Chapter 9 Input / Output

1) Which of the following is not a requirement for a computer system to handle I/O in a sufficient and effective manner?

a) Peripheral devices must be individually addressable.

b) Peripheral devices must operate only in block mode.

c) Peripheral devices can initiate communication with the CPU.

d) Programmed I/O is suitable only for slow devices and individual word transfers.

2) An important difference between the I/O requirements of keyboards and disk drives is that

a) keyboard input is fast while disk drives are slow.

b) keyboards require constant monitoring, while disk drives do not.

c) disk drives have I/O controllers and keyboards do not have I/O controllers.

d) disk data is always transferred in blocks, never as individual bytes as with the keyboard.

3) From the perspective of a computer, the network

a) is just another I/O device.

b) requires an Ethernet connection.

c) is complex set of interconnected hosts.

d) is addressable only in blocks of 32 bit addresses.

4) The method used to communicate events that need special attention to the CPU are known as

a) interrupts.

b) I/O controllers.

c) programmed I/O.

d) device controllers.

5) The method of transferring data one word at a time from the CPU to a device is called

a) polling.

b) programmed I/O.

c) vectored interrupt.

d) direct memory access.

6) Computers provide interrupt capability by providing one or more special control lines to the central processor known as

a) fault lines.

b) address lines.

c) interrupt lines.

d) instruction lines.

7) The program that determines the appropriate course of action in the event an interrupt occurs is called the

a) fault handler.

b) device handler.

c) interrupt handler.

d) instruction handler.

8) When an interrupt causes temporary suspension of the program in progress, all the pertinent information about the program being suspended, including the location of the last instruction executed, and the values of data in various registers are stored in an area of memory known as the

a) register dump block.

b) memory dump block.

c) program method block.

d) process control block.

9) After interrupting a program in execution, and saving the program’s context, the computer then branches to a special program known as the

a) driver routine.

b) servicing program.

c) program service program.

d) interrupt handler program.

10)Since many interrupts exist to support I/O devices, most of the interrupt handling programs are also known as

a) device drivers.

b) device handlers.

c) peripheral handlers.

d) peripheral controllers.

11) The method of continuously checking the various input devices to determine if input data is waiting is called

a) polling.

b) observing.

c) monitoring.

d) supervising.

12) Which of the following is not a function of how interrupts are used?

a) A completion signal

b) An abnormal event indicator

c) A means of allocating CPU time

d) A way of buffering large amounts of data

13) Which of the following is an example of an interrupt being used as an external event notifier?

a) A keyboard input

b) A program inadvertently attempts to divide by zero

c) A time quantum has passed and the CPU is interrupted to start another task

d) An application program requests service from the operating system using a software interrupt

14) External events like keyboard input, mouse clicks, printer “out of paper” messages, and power failures are handled by

a) interrupts.

b) device handlers.

c) peripheral controllers.

d) suspension subprograms.

15) The computer system provides an internal clock that sends an interrupt periodically to the CPU signaling that it’s time to start processing another program or thread. The time between interrupt pulses is known as a(n)

a) delta.

b) quantum.

c) unit quantity.

d) atomic quantity.

16) Events related to problems or special conditions within the computer system itself, like divide by zero, or attempting to execute a nonexistent op code, are called

a) irregular events.

b) unusual events.

c) abnormal events.

d) anomalous events.

17) Internal interrupts caused by events related to problems or special conditions within the computer itself are sometimes called

a) exclusions.

b) exemptions.

c) special errors.

d) traps or exceptions.

18) Instructions that are intended for use by an operating system program, but not by an application program, are called

a) control instructions

b) limited instructions

c) prevalent instructions

d) privileged instructions

19) The software interrupt is very similar to which type of instruction?

a) STORE

b) LOAD

c) IF-THEN-ELSE

d) SUBROUTINE JUMP

20) The mnemonic for the x86 architecture instruction that simulates an interrupt is

a) SVC.

b) INT.

c) JMP.

d) GTO.

21) One way to assure that multiple programs do not unintentionally alter another program's files or intermingle printer output is to

a) only execute one interrupt at a time.

b) give programs in execution the highest priority.

c) not allow programs in execution to be interrupted.

d) require that all I/O to shared devices be handled by the operating system.

22) When the device generating the interrupt request identifies its address as part of the interrupt, it is called

a) polling interrupt.

b) discrete interrupt.

c) vectored interrupt.

d) monitoring interrupt.

23) Multiple interrupts can be handled by assigning \_\_\_\_\_\_\_ to each interrupt.

a) priorities

b) a time delay

c) queue levels

d) execution tickets

24) Power failures, internal time-sensitive events, or external events that are time sensitive will trigger interrupts that are

a) low priority events.

b) high priority events.

c) dependent on the device generating the interrupt.

d) measured for time-to-complete and scheduled accordingly.

