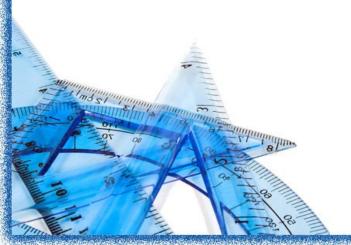
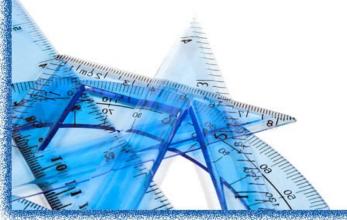
Structure of DOS application programs



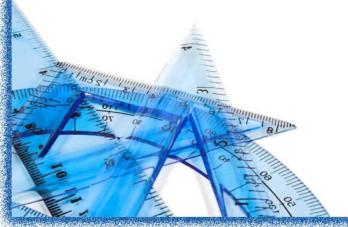
Contents:

- 1. PSP
- **2.** .COM and .EXE
- 3. TSR: Terminate and Stay Resident Programs



About DOS...

- Remember the new versions of Windows like XP does not actually contain DOS.
- Till Windows 98, it allowed us to boot from DOS mode.
- What we have is just the command prompt which is a partial emulation of it.



Type and copy commands

- Command types are:
 - Internal (internsic) : type ,copy, del
 - Carried out by code embedded in Command.Com
 - External (extrensic): chkdsk, backup
 - Stored in disk files.
 - Need to be loaded in transient program area.
 - After completions of its work is discarded from the memory.
 - Batch files: Contain intrensic, extrensic and batch commands.
 - Processed by special interpreter that is in the transient portion of command.com.

 www.free-power-point-templates.com

When you type anything on command prompt, what happens??

- Command.Com checks if it is internal command that can be executed directly.
- If not, search for external command, batch file with same name.
 - Search from current directory to each of the directories in most recent path.
 - In each directory, it tries to find a file with extension .COM or .EXE or .BAT.
 - If search fail, it gives bad command or filename.

When you type anything on comman prompt, what happens?? (ctd...)

- If .Com or .EXE is found:-
 - Command.Com uses DOS Exec function to load and execute it.
 - It is the EXEC function that builds Program Segment Prefix above the resident portion of Command.com in transient program area.
 - EXEC function loads the program just above the PSP.

What is PSP??

- Program Segment Prefix is 256 bytes long.
- A data structure used in DOS to store the state of the program.
- It is set up by MS-DOS at the base of the memory block allocated to a transient program.
- Contains various linkages and pointers needed by the application program.
 - We are going to use this to execute our simulation of type and copy

PSP - DOS Program Segment Prefix Layout

Description

Offset Size

00	word	machine code INT 20 instruction (CDh 20h)
02	word	top of memory in segment (paragraph) form
04	byte	reserved for DOS, usually 0
05	5bytes	machine code instruction long call to the DOS function dispatcher (obsolete CP/M)
06	word	.COM programs bytes available in segment (CP/M)
A0	dword	INT 22 terminate address; DOS loader jumps to this address upon exit; the EXEC function forces a child process to return to the parent by setting this vector to code within the parent (IP,CS)
0E	dword	INT 23 Ctrl-Break exit address; the original INT 23 vector is NOT restored from this pointer (IP,CS)
12	dword	INT 24 critical error exit address; the original INT 24 vector is NOT restored from this field (IP,CS)
16	word	parent process segment addr (Undoc. DOS 2.x+) COMMAND.COM has a parent id of zero, or its own PSP
18	20bytes	file handle array (Undocumented DOS 2.x+); if handle array element is FF then handle is available. Network redirectors often indicate remotes files by setting these to values between 80-FE.
2C	word	segment address of the environment, or zero (DOS 2.x+)
2E	dword	SS:SP on entry to last INT 21 function (Undoc. 2.x+) \neq
32	word	handle array size (Undocumented DOS 3.x+)
34	dword	handle array pointer (Undocumented DOS 3.x+)
38	dword	pointer to previous PSP (deflt FFFF: FFFF, Undoc 3.x+) $+$
3C	20bytes	unused in DOS before 4.01 +
50	3bytes	DOS function dispatcher CDh 21h CBh (Undoc. 3.x+) $+$
53	9bytes	unused
5C	36bytes	default unopened FCB #1 (parts overlayed by FCB #2)
6C	20bytes	default unopened FCB #2 (overlays part of FCB #1)
80	byte	count of characters in command tail; all bytes following command name; also default DTA (128 bytes)
81	127bytes	all characters entered after the program name followed by a CR byte

What can I do with PSP??

