KEYBOARD

nibaran@cse.jdvu.ac.in

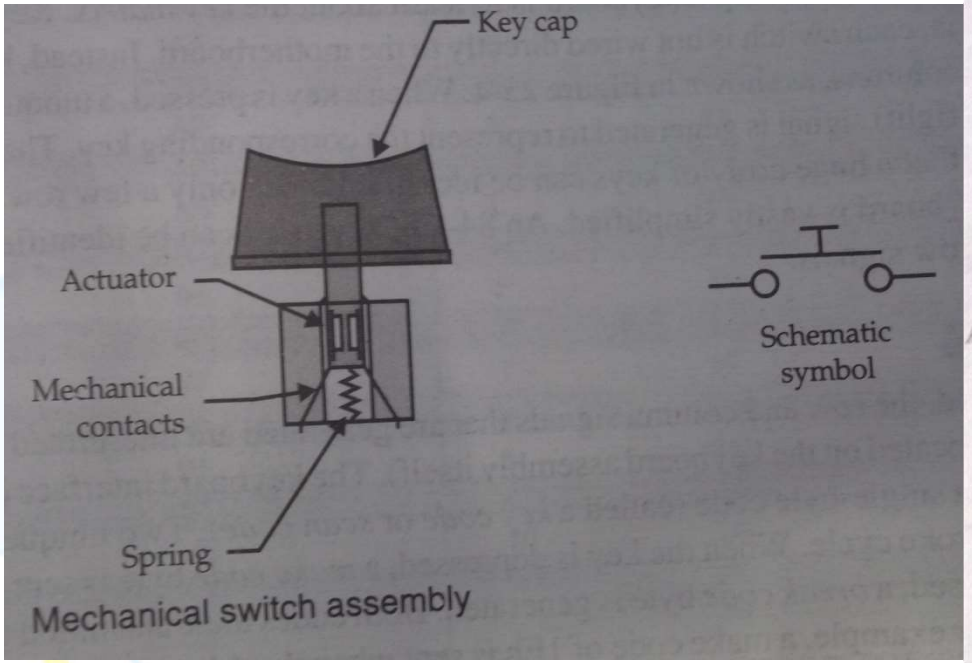
KEYBOARD

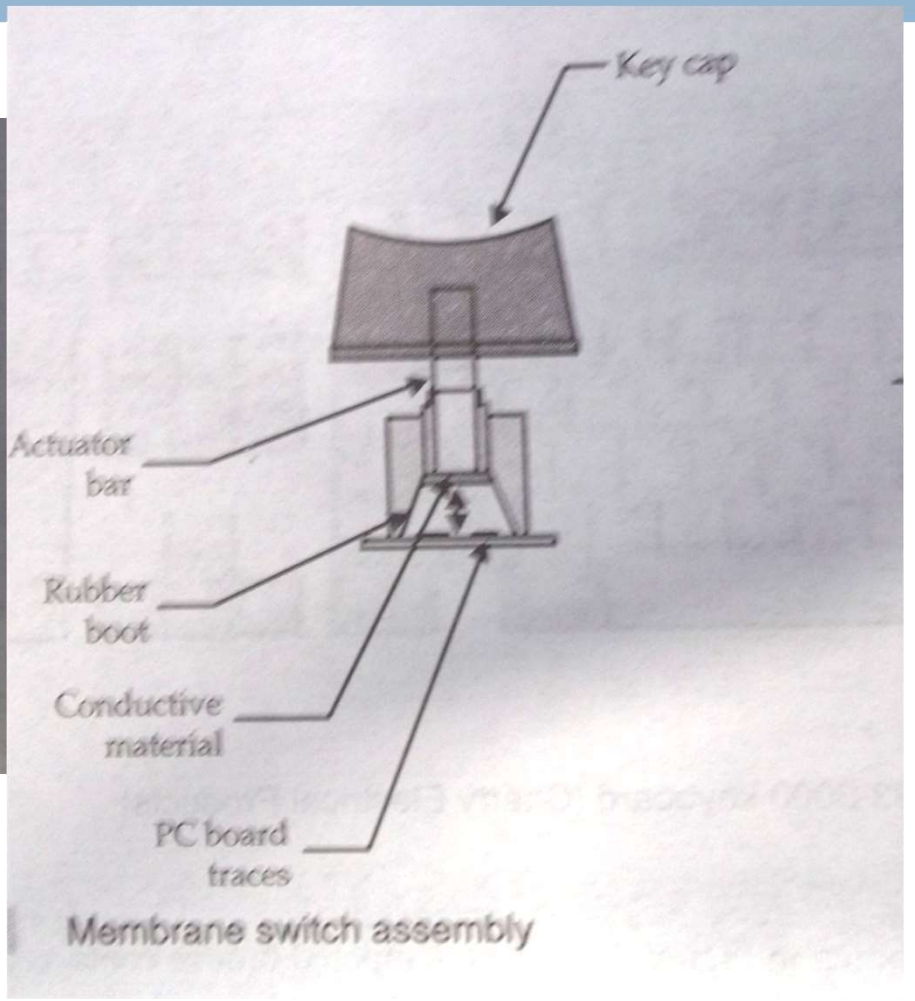
- In standard Keyboard three types of keys are present
  - Standard characters
    - A-Z, 0-9, %,$,#
  - Extended function keys
    - Program function keys such as F1 and shift+F1
    - Numeric keypads keys with Numlock toggled off: Home, End, Del, Ins, PageUp, and Page up and duplicate keys for them on the extended keyboard.
    - Alt+alpabetics and Alt+program function keys.
  - Special keys

