

Segments and Groups:

Name	Size	Length	Align	Combine	Class
CODE_SEG	16 Bit	002C	Para	Private	'CODE'
STACK_SEG	16 Bit	004C	Para	Stack	'STACK'

Procedures, parameters and locals:

Name	Type	Value	Attr
LAB15	P Far	0000	CODE_SEG Length= 002C Public
RPTDELAY	L Near	001C	CODE_SEG
DELAY	L Near	001F	CODE_SEG

Symbols:

Name	Type	Value	Attr
0 Warnings			
0 Errors			

FIGURE L15.2 (b) (Continued)

Part 3: Output Port for the Speaker in the Original IBM PC

As described earlier, the I/O address 61H points to the output port that controls the speaker and other peripherals, such as cassette and keyboard. Bit 0 on the port enables the timer to supply a clock signal, and bit 1 is used to enable the clock signal to the speaker. We will first read the status of these bits on the port and then change them to drive the speaker. Finally, we will run the program developed in Part 2.

Check	Step	Procedure
_____	1.	Load the DEBUG program.
_____	2.	Assemble and execute the instruction needed to input the contents of port 61H into the AL register. What are these contents? _____ What do bits 0 and 1 indicate with respect to the 8253 timer clock and the enable bit for the clock to the speaker? _____
_____	3.	Change the contents of AL so that the two least significant bits are logic 1. Do not change any other bits. What are the new contents of AL? _____
_____	4.	Assemble and execute the instruction needed to output the new contents of AL to the output port at 61H. What happens and why? _____ _____
_____	5.	Reload AL with the original contents as recorded in step 2. Again, assemble and execute an instruction to output the contents in AL to the output port at 61H. What happens and why? _____ _____

6. Load the program LAB15.EXE.
7. Unassemble the program to verify its loading.
8. Execute the program. What happens?  
\_\_\_\_\_  
\_\_\_\_\_

9. Change the program with DEBUG so that the divisor loaded into the 8253 is one-fourth the original value. Execute the program again. What is the difference when compared to the operation observed in step 8?  
\_\_\_\_\_  
\_\_\_\_\_

10. Change the program so that the tone duration is doubled. Execute the program. Describe the difference when compared to the results observed in steps 8 and 9.  
\_\_\_\_\_  
\_\_\_\_\_

LABORATORY 16: EXPLORING THE INTERRUPT SUBSYSTEM OF THE PC

Objective

Learn how to:

- Determine the address of an interrupt service routine.
- Explore the code of an interrupt service routine.
- Execute software interrupt service routines to determine the equipment attached to the PC and the amount of RAM.
- Execute the software interrupt service routines for print screen and the system boot.
- Use the interrupt 21 function calls to read and set the date and time.

Part 1: Interrupt Vector Table

The PC's interrupt vector table is located in the first 1024 bytes of RAM memory, that is, from address 00000H through 03FFH. Each vector takes up four bytes of memory; therefore, there are 256 vectors in the interrupt vector table. Here we will explore the contents of the interrupt vector table and use this information to calculate the starting address for several interrupt service routines. Check off each step as it is completed.

Check	Step	Procedure
_____	1.	Load the DEBUG program.
_____	2.	Dump the contents of the memory locations starting at 0:0. Compute the starting address of the service routines for the following interrupt types:  Interrupt type 2 (NMI) _____ Interrupt type 8 (Timer) _____ Interrupt type 9 (Keyboard) _____ Interrupt type 10 _____ Interrupt type 11 _____ Interrupt type 12 _____ Interrupt type 13 _____ Interrupt type 14 (Diskette) _____ Interrupt type 15 _____

## Part 2: Exploring the Code of an Interrupt Service Routine

Now we know the starting address of several interrupt service routines in ROM. Next we will examine the instruction sequence in the NMI service routine.