25) Interrupts that can be temporarily disabled by program instructions are called

a) variable.

b) maskable.

c) changeable.

d) transferable.

26) Interrupts that can never be temporarily disabled by program instructions are called

a) invariable.

b) unchangeable.

c) nonmaskable.

d) non-transferable.

27) Interrupts are normally checked

a) during the operation of the instruction cycle.

b) immediately, without regard to the instruction cycle.

c) simultaneously with the beginning of an execution cycle.

d) after one instruction is finished and before another begins.

28) An I/O technique that transfers block data directly between the I/O controller and computer memory, is called

a) direct block access.

b) direct RAM access.

c) direct block transfers.

d) direct memory access.

29) Data from disks, and tapes, and flash memory are transferred only in

a) bits.

b) chunks of data.

c) blocks of data.

d) characters or bytes of data.

30) Which of the following is not one of the three primary conditions for direct memory access to take place?

a) The I/O device must have an internal buffer.

b) There must be a method to connect together the I/O interface and memory.

c) There must be a means to avoid conflict between the CPU and the I/O controller.

d) The I/O controller associated with the particular device must be capable of reading and writing to memory.

31) Four pieces of data must be provided to the I/O controller for a particular I/O device to initiate the DMA transfer. Which of the following is not required?

a) The size of the block to be transferred.

b) The location of the data on the I/O device.

c) The length of time required to transfer the data.

d) The starting location of the block of data in memory.

32) The incompatibilities in speed between the various devices and the CPU make I/O synchronization difficult, especially if there are multiple devices attempting to do I/O at the same time. To handle these problems data is usually stored

a) in a buffer.

b) on the network.

c) on the disk drive.

d) in external storage.

33) I/O controllers that control a single type of device are often called

a) device controllers.

b) precision controllers.

c) peripheral controllers.

d) single-type controllers.

34) An I/O controller that is designed to control hard disks is called a

a) disk handler.

b) disk controller.

c) disk coordinator.

d) disk interrupt handler.

35) Which of the following is not a common function of an I/O disk controller?

a) The I/O disk controller manages main memory during the transfer.

b) The I/O disk controller recognizes messages addressed to it and accepts commands from the CPU.

c) The I/O disk controller has interrupt capability, which it uses to notify the CPU when the transfer is complete.

d) The I/O disk controller provides a buffer where the data from memory can be held until it can be transferred to the disk.

Discussion questions

1) Why is it so important that the CPU be allowed to do other tasks while waiting for a particular I/O operation to be completed? (Two or three sentences should be adequate to answer this question.)

Sol: The CPU is operating at much higher speeds than I/O operations. If the CPU waited for I/O operations, this would be extremely inefficient.

2) What would happen if interrupts were checked and processed during the middle of an execution cycle instead of after the execution cycle has completed? (Two or three sentences should be adequate to answer this question)

Sol: From the text: "Interrupts are normally checked at the completion of each instruction. That is, interrupts are normally checked after one instruction is finished and before another begins. This assures that conditions won't change in the middle of an instruction that would affect the instruction's execution."

3) Once a DMA transfer has been initiated, why is it important that data being transferred not be modified during this period? (Two or three sentences should be adequate to answer this question)

Sol: From the text: "Once the DMA transfer has been initiated, the CPU is free to perform other processing. Note, however, that the data being transferred should not be modified during this period, since doing so can result in transfer errors, as well as processing errors."

4) Explain why it is better to have an I/O device initiate an interrupt to the CPU rather than the CPU monitor the I/O device. (Two or three sentences should be adequate to answer this question)

Sol: Monitoring I/O devices is considered polling, which is inefficient for the CPU. By having the I/O device initiate communication to the CPU through an interrupt, it frees the CPU to perform other tasks.

5) Suppose that the keyboard device stored keystrokes in a buffer until it was full then sent the entire buffer for processing. What would be some adverse consequences? (Two or three sentences should be adequate to answer this question)

Sol: Student responses could be more creative, but generally, if user input is expected by an application, then the application would stall waiting for the buffer to fill up. If the user wanted to interrupt the computer, as in typing "CONTROL-ALT-DEL," nothing would happen until the buffer was full.

6) Consider the steps required to write a block of data from a disk to memory. Outline the major sequence of I/O events that must occur to make this possible. (There are three major steps)

Sol: Taken from the example in the text.

1) Four pieces of data are sent to disk controller: 1) the location of the block in memory; 2) the location where the data is to be stored on disk; 3) the size of the block 4) and the direction of transfer: Write.

2) The I/O service program sends a “ready” message to the disk controller and the DMA transfer process takes place.