Special Keys

- Alt, ctrl, and shift, which normally works association of other keys as well caps lock, numlock, and scrolllock which indicate a condition. Bios does not deliver these keystroke as ASCII characters to program. Instead BIOS treats these differently from other keys by updating their current state in the shift status byte in the BIOS keyBoard area
  - Original PC was 83 keys
  - Enhanced keyboard 101 keys
  - Windows has 104 keys

DIFFERENT TYPES OF KEY SWITCH ASSEMBLY





nibaran@cse.jdvu.ac.in

KEYBOARD BUFFER

- Keyboard buffer
  - Located at 40:1Eh data area location of bios
  - Types upto 15 characters at a time
- When a key is pressed or released, key board processor(intel 8048 ;8 bit processor) senses the pressing and releasing of keys. Based on the key(s) the keyboard sends code to another 8-bit processor, INTEL 8042, system board. L;

KEY ENTRY PROCESS

- When a key is pressed. Keyboard's processor generates the key's scan code and requests INT 09h
- The keyboard (int 9) interrupt service routine reads the scan code from the keyboard input port and processes the scan code as appropriate.
  - Note that the scan code the system receives from the keyboard microcontroller is a single value even though some keys on the keyboard represent up to four different values.
    - For example the "A" key on the keyboard can produce A a ctrl-A or alt-A. The actual code the system yields depends upon the current state of the modifier keys (shift ctrl alt capslock and numlock). For example if an A key scan code comes along (1Eh) and the shift key is down the system produces the ASCII code for an uppercase A

KEY ENTRY PROCESS

- When the scan code arrives at the PC a second microcontroller chip receives the scan code, does a conversion on the scan code makes the scan code available at I/O port 60h and then interrupts the processor and leaves it up to the keyboard ISR(BIOS routine) to fetch the scan code from the I/O port.
- The routine combines the scan code with its associated ASCII character and delivers the two bytes to the keyboard buffer.

nibaran@cse.jdvu.ac.in

ENTER THROUGH ASCII CODE

- The keyboard ISR provides a special facility that lets you enter the ASCII code for a keystroke directly from the keyboard.
  - Hold down the alt key and typing out the decimal ASCII code (0..255) for a character on the numeric keypad.
    - The keyboard ISR will convert these keystrokes to an eight-bit value attach at H.O. byte of zero to the character and use that as the character code.

THE INTERRUPT DURING KEY PRESS

- The PC keyboard actually generates two scan codes for every key you press. It generates a down code when you press a key and an up code when you release the key.
- If you hold the key more than one-half second, the processor become typematic; means key operation repeats automatically

KEYBOARD BUFFER

- The keyboard ISR inserts the 16 bit value into the PC's type ahead buffer. The system type ahead buffer is a circular queue that uses the following variables
  - 40:1A - HeadPtr word ?
  - 40:1C - TailPtr word ?
  - 40:1E - Buffer word 16 dup (?)

m	n	<enter>	*	a	b	c	d	e	f	g	h	I	J	k	l
41E	420	422	424	426	428	42a	42c	42e	430	432	434	436	438	43a	43c

Note that the TailPtr variable always points at the next available location in the type ahead buffer. Since there is no "count" variable providing the number of entries in the buffer we must always leave one entry free in the buffer area; this means the type ahead buffer can only hold 15 keystrokes not 16.

