

[Show All Answers](#)[Hide All Answers](#)

Interrupts and real time

Problem 1. A computer system has three devices whose characteristics are summarized in the following table:

Device	Service Time	Interrupt Frequency	Allowable Latency
D1	400 us	1/(800us)	400us
D2	250 us	1/(1000us)	50us
D3	100 us	1/(800us)	300us

Service time indicates how long it takes to run the interrupt handler for each device. The maximum time allowed to elapse between an interrupt request and the start of the interrupt handler is indicated by allowable latency.

- A. If a program P takes 100 seconds to execute when interrupts are disabled, how long will P take to run when interrupts are enabled?

[Hide Answer](#)

We need to compute how much CPU time is spent servicing interrupts = sum [$\langle \text{service time} \rangle * \langle \text{interrupt frequency} \rangle$]

$$D1: 400\text{us} * (1/800\text{us}) = 50\% \text{ of cpu time}$$

$$D2: 250\text{us} * (1/1000\text{us}) = 25\% \text{ of cpu time}$$

$$D3: 100\text{us} * (1/800\text{us}) = 12.5\% \text{ of cpu time}$$

87.5% of the CPU time is spent servicing interrupts, leaving 12.5% of the CPU to run P, so P will take approximately $(1/.125)*100 = 800$ seconds to run. "Approximately" because the actual answer depends on when exactly P starts running with respect to the sequence of interrupts.

- B. Can the requirements given in the table above be met using a weak priority ordering among the interrupt requests?

[Hide Answer](#)

No. Once D1 or D3 start running, we will miss allowable latency for D2.

- C. Can the requirements given in the table above be met using a strong priority ordering among the interrupt requests?

[Hide Answer](#)

Yes, if priority is $D2 > D3 > D1$.

Problem 2. Surreal Time Systems is configuring a Beta for a dedicated application involving three I/O devices whose characteristics are summarized below:

Device	Interrupt Frequency	Service Time
A	1/(1000 us)	600 us
B	1/(500 us)	100 us
C	1/(1000000 us)	100000 us

Each of the three devices causes periodic interrupts as the given frequency. Each interrupt requires the service time specified for that device. When the processor is not servicing any interrupt, it runs a compute-bound user-mode (background) task L which composes limericks like

*There was an old lady from Crewe
whose limericks stopped at line two.*

For each of the following questions, assume all devices are requesting service at their maximum rate.

- A. Assuming no interrupt priorities, what is the approximate worst-case latency seen by each device?

Hide Answer

for A: max latency = sum service times for B and C = 100100 us
for B: max latency = sum service times for A and C = 100600 us
for C: max latency = sum service times for A and B = 700 us

- B. Now assume that each interrupt handler must complete execution before the next request from the same device in order to avoid losing data. To accommodate this real time constraint, the processor is enhanced with a 4-level strong priority system with priorities 0 (background), 1, 2 and 3 (highest). What priorities would you assign to each device?

Hide Answer

B has to have the priority 3 because its handler must complete execution in 500us and the handlers for both A and C, if allowed to run, would prevent that from happening since both run for more than 500us.

Similarly, A has to have priority 2, leaving C to run at priority 1.

- C. Suppose that, in the absence of interrupts, L composes an average of 100 limericks per hour. What is its rate when all three devices are interrupting?

Hide Answer

We need to compute how much CPU time is spent servicing interrupts:

- A: $600\text{us} * (1/1000\text{ us}) = 60\%$ of cpu time
 B: $100\text{us} * (1/2000\text{ us}) = 20\%$ of cpu time
 C: $100000\text{us} * (1/1000000\text{ us}) = 10\%$ of cpu time

leaving 10% of the CPU to run background tasks. So L would compose approximately 10 limericks per hour.

Problem 3. A computer system is interfaced to four devices: a printer, a disk, a keyboard, and a display. The characteristics of the devices are summarized in the following table.

Device	Interrupt service time	Interrupt frequency	response-time requirement
Printer	1000 us	$1/(2000\text{ us})$	1000 us
Disk	300 us	$1/(1000\text{ us})$	200 us
Keyboard	2000 us	$1/(100000\text{ us})$	2000 us
Display	100 us	$1/(1000\text{ us})$	200 us

- A. A program P, which performs only computation (no input/output), takes 100 s to run when no input/output is being performed. How long will it take for P to run when all of the above devices are operating at their maximum speeds?

Hide Answer

We need to compute how much CPU time is spent servicing interrupts:

- Printer: $1000\text{us} * (1/2000\text{ us}) = 50\%$ of cpu time
 Disk: $300\text{us} * (1/1000\text{ us}) = 30\%$ of cpu time
 Keyboard: $2000\text{us} * (1/100000\text{ us}) = 2\%$ of cpu time
 Display: $100\text{us} * (1/1000\text{ us}) = 10\%$ of cpu time

leaving 8% of the CPU to run background tasks. So P would take approximately $(100/.08) = 1250$ seconds to run.

- B. Suppose that the interrupt system enforces a nonpreemptive (weak) priority ordering printer > disk > keyboard > display among interrupt requests. Assuming the characteristics given in the table above, what is the maximum time that might elapse between a disk interrupt request and execution of the first instruction of its handler? Assume that the time taken for state save and context switch is negligible.

Hide Answer

The latency seen by the disk with a weak priority ordering is the sum of the maximum service time for any device (keyboard = 2000 us) plus the service times for higher priority devices (printer = 1000 us). So the latency is 3000 us.