3) After the transfer is complete, the disk controller sends an interrupt to the CPU signaling completion, and the interrupt handler either returns control to the program that initiated the request or notifies the operating system that the program can be resumed.

7) Suppose an application program has initiated a DMA transfer from an I/O device, why does the I/O device controller interrupt the CPU when it's complete? (Two or three sentences should be adequate to answer this question)

Sol: In DMA transfers, the CPU is performing other tasks while the data transfer takes place; the CPU is unaware of when the data transfer has completed. The device controller has to notify the CPU when it's done so the interrupt handler can return control to the application that initiated the request.

8) Describe how the computer uses interrupts to control the flow of data to the printer.

Sol: From the text: "The computer sends one block of data at a time to the printer. The size of the block depends on the type of printer and the amount of memory installed in the printer. When the printer is ready to accept more data, it sends an interrupt to the computer. This interrupt indicates that the printer has completed printing the material previously received and is ready for more."

9) Discuss how writing a device driver for a hard disk would require an in-depth understanding of how the hardware functions. Would you also need an in-depth understanding of the OS platform for which the device was designed to function? (Recommend using over 100 words to adequately answer this question)

Sol: I/O controllers work at the CPU interface and the device interface. The I/O device controller must understand the specific OS instructions for CPU interfacing tasks: accepting I/O commands from the CPU, transferring data between the controller and the CPU or memory, and sending interrupts and status information to the CPU. At the device interface, the I/O controller must be able to control the device, such as moving the mechanical read/write head to the correct track in a disk drive, when to read or write data on the track, and maintaining rotational speed of the disk. Thus, writing a device driver would require an in-depth understanding of the OS and hardware.

10) How is an arithmetic overflow error or divide by zero error interrupt different than an interrupt from the hard disk controller signaling that a data transfer has completed? (Three or four sentences are sufficient)

Sol: Arithmetic overflow or divide by zero errors are abnormal events and it may not be possible to complete the program in execution. In that case, the OS attempts to gracefully recover from the error through traps or exceptions. A hard disk controller interrupting the CPU to signal that a data transfer has completed is used by the OS to resume control of the program requesting the data transfer and does not create the possibility that the system is halted.

11) A program in execution generates an interrupt with low priority. Before that interrupt is handled, it generates a high priority interrupt. Discuss how these two interrupts change the flow of execution from the start of the low priority interrupt through the completion of interrupt handling for this sequence of interrupts. (Three or four sentences are sufficient)

Sol: From the text: “This [multiple interrupt problem] leads to a hierarchy of interrupts, in which higher-priority interrupts can interrupt other interrupts of lower priority, back and forth, eventually returning control to the original program that was running.”

12) The system designer of a workstation computer wants to have enough interrupt codes (IRQs) to support 32 different interrupt lines, IRQ0 – IRQ31. How many bits are needed to address all the IRQs? Show your work and how you arrived at the solution.

Sol: Without any control or synchronous bits, 2 to the 5th power equal 32. Therefore 5 bits are needed.

13) Suppose that all programs in a particular CPU are given 50 clock cycles to process before getting swapped out for another program. Suppose also, that it takes 5 CPU clock cycles to swap out the process control block (PCB) for a particular program and restore the next program’s PCB.

a) What percent of the CPU clock cycles are used for processing 100 programs? (Hint: calculate: Program clock cycles / (Swap clock cycles + Program clock cycles)). Show your work and how you arrived at the solution.

Sol: Just to build the formula, look at 2 programs (n=2):

|<--------50cycles-------->|<--5cycles-->|<--------50cycles-------->|

|<--------Program------->|<--PCB----->|<-------- Program ----->|

Notice that the total number of clock cycles to swap out the PCB is (n-1)\*(5), where n is the number of programs.

Total number of Clock cycles to swap out the PCB = (n-1)\* 5 = 1\*5 = 5

Total number of Program clock cycles = n\*50 = 2\*50 = 100.

Percent of CPU clock cycles used for processing 2 programs = 100/(5 +100) = 100/105 or 95.2%

With the same reasoning, 100 programs will have

Total number Clock cycles to swap out the PCB = (n-1)\* 5 = 99\*5 = 495

Total number Program clock cycles = n\*50 = 100\*50 = 5000.

Percent of CPU clock cycles used for processing 100 programs = 5000/(495 +5000) = 91%

…………………………………………………………………………………………………………………

b) With the same conditions as above assume that each of the 100 programs has one and only one interrupt lasting 25 clock cycles. What percent of the CPU clock cycles are used for processing 100 programs?

To get started, here is an example for 2 programs and 2 interrupts.