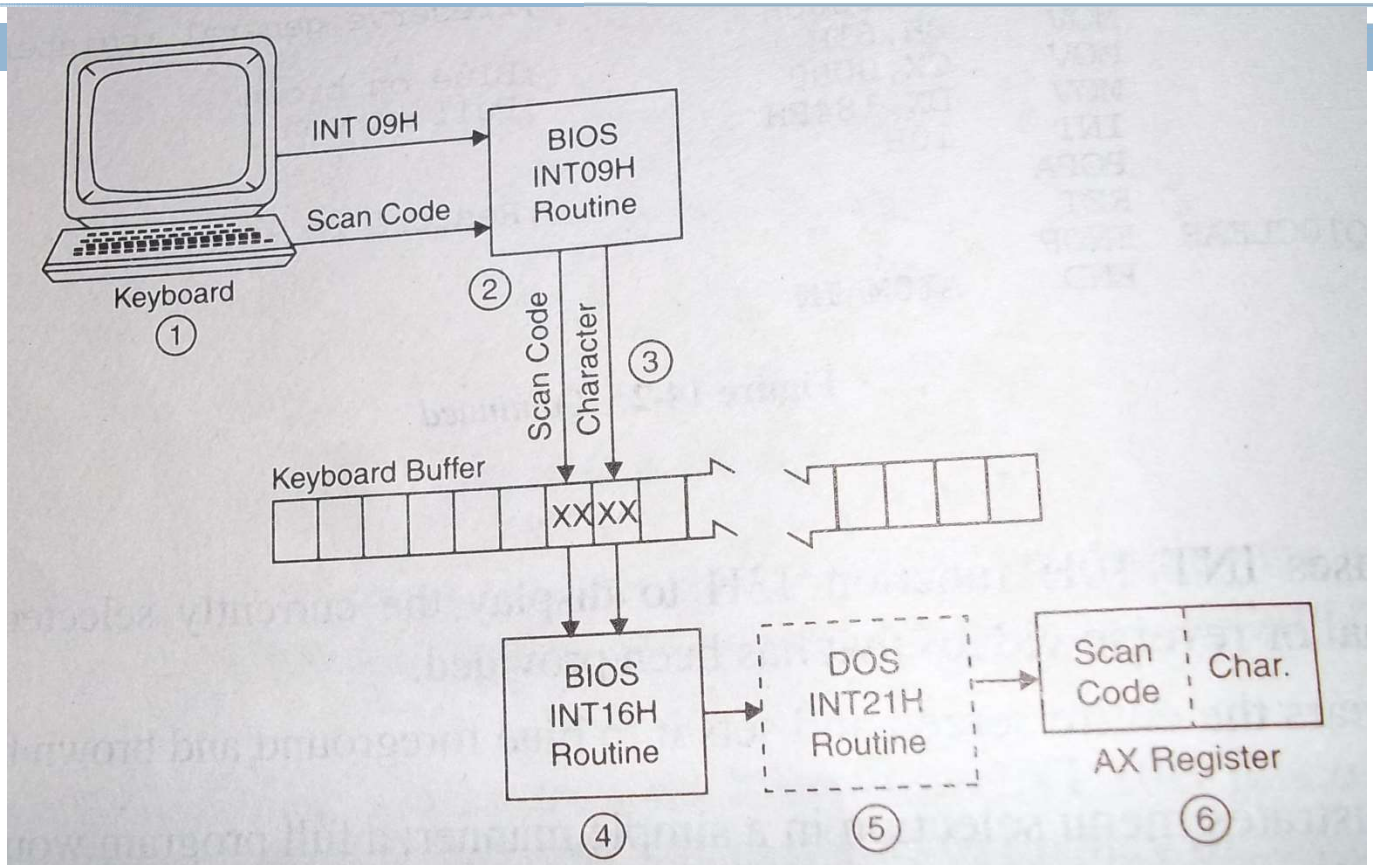
nibaran@cse.jdvu.ac.in

BIOS KEY BOARD DATA AREA

- In BIOS data area started from 40[0]h in low memory location include two keyboard data areas that indicate the current status of the control keys.
- Keyboard Data area 1 contain two byte
  - First byte at 40:17
  - Second byte at 40:18
- Keyboard Data area 2 contain keyboard buffer

