Hacettepe University
Computer Science and Engineering Department

BBM 342 Operating Systems Project 1

Advisors	Dr. Ahmet Burak Can, R.A. Kazım SARIKAYA
Subject	Operating Systems – Interrupt Handling
Development Environment	MS-DOS, Turbo C++ 3.0
Programming Language	Assembly, ANSI C
Submission Date	22.03.2013
Due Date	14.04.2013

INTRODUCTION

In this experiment you are going to learn how to use interrupt handling on Intel 80x86 architecture. You will implement a simple game, Pacman. One player versus to computer plays this game. The game area is considered as a labyrinth, which is given from an input file and output prefix from console while the program will be called (PACMAN.EXE <input> <output_prefix>). The input file consists of numbers contain information about the labyrinth of the game. The first line of the input file is dimensions of the labyrinth. The rest of the file contains the matrix that represents the labyrinth. A sample game input which contains two monsters and four diamonds is given below:

12x6
111111111111
000111000131
112003410000
111110111011
000300130002
111111111111

The number “1” denotes the walls, the number “0” denotes the paths, the number “2” denotes monsters, the number “3” denotes diamonds, and the number “4” denotes the pacman. At the start of the program, the input file is loaded and the game starts automatically in text screen mode. The pacman can be moved by using the arrow keys. The monsters continuously move in the labyrinth and their movement is random. However, they should be able to choose other (non-blocked) paths when their path is blocked by a wall. When pacman reaches an exit, the game ends after showing count of diamonds collected. If any monster touches to the pacman, the game should be over. Also the key “D” (not “d”) should dump the snapshot of the game (labyrinth, paths, monsters and pacman) into incrementally named files (<output_prefix>0.txt, <output_prefix>1.txt…) as in the input file format. Thus, a dump file can be used as an input file

(allows resuming the game). You do not use scrolling text output to the screen. You should draw the whole labyrinth into the same place.

The speed of the game should be 1 move/second that means every monster can move at 1 point in second. You should prepare a required timer for the speed.

The size of a labyrinth will not exceed the screen resolution for text mode (80x25). However the size of the labyrinth can change.

BACKGROUND INFORMATION

A partial hardware schema of the IBM-PC/AT, illustrating the interrupt mechanism with the keyboard and timer only, is shown in Figure-2. Every programmable register shown in this figure has its physical port address which is unique within the I/O address space of the PC. Programmable registers are also called I/O Ports. I/O Ports are normally accessed by means of in/out instructions of 80X86 processors.

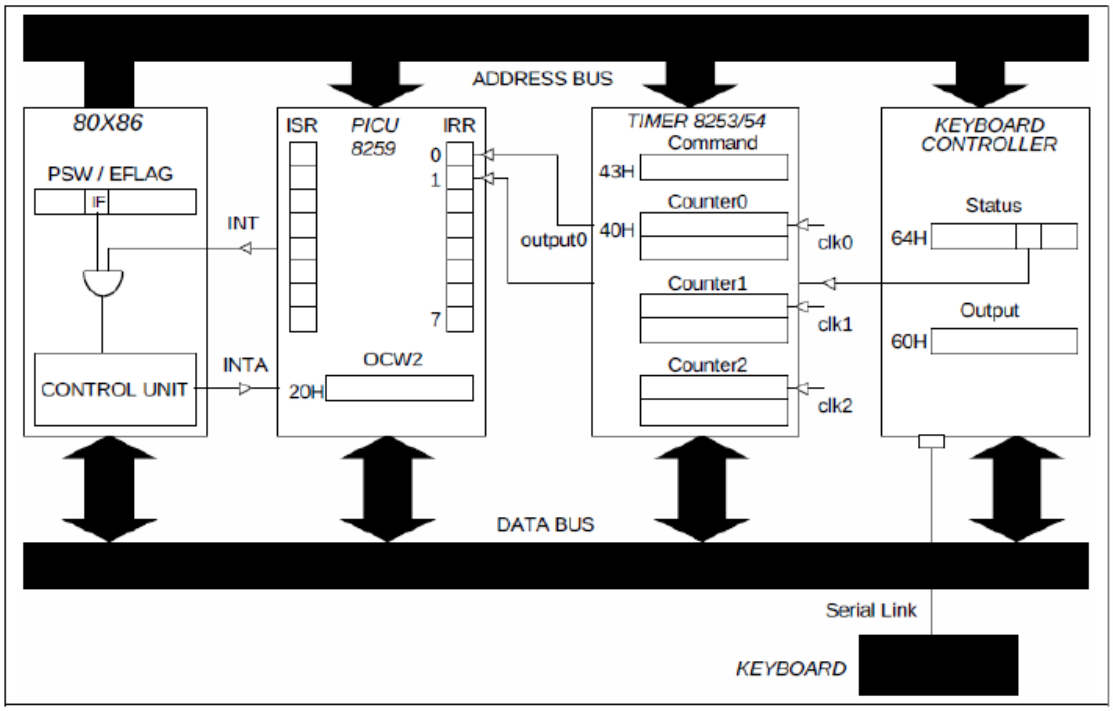


Figure-2. IBM-PC/AT compatible Microcomputer Hardware Schema

KEYBOARD

When a key is pressed on keyboard, an interrupt along with a scan code named "*make code*" is produced and when the key released a "*break code*" is produced by the keyboard controller. On a PC, keyboard is controlled by a chip and assigned to port numbers 60h and 61h. When a key is pressed on keyboard, scan value is put in register at 60h. You can get this scan code with the following command:

```
in al,60h
```

After getting the scan code, you have to reset the command register of the keyboard at 61h with following commands:

```
in al,61h
or al,82h
out 61h,al
and al,7fh
out 61h,al
```

