National University of Computer and Emerging Sciences

Laboratory Manual

for

Computer Organization and Assembly Language Programming

Lab Instructor	Sana Ejaz
Semester	Fall 2024

Department of Computer Science

FAST-NU, Lahore, Pakistan

OBJECTIVES:

- Understand the use of IN and OUT instructions for direct hardware communication.
- Learn to manipulate and handle Programmable Interrupt Controller (PIC) ports.
- Experiment with interrupt chaining and unhooking interrupts for custom handling.
- Explore the basics of the Programmable Interval Timer (PIT) and its integration with interrupts.
- Gain insight into terminating and staying resident (TSR) programs and their applications.

Task 1: Basic Hardware Communication with IN and OUT Instructions

Part 1: Read from a Port

1. Write an assembly program to read the status from the PIC control port.
2. Display the result of this read operation on the screen.

Instructions:

- Use `IN` instruction with the correct PIC port address.
- Use INT 21h (DOS interrupt) to display the value read from the port.

Part 2: Write to a Port

1. Write another program that configures the PIT to generate interrupts at a specific frequency.
2. Configure the PIT by writing to its control register and setting up the interval.

Instructions:

- Use `OUT` instruction to send data to the PIT control register.
- Choose a control word that sets a repeating interval, which can be displayed to observe the timer configuration

Task 2: Interrupt Chaining and Unhooking an Interrupt

Part 1: Chain a Custom Interrupt

1. Hook INT 08h (the timer interrupt) with a custom handler that increments a counter on each timer tick.
2. Chain this interrupt to retain the original INT 08h handler so that it executes after your custom code.

Instructions:

- Save the original interrupt vector for INT 08h.
- Replace INT 08h with your custom handler address, which increments a counter and then calls the original INT 08h handler. Use 0x3509 to get current vector for INT 08h and 0x2509 to set.

Part 2: Unhook the Custom Interrupt

1. Write code to unhook your custom handler and restore the original INT 08h vector.
2. Ensure that the program terminates safely and restores the interrupt environment to its original state.

Instructions:

- Use the DOS interrupt to replace the interrupt vector with the original handler.
- Verify that the original handler is properly restored by observing that the counter no longer increments after unhooking.

Task 3: Creating and Debugging a Terminate and Stay Resident (TSR) Program with INT 09h.

1. Set up a TSR that hooks the keyboard interrupt.
2. Add a breakpoint within the handler to facilitate debugging and analysis.
3. Track specific key presses and observe program behavior with the breakpoint interrupt (INT 3).

Instructions:

- Write a custom handler for INT 09h to monitor key presses. Use 0x3509 to get current vector for INT 09h and 0x2509 to set.
- Set the TSR to stay in memory and keep your handler active.
- Use INT 3 within the handler to halt execution on specific key presses (e.g., 'A' key) for debugging.
- Observe and document any changes in register values when execution halts.

Activity to try at home:

Try generating system sound using channel 2 (address port 0x42)