	40:17	40:18
Bit	Action	Action
0	Right shift pressed	Left ctrl pressed
1	Left shift pressed	Left alt pressed
2	Right ctrl pressed	SysReq pressed
3	Right Alt pressed	Ctrl/NumLock(pause) active
4	Scroll lock state active	Scroll Lock pressed
5	Numlock state active	NumLock pressed
6	Capslock state active	CapsLock pressed
7	Insert active	Insert pressed

KEYBOARD INTERFACE



1. Keyboard generates INT 09H.
2. INT 09H operation accepts scan code from keyboard and finds its associated character (if any).
3. INT 09H delivers character and scan code to the keyboard buffer.
4. & 5. Program requests INT 16H either directly or via INT 21H.
6. INT 16H accesses buffer and delivers character to AL and scan code to AH.

INT 21 H FUNCTION

Number	Functions
01	Keyboard input with echo
07	Direct keyboard input without echo. The operation does not respond to a Ctrl+Break request
08	Keyboard input without echo
0A	Buffered keyboard Input
0B	Check keyboard status. This operation returns FFH in AL if an input character is available in the keyboard buffer and 00h if no character is available.

0C	This operation associated with function 01h,07h,08h or 0Ah. Load the required function in AL
	MOV AH,0Ch
	MOV AL,function
	INT 21H
	This operation clears the keyboard buffer, execute the function in AL, and accepts(or waits for) a character, according to function request

INT 21H FUNCTION 0AH

- The input area for keyed in characters requires a parameter list containing specified fields that the INT operation is process.(like a structure)
  - PARLIST LABEL BYTE ; label is used for assigning a name to the parameter list
  - MAXLEN DB 25 ; maximum number of input characters
  - ACTULEN DB ? ;actual number of input characters
  - KBDATA DB 25 DUP(‘ ‘) ; character entered from keyboard

- MOV AH,0Ah
- LEA dx, PARLIST
- INT 21H
  - The INT operation waits for a user to type characters and checks that the numbers does not exceed maximum length including “Enter”.
  - If you enter 8 character like Jadavpur then

ASCII	25	08	J	a	d	a	v	p	u	r	#	....
	MAXLEN	ACTULEN										

  - It bypasses extended functions such as F1, home Page down, and arrows

```
TITLE     A08CTRNM (EXE)  Accept names from keyboard,
                          center them on screen, sound bell
          .MODEL  SMALL
          .STACK  64
          .DATA
PARLIST   LABEL   BYTE              ;Name parameter list:
MAXNLEN   DB      20                ;  maximum length of name
ACTULEN   DB      ?                 ;  no. of characters entered
KBNAME    DB      21 DUP(' ')       ;  entered name
PROMPT    DB      'Name? ', '$'
;-------------------------------------------------------
          .CODE
.386                                ;Directive for MOVZX
A10MAIN   PROC    FAR
          MOV     AX,@data          ;Initialize segment
          MOV     DS,AX             ;  registers
          MOV     ES,AX
          CALL    Q10CLEAR          ;Clear screen
A20:
          MOV     DX,0000           ;Set cursor to 00,00
          CALL    Q20CURSOR
          CALL    B10INPUT          ;Provide for input of name
          CALL    Q10CLEAR          ;Clear screen
          CMP     ACTULEN,00        ;Name entered?
          JE      A30               ;  no, exit
          CALL    C10CENTER         ;Set bell and '$' and center
          CALL    D10DISPLY         ;Display name
          JMP     A20               ;Repeat
A30:
          MOV     AX,4C00H          ;End processing
          INT     21H
A10MAIN   ENDP
;                   Display prompt and accept input of name:
;                    ---------------------------------------
B10INPUT  PROC    NEAR
          PUSH    AX                ;Preserve used
          PUSH    DX                ;  registers
          MOV     AH,09H            ;Request display
          LEA     DX,PROMPT         ;  of user prompt
          INT     21H
          MOV     AH,0AH            ;Request keyboard
          LEA     DX,PARLIST        ;  input
          INT     21H
          POP     DX                ;Restore
          POP     AX                ;  registers
          RET
B10INPUT  ENDP
;                    Set bell and '$' delimiter and
;                      set cursor at center of screen:
;                     ------------------------------
C10CENTER PROC    NEAR              ;Uses BX and DX
          MOVZX   BX,ACTULEN        ;Replace 0DH with 07H
          MOV     KBNAME[BX],07
          MOV     KBNAME[BX+1],'$'  ;Set display delimiter
          MOV     DL,ACTULEN        ;Locate center column:
          SHR     DL,1              ;  divide length by 2,
          NEG     DL                ;  reverse sign,
          ADD     DL,40             ;  add 40
          MOV     DH,12             ;Center row
          CALL    Q20CURSOR         ;Set cursor
          RET
C10CENTER ENDP
;                   Display centered name:
;                    ---------------------
D10DISPLY PROC    NEAR              ;Uses AH and DX
          MOV     AH,09H
          LEA     DX,KBNAME         ;Display name
          INT     21H
          RET
D10DISPLY ENDP
```

```
          .DATA
PARLIST   LABEL   BYTE
MAXNLEN   DB      20
ACTULEN   DB      ?
KBNAME    DB      21 DUP(' ')
PROMPT    DB      'Name? ', '$'
```

```
          MOV     AH,09H                    ;
          LEA     DX,PROMPT                 ;
          INT     21H
          MOV     AH,0AH                    ;
          LEA     DX,PARLIST                ;
          INT     21H
          POP     DX                        ;
          POP     AX                        ;
          RET
          ENDP
                    Set bell and '$' d
                      set cursor at ce
                     --------------------
          PROC    NEAR
          MOVZX   BX,ACTULEN
          MOV     KBNAME[BX],07
          MOV     KBNAME[BX+1],'$'
          MOV     DL,ACTULEN
          SHR     DL,1
          NEG     DL
```

```
;
;
D10DISPLY PROC    NEAR
          MOV     AH,09H
          LEA     DX,KBNAME
          INT     21H
          RET
D10DISPLY ENDP
```