EXAMPLE: 2 programs and 2 interrupts:

Program one gets interrupted at the 42nd clock cycle (arbitrarily chosen), and the interrupt gets serviced leaving 8 clock cycles to be completed (8 +42 = 50 cycles):

|<--------42cycles-------->|<--5cycles-->|<---25cycles---->|<--5cycles-->|<----8cycles-------->|

|<--------Prog1 start----->|<----swap-->|<---interrupt---->|<--swap----->|<--Prog1 complete->|

After swapping program two into the CPU, program two gets interrupted at the 12th cycle (arbitrarily chosen), and the interrupt gets serviced leaving 38 clock cycles (12 +38 = 50 cycles):

|<--5cycles-->|<---12cycles-->|<--5cycles-->|<---25cycles---->|<--5cycles-->|-------38cycles---->|

|<--swap--->|<---Prog2 start->|<----swap--->|<---interrupt---->|<--swap-->|<-Prog2 complete--->|

The total number of clock cycles used to swap out the PCB is 3n-1, where n is the number of programs.

Total number of Clock cycles to swap out the PCB = (3n-1)\* 5 = 5\*5 = 25

Total number of Program clock cycles = n\*50 = 2\*50 = 100.

Total number of Interrupt clock cycles = n\*25 = 2\*25 = 50.

Percent of CPU clock cycles used for processing 2 programs = 100/(25 +100+ 50) = 100/175 or 57.1%.

Now, calculate the percent of the CPU clock cycles are used for processing 100 programs were each program gets interrupted only once.

Sol:

Total number of Clock cycles to swap out the PCB = (3n-1)\* 5 = 299\*5 = 1495

Total number of Program clock cycles = n\*50 = 100\*50 = 5000.

Total number of Interrupt clock cycles = n\*25 = 100\*25 = 2500.

Percent of CPU clock cycles used for processing 2 programs = 5000/(1495 +5000+ 2500) = 5000/8995 or 55.5%.

14) If my CPU runs at 4.0GHz, and on average takes 10 clock cycles to complete an instruction, how many instructions will be completed in the time it takes to type "MY CPU IS RUNNING NOW"? Assume it takes 5 seconds to type the message. Show your work and how you arrived at the solution.

Sol: 4.0 G (cycles/sec) (1 instruction/10 cycles) (5 seconds) = 2,000,000,000 instructions

15) In terms of performance gains, what role does an I/O controller located inside the peripheral device perform? (50 to 100 words should be sufficient to answer this question)

Sol: From text: "In general I/O controllers simplify the task of interfacing peripheral devices to a CPU. I/O controllers offload a considerable amount of work from the CPU. They make it possible to control I/O to a peripheral with a few simple I/O commands from the CPU. They support DMA, so that the CPU may be free to perform other tasks. And, as we have already noted, device controllers provide the specialized circuitry required to interface different types of peripherals to the computer." So, in short, I/O controllers located inside the peripheral device allow the CPU to offload I/O tasks so the CPU can perform other tasks, increasing the performance of the system.

Solutions

|  |  |  |
| --- | --- | --- |
| Problem | Answer | Section in text / comments |
| 1 | b | 9.1 Characteristics of Typical I/O Devices |
| 2 | d | 9.1 Characteristics of Typical I/O Devices |
| 3 | a | 9.1 Characteristics of Typical I/O Devices |
| 4 | a | 9.0 Introduction |
| 5 | b | 9.2 Programmed I/O |
| 6 | c | 9.3 Interrupts |
| 7 | c | 9.3 Interrupts |
| 8 | d | 9.3 Interrupts |
| 9 | d | 9.3 Interrupts |
| 10 | a | 9.3 Interrupts |
| 11 | a | 9.3 Interrupts |
| 12 | d | 9.3 Interrupts |
| 13 | a | 9.3 Interrupts |
| 14 | a | 9.3 Interrupts |
| 15 | b | 9.3 Interrupts |
| 16 | c | 9.3 Interrupts |
| 17 | d | 9.3 Interrupts |
| 18 | d | 9.3 Interrupts |
| 19 | d | 9.3 Interrupts |
| 20 | b | 9.3 Interrupts |
| 21 | d | 9.3 Interrupts |
| 22 | c | 9.3 Interrupts |
| 23 | a | 9.3 Interrupts |
| 24 | b | 9.3 Interrupts |
| 25 | b | 9.3 Interrupts |
| 26 | c | 9.3 Interrupts |
| 27 | d | 9.3 Interrupts |
| 28 | d | 9.4 Direct Memory Access |
| 29 | c | 9.4 Direct Memory Access |
| 30 | a | 9.4 Direct Memory Access |
| 31 | c | 9.4 Direct Memory Access |
| 32 | a | 9.4 Direct Memory Access |
| 33 | a | 9.4 Direct Memory Access |
| 34 | b | 9.4 Direct Memory Access |
| 35 | a | 9.4 Direct Memory Access |