HACETTEPE UNIVERSITY
COMPUTER SCIENCE AND ENGINEERING DEPART-
MENT

Name and Surname:	Furkan Arslan
Identity Number:	21026686
Course:	BBM342- **Operating Systems**
Subject :	Operating Systems – Interrupt Handling
Experiment:	Experiment 1
Submission Date:	22.03.2013
Deadline:	18.04.2013
Teaching Assistants:	Dr. Ahmet Burak Can, R.A. Kazım SARIKAYA
E-Mail:	b21026686@cs.hacettepe.edu.tr

2. Software Using Documentation

2.1 Software Usage

The software is a kind of packman game. Like official packman, this packman includes one packman, some monsters and some diamonds. The packman is shown as number 4, monsters are shown as 2, diamonds are shown as 3 and 1 denotes walls. For separate them more clearly, all of them have different colors.

For starting game, two arguments must be given to the software. First one is input file which include game map size and game map. Game map shows start location of game characters. Second one is output prefix. The prefix define output file name. The output action will explain later. When the game started, the monsters move randomly in screen one second periodicity.

Main purpose of the game is reach to one of the exits before catching one of the monsters. For this purpose, user can control the packman with arrow keys. As usual, the up arrow key is moved packman one up block, the down key is moved one down, the left and right keys are moved it one left and right. The packman only can moves within empty paths. It cannot pass through walls. It can also eat diamonds for gain score.

When user presses the D key, whole matrix will be printed an output file which has increase number name. For example with first press, the output file name will be <output_prefix>0.txt, the second one has <output_prefix>1.txt .

2.2 Provided Possibility

For more enjoyable game, the game elements colors are different. Packman color is red, walls are purple, monsters are white and path is black.

2.3 ERROR MESSAGES:

There is no error message in this software.

3. Software Design Notes

3.1 Description Of The Program

3.1.1 Problem

In this example I must develop kind of packman game. The game should include packman rules. The packman can move within empty path and can eat diamonds. Furthermore the monsters move one block randomly every second. If packman reach to one of paths, user win the game otherwise one monster catches packman, user lost the game. The packman game must include all this rules.

Other problem is that detecting keyboard press. The software must redirection to keyboard port and must check whether predefined keys are pressed or not. The keys are d key and direction arrows which are up, down, left and right keys. The system firstly must detect after must make packman move with arrow keys press. When d key pressed, the system must dump whole matrix into output file.

For monster move, the system must use timer vector and must reprogram program interval timer (pit). After every one second, the monster must be moved randomly one block.

The last problem is that writing packman map to screen. For this problem, the software must use video memory. The map must be place into video memory and changes like packman move also must cause changing in the memory for showing the new packman location in the screen.

When the game ends, the redirection vectors must be restored back to original locations.

3.1.2 Solution

The keyboard detecting keyboard press problem is solved by using redirection to 9 interrupt vector. After this redirection whenever a key is pressed, the new interrupt function will be called. The function determines scan code firstly, then resets command register for become ready other interrupts. The scan code defines which key is pressed. The most significant bit of scan code determines whether keyboard is pressed or not. As for it is high, it means that the key is pressed then the function checks other bits. They give unique key values. The function checks up, down, left, right and d key values and makes their specific actions. After that the PIC reset. The old keyboard interrupt is restored after the game over.

Second solution is about programming timer. The monsters move every second. This happens with using timer interrupt and PIT. The PIT changes usual timer routine. The pit includes five modes and three counters. In this software we use mode two and counter zero. Mode two is standard dive by N counter. The output will be low for one period of the input clock then it will remain high for the time in the counter. The PIT also includes an 8 bit command register. For it help we can programmable to PIT. The command register least significant bit defines binary usage, after three bit (1-3) define PIT mode and after that two bit (4-5) defines write-read mode and most significant bit defines counter selection. Due to our decision we store 0x34 to command register. PIT output frequency determines by using a formula with correct arguments. After determined the frequency, it store in chosen counter. After all this process, we programmed timer. For understand one second pass, the timer interrupt vector which number is 8 must be redirection. The new interrupt function determines one second passes then make monster moves. The old timer interrupt is restored after the game over.

Other solution is about printing packman map. For this process, the video memory is used. After reading packman initial map, the map is stored in the memory. The reaching specific locations like packman location are calculated that using a formula ((80 x row + column) x2).

REFERENCES

http://physinfo.ulb.ac.be/cit_courseware/cprogram/advcw1.htm

http://www.osdever.net/bkerndev/Docs/pit.htm

http://kernelx.weebly.com/programmable-interval-timer.html

http://www.oocities.org/garyneal_71/GameLib/lowLevelKeyboardDriver.html

http://www.intel-assembler.it/portale/5/Programming-the-Intel-8253-8354-pit/howto-program-intel-pit-8253-8254.asp

http://vitaly_filatov.tripod.com/ng/tc/tc_000.244.html

http://ftp.sunet.se/pub/simtelnet/msdos/turbo_c/timertst.c