Understanding DOS by CodeCracker/SND

Tools used:
1. Dos Box 7.4
For executing DOS program on Windows x64 (64 bits) I have to use DosBox
dosbox-74: DOS emulator, the advantage is that you can run DOS programs on Windo
ws 7 64 bits.
We execute the dosbox.exe
and we type this command:
```
MOUNT [Drive-Letter] [Local-Directory]
```
to mount Local-Directory on DriveLetter specified
like
```
mount X D:\DirectoryName
```

2. GUI Turbo Assembler
Just google for GUI Turbo Assembler v3.0.1.msi
This is a gui for the old turbo assembler.
It require Microsoft .NET Framework 4.
It uses dosbox.

How to use GUI Turbo Assembler guide:
Step 1: Create file (assembler code)
Enter assembler code and save the file,
attention: file name should be in dos style:
not bigger then 8 characters in size and shouldn’t contain
special characters like space (" ").
Step 2: Create the executable file (.exe)
Click on Tools->Build (Alt+B)
Now you will be asked "Source file has not been assembled. Do you want to assemb
le it now?"
Ofcourse choose "Yes" to that question.
Next question: "Source file has been assembled. Do you want to build it now?"
Ofcourse choose "Yes" to that question.
Step 3: Execute the exe file
You could choose Tools->Assemble, Build & Run (F9)
then you should see the program executed in a console window!
Alternative you could execute the generated exe file in dosbox.

3. Tool called: symdeb
A commandline DOS debugger, you must type command in command line
of the program!
Under dos command line we type:
```
symdeb exefilename
```
for exemple
```
symdeb CrackME!.EXE
```
in order to start debugging CrackME!.EXE
By entering t
(trace) in symdeb command line and pressing Enter
you will trace line by line!
? - program help
q - quit from program
d codesegment:address - will display data at that address
u codesegment:address - will display code at that address
r - (simple) show registers and dissasm to cs:ip
? E - this means asking what is E?
for editing enter "e cs:ip", the default data type is byte

For specifing the type of data:
<type> : Byte, Word, Doubleword, Asciz, Shortreal, Longreal, Tenbytereal
- you should enter only the first letter like B, W, D etc.

The program (symdeb) has an error: I doesn’t recognize whell the space (’ ’)
Stupid solution: don’t enter any space
so instead of: e B cs:ip
enter: eBcs:ip
It looks like sheet but it works! It is a bug from symdeb,
anyway nothing is perfect in this world!

4. Tools called: insight (insight.com)
Insight 1.24, a visual DOS debugger,
GUI for GDB debugger
First time press F10 to show the menu,
then click File->Load... F3

Most important commands:
Step over: F8
Step into: F7
Run : Ctrl+F9 - crap: when I use the hotkey program silence exits! (DON’T USE ho
tkeys!)
Ctrl+G: go to address code - specify segment:address like cs:address
Ctrl+D: go to address dump - data - specify segment:address like ds:address
Ctrl+R: switch bettwen 32 bits registers and 16 bits registers
Ctrl+S: search
Ctrl+B: Copy/Fill/Write/Read
Ctrl+M: memory blocks
quit - exit from current crap!
Alt+F5 - show dos window
Alt+U - Show and Dump User Screen correspondingly
Alt+L - something regarding dump
Alternative tool: SD

5. The Decompiler project was initiated in order to decompile MS-DOS EXE
and COM binaries. The project has both a command-line and a GUI tool:
https://sourceforge.net/projects/decompiler/
Use the following command with the command-line tool to decompile COM programs:

```
decompile --default-to ms-dos-com myprog.com
```

In the GUI, use the menu command File > Open as... to open the COM file and
specify a start address like 0800:0100.

Honorable tools to mention:
CUP386 - generic manual unpacker
DeGlucker - DOS debugger, interface similarly with SoftIce
but not terminate and stay resident (TSR)

TASM EXAMPLES (the power of examples)
Example 1: Simple hello word program

```
; Turbo Assembler Copyright (c) 1988, 1991 By Borland International, Inc.
; HELLO.ASM - Display the message "Hello World"
.model small
.stack 100h
.data
HelloMessage DB ’Hello, world!’,13,10,’$’
.code
start:
```

```
mov ax,@data
mov ds,ax ;set DS to point to the data segment
mov ah,9 ;DOS print string function
mov dx,OFFSET HelloMessage ; point to "Hello, world!"
int 21h ;display "Hello, world!"
mov ah,4ch ;DOS terminate program function
int 21h ;terminate the program
end start
```

Example 2: Copy data with repz movsb
Source DS:SI, Destination ES:DI

```
; MOVSB.ASM - Copy string from source to destination
.model small
.stack 100h
.data
SourceStr1 DB ’String to be copied!’,’$’
DestinationStr2   DB   50   DUP(0)
.code
start:
mov ax,@data
mov ds,ax     ;initialize ds
mov es,ax     ;and es
lea si,[SourceStr1] ;si points to source string
lea di,[DestinationStr2] ;di points to destination string
mov cx,5      ;number of chars (bytes) to be copied
; in cx supposed to be length in chars of SourceStr1
rep movsb     ;copy the string

mov ah,4ch ;DOS terminate program function
int 21h ;terminate the program
end start
```

Example 3: Altering code section

```
; csalter.asm - alter code (cs), in DOS no protections
.model small
.stack 100h
.code
start:
mov ax,01 ; this is the instruction which will be altered
mov word ptr [cs:start],09090h ;write two bytes (2 nops) on start
mov byte ptr [cs:start+2],090h ; write one byte (1 nop) on start+2
; yep, no memory protections in DOS, is kind of stupid!

mov ah,4ch ;DOS terminate program function
int 21h ;terminate the program
end start
```

In DOS you can only use psihical memory and doesn’t have virtual memory,
like Windows. In DOS memory is not linear like in Windows but
segmented memory, google for x86 memory segmentation,
The physical address is calculated by shifting the segment address (16-bit)
left 4 times and adding it with the 16-bit offset address.
In other words by multiplying the segment with 10 hexadecimal
(which is 16 in decimal), and adding offset address.
200A:B608 and 2138:A328 gives the same physical address: 2B6A8

The intersting part: if we substract 1 from cs (segment address)
and we add 10 hexadecimal to offset address we get same psihical address.
Naturally same physical address means same value!