- We can write a program to simulate type command.
- Type command types the contents of a file.
- At command prompt if I write
 - c:\type first.asm

It will display the contents of the file.

Similarly we can write our program where instead of type it would be say

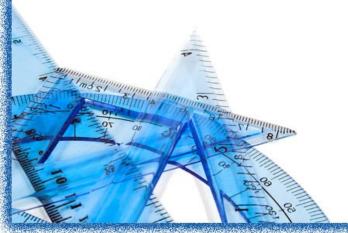
c:\mytype first.asm

Using PSP

- With 80h and 81h offset of PSP, we will read the file names.
- 80h will give us the length of the command string
- 81h all the characters entered after the no. of arguments.
- This is as good as command line arguments in C.
 - How to get the address of PSP?
 - Int 21h function 62h

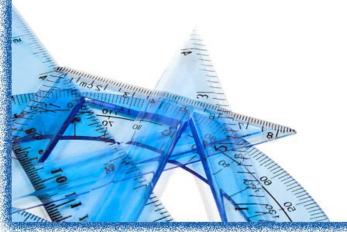
Which interrupts do I use??

- Int 21h, function 3c,3d,3E,3f,4041...
- These will be used for creating, opeting, closing files and so on..
- Using them we will perform the file operations.



 Programs that run under DOS are in two flavours:

- -COM
- EXE

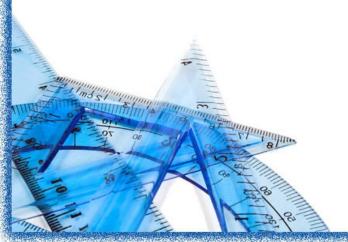


Comparison between .COM and .EXE Files

Parameters	.COM File	.EXE File
Location	On disk files which hold an absolute image of the machine instructions to be executed	On disk in a special type of file
Segments	stack and data and code segment are in one segment	Multiple code, data and stack segments.
Program size	64kbytes	As large as the available memory
Initialization	CS Starts at 100h	It is not fixed
Size of file	Exact same as the program	Program size + size of header
Subroutine calls	NEAR	NEAR and FAR
Loading time	Compact and faster	Header, relocation table and other information used by DOS Along with code, so loading time more
Header	No	Yes(For relocation process)

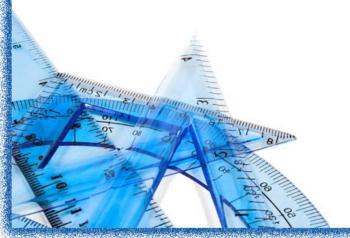
Type of programs

- Transient programs
 - .EXE
 - .COM
- TSR programs
 - .COM



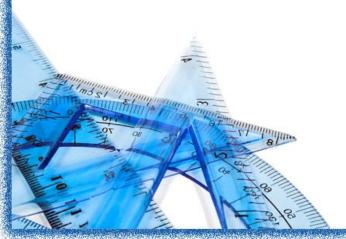
TSR Programs

- Can be loaded after DOS is loaded,
- Stay in the memory, even if they are not active on your screen,
- They appear to exit, but remain in the memory to perform tasks in the background,



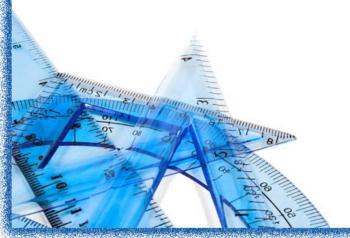
Structure of TSR

- Divided into three sections:
 - Data area
 - Resident routine and
 - Initialisation routine



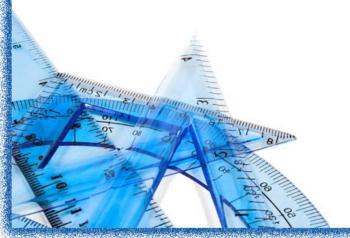
Data Area

- Where different data definitions are included as per the requirement of program
- e.g. Original entry in interrupt vector table, temporary space for registers etc.