At the end of every interrupt service routine, you clear programmable interrupt controller (PIC) service bit by sending End Of Interrupt (EOI) command, 20h to PIC port at address 20h.

```
mov al,20h
out 20h,al
```

THE TIMER

A timer is a device which generates periodic pulses with a programmed frequency. The IBM-PC/AT compatible PC's use the Intel 8254 chip as a timer. The Intel 8254 includes 3 independent 16 bit counters (Counter0, Counter1, Counter2) and 3 related clock inputs. Each counter is used to count down the clock pulses applied to its clock input. Each clock pulse decrements by one the content of the related counter. Every time a counter reaches zero a pulse is generated on its output and the initial count value is then reloaded into the counter. These type of counter operations allow the division of a clock frequency by a predetermined value. In the IBM-PC/AT compatible PC's the output0 of the 8254 is connected to the first input of the PICU in order to generate a real time clock.

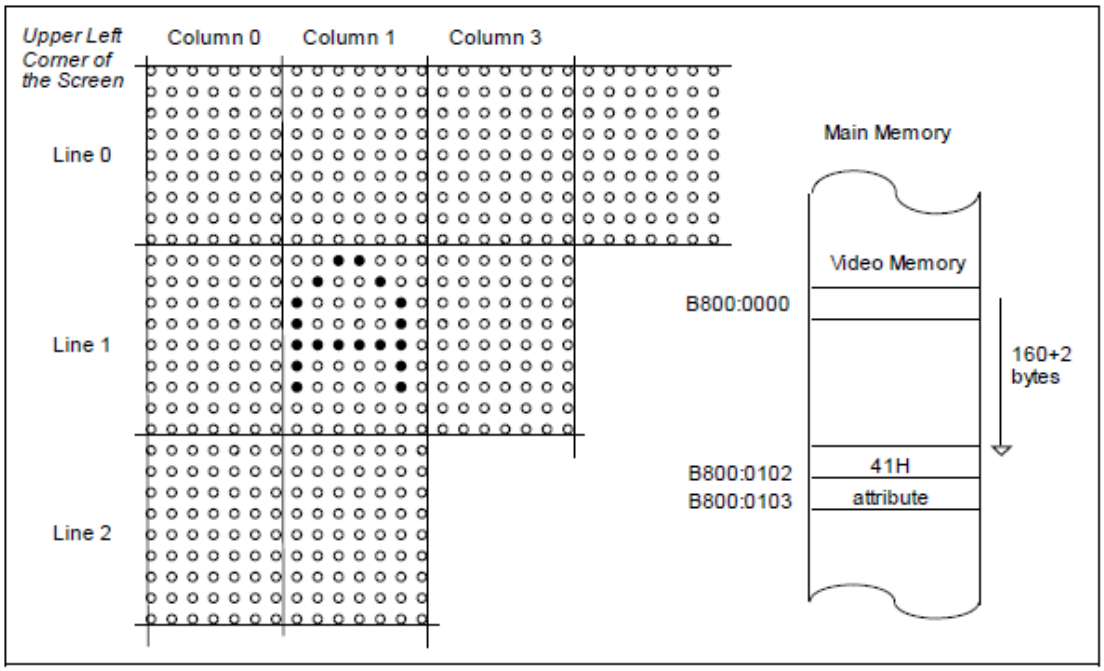
Besides the counters, the 8254 includes a 8-bit Command and a 8-bit Status registers as well. In this experiment your are only supposed to use the Counter0 and the Command register of the timer. The physical addresses of the Command register and the Counter0 are 43H and 40H respectively. Since the clock frequency applied to Counter0 is 1.193.180Hz, its initial value which will give the desired output frequency can be calculated as follows:

Counter initial value = 1,193,180 / *output frequency*

The 2 byte initial value computed by the above formula should be loaded to Counter0, the least significant byte first. 02H should be loaded into the Command register.

VIDEO MEMORY

With IBM-PC/AT compatible PC's the most currently used screen interface is VGA (Video Graphic Adapter). VGA can be programmed in 2 different modes: The Text and the Graphic mode. In the graphic mode the screen can be thought as a grid of dots; in the text mode as a grid of characters. In this experiment only the text mode will be considered. A typical video monitor has 25 rows and 80 columns of characters. Characters are displayed on the screen within an area of 7x9 dots (Figure-2).



VGA displays on the screen characters whose ASCII codes are placed at a special segment in the main memory. This special location is called the Video Memory. It starts at the B800:0000H physical address. Within this segment each character occupies 2 bytes: the first one is for the code and the second for its attribute which specifies, for instance, reverse video, blinking, high intensity and underlining. Character codes and attributes are located within the video memory in a linear fashion. The code and the attribute of a character displayed at line 3 and column 26 will be located at (3.80+26).2 starting from B800:0000H.

NOTES

- Do not use system calls related to input/output and time handling. Also do not use int 21h. (Develop your own assembly routines.)
- Make sure that you *backup and restore original interrupt vector and video memory buffer at the beginning and at the end* of your program.
- Your submission will be in the format below

  < StudentID>

  |-- report

  report.pdf

  |-- source

  GAME.PRJ

  GAME.C
- Submit your work as a Turbo C++ Project (with a .PRJ file).
- You have to use "Online Experiment Submission System". http://submit.cs.hacettepe.edu.tr. Other type of submissions especially by e-mail WILL NOT BE ACCEPTED.
- Downloaded or modified source codes from Internet resources will be considered as cheating.
- Do not use graphics.h or conio.h libraries.
- The development environment is available at ftp site near the experiment sheet.
- Using user friendly colors will be graded as bonus.