Christopher Mai
chrmai1
CS421_1
HW1P1

1. a)Discuss the advantages/disadvantages of using polling versus interrupts for managing I/O devices.

Polled I/O queries each channel or port in turn to determine if it has information for input or is ready to access data for output. Polling each port and device is time consuming. It easily wastes quite a few cpu cycles. However, polling can be used on fairly simple hardware.

Interrupt Driven I/O requires more complex hardware capable of interrupting the cpu. Interrupt driven I/O is also more efficient than polling when it comes to managing I/O devices as fewer cpu cycles are wasted.

b) Could the locations of interrupt hanlders be stored in a linked list instead of a fixed size array? Why or why not?

Could it? Yes, I thinks so. Is it recommended? No. Linked Lists rely on sequential access. Also, due to the nature of the linked lists more information must be stored to find the nodes that come after while an array has set positions for each node. Not to mention one of the weaknesses of arrays, resizing, is irrelevant as the size of interrupt handler array is determined by the number of interrupt pins on the processor. As it stands, arrays are faster and more efficient than a linked list for storing interrupt handlers.

c) How does a hardware I/O device send an interrupt to the CPU? How does the CPU handle the interrupt?

CPU hardware has a wire called the interrupt-request line that the cpu checks after each instruction. When an interrupt request is detected the cpu performs a state save and switches to the interrupt handler routine. It determines the cause of the interrupt, performs the necessary instructions and restores the previous state as to continue operation.

d) There can be interrupts that are lost, i.e. not handled properly by the CPU. Explain how could this happen? Why could it happen?

When an interrupt is made the programmable interrupt controller records in important information about that interrupt and queues it up for the cpu. Like all queues/stacks/storage spaces there is a limit to what can be stored. As such interrupts that do not fit in the queue may get lost. Interrupts may get lost if there are enough devices to flood the cpu with interrupt requests. Some architectures also limit 1 interrupt per device overwriting the older with newer.

e) Disabling interrupts at the CPU is a common method to avoid lost interrupts. Yet, there can be devices that want to send interrupts to the CPU, while interrupts are disabled. What happens to those interrupts? Are they lost? How are they handled?

Like all interrupts the PIC records information relevant to that interrupt and queues it up for the cpu. Interrupts that are sent while interrupts have been disabled at the cpu are recorded and queued. When interrupts are re-enabled the processor may choose to act on said interrupts. For the most part these interrupts are properly saved.