Resident Routine

- Portion of program which will be made resident in the memory to perform specified task.
- During execution of specified task original register contents may get change so these contents must be preserved and again loaded before calling original interrupt service routine

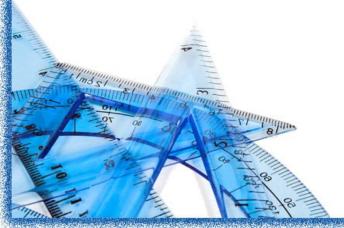


Initialisation Routine

- Does the preliminary work to make resident routine stay resident in the memory,
- It executes only once,
- It performs following steps
 - 1) Get the original address of specified interrupt from IVT and save it
 - 2) Store the address of resident program in the IVT in place of original address,
 - 3) Calculate the size of the resident routine including Data area and PSP. Reserve the memory area of this size and make the program resident

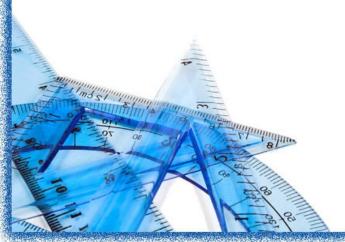
Initialisation Routine Continued...

- To perform these three steps we use three INT 21H functions
 - Function 35H: Get Interrupt Address
 - Call with AH=35H, AL= int #
 - Returns vector address in ES:BX
 - Function 25H: Set Interrupt Address
 - Call with AH=25H, AL=int #, DS:DX= Segment:Offset of interrupt handling routine
 - Function 31H: Make program resident
 - Call with AH=31h, AL=Return code, DX: amount of memory to reserve.



Examples

- Examples:
 - Beep system speaker on Key Press.
 - Display Real Time Clock
 - Screen Saver



```
TITLE
         A23RESID (COM)
                         Resident program: Beep if use
                         Esc kev
CODESEG
         SEGMENT PARA
         ASSUME CS: CODESEG
         ORG
                100H
BEGIN:
         JMP
                B10INIT
                                      ;Jump to initialization
                                      :INT 09H address
SAVEINT9 DD ?
DURATION DW
                100H
A10TEST: PUSH
                AX
                                      ; Save registers
         PUSH CX
                AL,60H
                                      ;Get keystroke from port
         IN
         CMP AL, 01
                                      ;Scan code 01 (Esc)?
         JNE
               A50EXIT
                                      ; no, exit
         IN
                AL, 61H
                                      ;Get port status
         PUSH
                AX
                                     ; and save
         OR
                AL,00000011B
                                      ;Turn on speaker
         OUT
                61H.AL
                CX,512H
         MOV
                                      :Length
A20:
         LOOP
                A20
         OR
                AL,00000010B
                                      :Set bit 1 on
         MOV
                CX,512H
A30:
         LOOP
                A30
         POP
                AX
                                     ; Port status
                                     ;Turn off speaker
         AND
                AL,11111100B
         OUT
                61H, AL
A50EXIT: POP
                CX
                                      ; Restore registers
         POP
                AX
         JMP
                CS:SAVEINT9
                                      ; Resume INT 09H
                                      ; Initialization:
B10INIT:
         CLI
                                      ; Prevent interrupts
                AH, 35H
         MOV
                                      :Get address of INT 09H
         MOV
                AL, 09H
                                      ; in ES:BX
         INT
                21H
                WORD PTR SAVEINT9, BX; and save it
         MOV
                WORD PTR SAVEINT9+2, ES
         VOM
         MOV
                AH, 25H
         VOM
                AL,09H
                                     ; Set new address for
                DX, OFFSET A10TEST ; INT 09H in A10TEST
         MOV
         INT
                21H
         VOM
                AH, 31H
                                      ; Request stay resident
         VOM
                DX, OFFSET B10INIT
                                      :Set size
         STI
                                      ;Restore interrupts
         INT
                21H
CODESEG
         ENDS
         END
                BEGIN
```