle (border only) using 5 input parameters values: `x0`, `y0` (top-left corner coordinates), `w` (width), `h` (height); and `col` designating the drawing color.