Check	Step	Procedure
_____	1.	Unassemble the code of the service routine for NMI.
_____	2.	What is the address of the last instruction in the NMI service routine?
_____	3.	Does this service routine invoke another software interrupt? _____ If yes, which one? _____
_____	4.	Does this service routine call another routine? _____ If yes, where is that routine located? _____
_____	5.	Describe the operation implemented by the three instructions that follow the IN AL, 62 instruction the beginning part of the service routine.
_____	6.	From the hardware discussion in Chapter 12 of <i>The 8088 and 8086 Microprocessors: Programming, Interfacing, Software, and Applications, 4th Ed.</i> , determine the conditions in the hardware that will enforce bypassing of the rest of the instructions (those immediately after the three) of the NMI routine.
_____	7.	Now determine the conditions of the hardware that will enforce executing the rest of the instructions of the service routine.

## Part 3: Determining the PC Equipment and RAM Implementation

The service routines for INT 11H and INT 12H are used to determine what equipment is attached to the PC and how much RAM is implemented. Execution of the service routine for INT 11H returns a word in AX. The bits of this word indicate what type of equipment is attached to the PC. The bits of the word are encoded as follows:

- Bit 0 = 1 if the PC has floppy disk drives attached
- Bits 3,2 = system board R/W memory size (00 = 16K, 01 = 32K, 10 = 48K, 11 = 64K)
- Bits 5,4 = video mode (00 = unused, 01 = 40X25 BW using color card, 10 = 80X25 BW using color card, 11 = 80X25 BW using monochrome card)
- Bits 7,6 = number of floppy disk drives (00 = 1, 01 = 2, 10 = 3, 11 = 4 only if bit 0 = 1)
- Bits 11,10,9 = number of RS232 cards
- Bits 15,14 = number of printers attached
- Other bits = don't care

By executing the service routine for INT 12H, a number is returned in AX that indicates the number of kilobytes of R/W memory in the system. In this part of the laboratory, we will execute these software interrupt routines.

**NOTE:** Some variations in bit meanings may be experienced if you are using a compatible or PC other than an original IBM PC. In this case, refer to the reference documentation for the PC to determine the bit functions.

Check	Step	Procedure
_____	1.	Load the DEBUG program.
_____	2.	Display the contents of all registers. What is the value of AX? _____
_____	3.	Assemble the instruction INT 11H followed by an NOP instruction at the current code segment address. Execute up to the NOP instruction and then display the registers. What are the new contents of AX? _____ What do the contents of AX indicate with respect to the following equipment? a. Floppy disk drives present or not. _____ b. System board RAM size. _____ c. Video mode. _____ d. Number of floppy disk drives. _____ e. Number of RS232 cards. _____ f. Number of printers attached. _____
_____	4.	Assemble the instruction INT 12H followed by an NOP instruction at the current code segment address. Execute up to the NOP instruction and then display the new contents of the registers. What is the value in AX? _____
_____	5.	How much R/W memory is in your PC? _____
_____	6.	Quit the debugger.

## Part 4: Print Screen and System Boot Interrupts

Now we will examine the operation of two other interrupt service routines. The INT 5H service routine can be used to print what is displayed on the screen. On the other hand, the service routine for INT 19H is used to boot the system. Here we will execute these interrupts to observe the functions that they perform.

Check	Step	Procedure
_____	1.	Load the DEBUG program.
_____	2.	Assemble the instruction INT 5H followed by an NOP instruction at the current memory location pointed to by CS:IP. Execute up to the NOP instruction and then describe what happens.
_____	3.	Assemble the instruction INT 19H followed by an NOP instruction at the memory location pointed to by CS:IP. Execute up to the NOP instruction. What happens?

## Part 5: The INT 21H Function Calls

INT 21H is provided to invoke DOS operations for a wide variety of functions, such as character I/O, file management, and date and time setting and reading. Here we will learn to use the time setting and reading functions. To call any function, the register AH is loaded with the function number and then the instruction INT 21H is executed. If the specified function cannot be performed, DOS returns FF<sub>16</sub> in the AH register.