INT 16H FUNCTIONS

- Provide low-level access to the keyboard, more so than MS-DOS.
- Input-output cannot be redirected at the command prompt.
- Function number is always in the AH register
- Important functions:
  - set typematic rate
  - push key into buffer
  - wait for key
  - check keyboard buffer
  - get keyboard flags

nibaran@cse.jdvu.ac.in

INT 16H FUNCTIONS

Number(in H)	functions

03	Set typematic rate; to change the typematic rate this function can be used
	MOV AH,03h
	MOV AL,05h
	MOV BH, repeat-delay ;0=1/4sec,1=1/2 sec
	;2=3/4sec and 3=1 sec
	MOV BL, repeat-rate ;(range 0-31; 0 fastest)
	INT 16h

05	Keyboard write in buffer this operation allows a program to insert characters in the keyboard buffer as if a user had pressed a key. Load the ASCII character into CH and its scan code into CL. The operation allows you to enter characters into the buffer until it is full. If full Carry flag and AL will set to 1

nibaran@cse.jdvu.ac.in

INT 16H FUNCTIONS

Number(in H)	functions
10	Read keyboard character This standard keyboard operation checks the keyboard buffer for an entered character. If none is present, it waits for the user to press a key.

Key pressed	AH	AL
Regular ASCII character	Scan code	ASCII character
Extended function key	Scan code	00h
Extended duplicate control character	Scan code	E0h

FUNCTION 10H: WAIT FOR KEY

- If a key is waiting in the buffer, the function returns it immediately. If no key is waiting, the program pauses (blocks), waiting for user input

```
.data
scanCode  BYTE ?
ASCIICode BYTE ?

.code
mov ah,10h
int 16h
mov scanCode,ah
mov ASCIICode,al
```

INT 16H FUNCTIONS

Number(in H)	functions
11	Determine if character present If an entered character is present in the Keyboard buffer, the operation clears the Zero Flag and delivers the character to AL and its scan code to AH; the entered character remains in buffer. If no character is present, the operation sets the zero flag and does not wait. it provides a look-ahead feature because the character remains in keyboard buffer until function 10h reads it.

nibaran@cse.jdvu.ac.in

INT 16H FUNCTIONS

12	Return keyboard shift status This operation delivers the keyboard status byte from BIOS data area1 at location 40:17h to AL and the byte from 40:18h to AH.

12	MOV AH,12h ;request shift status INT 16h ;call interrupt service. AND AL,00000011B ;left or right shift is pressed JZ exit ;yes

nibaran@cse.jdvu.ac.in