Check	Step	Procedure
_____	1.	Read the program in Figure L16.1 and determine the following: a. Function number and AH contents to read date. _____ b. Function number and AH contents to read time. _____ c. Function number and AH contents to set date. _____ d. Function number and AH contents to set time. _____

- e. What date is already set by the system? \_\_\_\_\_
- f. What time is already set by the system? \_\_\_\_\_
2. Load the DEBUG program.
3. Assemble the program starting at the address specified by the current CS:IP.
4. Execute the program up to the point where it reads the date. Display all registers. Determine the date. \_\_\_\_\_
5. Execute the program up to the point where it reads the time. Determine the time. \_\_\_\_\_
6. Execute the program up to the point where it loads registers to set a new date. Note the contents of registers CX, DH, and DL. How have they changed? \_\_\_\_\_
7. Execute the program up to the point where it loads the registers to set a new time. Enter a new date and note the contents of the CH, DH, CL, and DL registers. How have they changed? \_\_\_\_\_
8. Quit the debugger. Do you think a new date and time are set? \_\_\_\_\_
9. Use DOS commands TIME and DATE to verify that the new date and time are set. You may now want to reset your system back to the original date and time.

```

mov ah,2ah      ; get date, AL=day of the week, CX=year, DH=month, DL=day
int 21h

mov ah,2ch      ; get time, CH=hour, CL=minutes, DH=seconds,
int 21h          ; DL=hundredths of a second

mov ah,2bh      ; set date as follows:
mov cx,07c3     ; cx = year
mov dh,0a       ; dh = month
mov dl,0a       ; dl = day
int 21h

mov ah,2d       ; set time as follows:
mov ch,11h      ; ch = hour
mov dh,10h      ; dh = seconds
mov cl,32h      ; cl = minutes
mov dl,4ah      ; dl = hundredths
int 21h

```

FIGURE L16.1 Program for Laboratory 16, Part 5.

## LABORATORY 17: USING BIOS ROUTINES FOR KEYBOARD INPUT AND DISPLAY OUTPUT

### Objective

Learn how to:

- Use the read keyboard and display character BIOS routines.
- Display prompt messages on the screen.
- Display characters entered at the keyboard on the screen.
- Write a program that makes decisions based on inputs from the keyboard.

### Part 1: Executing a Keyboard and Display Interaction Routine

The use of the keyboard and display in a program is actually quite easy. This is because the BIOS of the PC contains special routines to control them. These routines can be used by simply inserting an appropriate software interrupt instruction in the program. For example, the *read keyboard* routine is called by loading 0 into the AH register and then executing the instruction

```
INT 16H
```

When this instruction is executed, the program waits looking for a key entry from the keyboard. As a key is depressed, its corresponding ASCII code is placed into AL and control is returned to the next instruction in the main part of the program. Instructions in the main program can read this ASCII data from AL and process it in the appropriate way.

The second routine that we will consider at this point is the *display character* routine. This routine is invoked by loading AL with the ASCII code for the character that is to be displayed, loading AH with 14<sub>10</sub>, and then executing the software interrupt instruction

```
INT 10H
```

Execution of this routine causes the ASCII code in AL to be read and the corresponding character displayed on the screen.

These are not the only two routines provided in the BIOS. It also includes routines for control of the printer, disk drives, and cassette. Let us now work with some of these routines. *Check off each step as it is completed.*

Check	Step	Procedure
_____	1.	Bring up the DEBUG program.
_____	2.	Assemble the program that follows into memory starting at address CS:100.
		CS:100H MOV SI,300H ; _____
		CS:103H MOV AH,00H ; _____
		CS:105H INT 16H ; _____
		CS:107H MOV [SI],AL ; _____
		CS:109H INC SI ; _____
		CS:10AH CMP AL,0DH ; _____
		CS:10CH JNZ 103H ; _____
		CS:10EH MOV SI,300H ; _____
		CS:111H MOV AH,0EH ; _____
		CS:113H MOV AL,[SI] ; _____
		CS:115H INT 10H ; _____
		CS:117H INC SI ; _____
		CS:118H CMP AL,0DH ; _____
		CS:11AH JNZ 111H ; _____
		CS:11CH NOP ; _____

Disassemble the program to verify that it has loaded correctly. Write comments for the program to explain what each instruction does. What is the ending address of the program? \_\_\_\_\_

How many bytes of memory does the program take up? \_\_\_\_\_

What condition stops the loop implemented with the first JNZ instruction? \_\_\_\_\_

How many times does this loop get repeated? \_\_\_\_\_

What is the purpose of this loop?

\_\_\_\_\_

What condition causes the jump performed by the second JNZ instruction?

\_\_\_\_\_

Describe the operation performed by this loop.

\_\_\_\_\_