- C. Can the requirements given in the table above be met using a *weak* priority ordering among the interrupt requests? If so, give such an ordering; if not, explain.

Hide Answer

No, the interrupt service time for the keyboard (2000 us) will prevent the system from meeting any of the other response time requirements (all less than 2000 us).

- D. Can the requirements given in the table above be met using a *strong* priority ordering among the interrupt requests? If so, give such an ordering; if not, explain.

Hide Answer

Yes, with the ordering Display > Disk > Printer > Keyboard. A detailed worst-case timing diagram is shown below. The worst-case is when all 4 interrupts happen simultaneously and then subsequent interrupts happen their maximum frequency. Since this is a strong priority system, the handlers for lower priority devices are interrupted to service higher priority interrupts.

```

time  interrupt           running
   0  p,disk,k,dpy       Display (100)
  100                               Disk (300)
  200                               "
  300                               "
  400                               Printer (1000)
  500                               "
  600                               "
  700                               "
  800                               "
  900                               "
 1000 disk,dpy           Display (100); interrupts printer
 1100                               Disk (300)
 1200                               "
 1300                               "
 1400                               Printer (resume with 400 left)
 1500                               "
 1600                               "
 1700                               "
 1800                               Keyboard (2000)
 1900                               "
 2000 p,disk,dpy       Display (100); interrupts keyboard
...

```

At time 2000 the cycle starts over again with the exception of the keyboard interrupt which will happen next at time 100,000. The keyboard handlers runs for 200us in every 2000us cycle and so will complete its task by time 20000.

Problem 4. A computer must service three devices whose interrupting frequencies, service times, and assigned priorities are given in the table below.

Device	Service time (ms)	Maximum Frequency (1/ms)	Priority
D1	10	1/100	3 (highest)
D2	50	1/1000	2
D3	200	1/5000	1 (lowest)

- A.
 B. Assuming a *strong* priority system, compute for each device the maximum time between service request and the *completion* of service for that device.

Hide Answer

A detailed worst-case timing diagram is shown below. The worst-case is when all 4 interrupts happen simultaneously and then subsequent interrupts happen their maximum frequency. Since this is a strong priority system, the handlers for lower priority devices are interrupted to service higher priority interrupts.

time	interrupt	running
0	d1,d2,d3	D1 (10)
10		D2 (50)
20		"
30		"
40		"
50		"
60		D3 (200)
70		"
80		"
90		"
100	d1	D1 (10); interrupting D3
110		D3 (resume with 160 left)
120		"
130		"
140		"
150		"
160		"
170		"
180		"
190		"
200	d1	D1 (10); interrupting D3
210		D3 (resume with 70 left)
220		"
230		"
240		"
250		"
260		"
270		"
280		idle!!!
290		...

From the diagram we can see that the maximum time to completion is 10ms for D1, 60ms for D2 and 280ms for D3. Note that the D1 interrupt occurs several times during the handler for D3 which complicates the calculation and is why it is usually best to draw out the diagram shown above.

- B.
 C. What percentage of the processor's time is devoted to servicing D1?

Hide Answer

$(10\text{ms service time}) * (1/100\text{ms interrupt frequency}) = 10\%$ of the CPU time.

- C. What percentage of the processor's time is left for noninterrupt programs?

Hide Answer

We need to compute how much CPU time is spent servicing interrupts:

$$D1: 100\text{ms} * (1/100 \text{ ms}) = 10\% \text{ of cpu time}$$

$$D2: 50\text{ms} * (1/1000 \text{ ms}) = 5\% \text{ of cpu time}$$

$$D3: 200\text{ms} * (1/5000 \text{ ms}) = 4\% \text{ of cpu time}$$

leaving 81% of the CPU to run background tasks.

- D. Assume that if a device interrupts again before a pending interrupt on that same device has been serviced, the later interrupt is ignored (lost). Will the system outlined in the table above lose interrupts using a *strong* priority scheme (with priorities as given)?

Hide Answer

No. See the worst-case timing diagram in the answer for part (A).

- E. Under the assumption of question (4), will the system outlined in the table above lose interrupts using a *weak* priority scheme (with priorities as given)?

Hide Answer

Yes. D3 will prevent D1 from being serviced in time.

Problem 5. Consider the following priority-interrupt scenario:

Task	Service time (ms)	Maximum allowed latency (ms)	Maximum Frequency (1/ms)
A	30	500	1/3000
B	20	70	1/1000
C	50	25	1/500
D	10	10	1/50

- A. Can you use a weak priority scheme for the scenario outlined in the table above? Explain.

Hide Answer

No. A prevents C from being serviced in time.

- B. Assume that all the interrupts listed in the table above occur at their maximum frequency. What percent of the processor's time is used to handle interrupts?

Hide Answer

$$A: 30\text{ms} * (1/3000 \text{ ms}) = 10\% \text{ of cpu time}$$

$$B: 20\text{ms} * (1/1000 \text{ ms}) = 5\% \text{ of cpu time}$$

$$C: 50\text{ms} * (1/500 \text{ ms}) = 10\% \text{ of cpu time}$$

D: $10\text{ms} * (1/50 \text{ ms}) = 20\%$ of cpu time
 Total = 45%

- C. Assume a strong priority system in which 3 is the highest priority, 0 the lowest. Assign a unique priority to each task in the table above to meet the specifications given. Show the maximum time between interrupt and *completion* of service for each of the tasks if your priority scheme is used.

Hide Answer

A detailed worst-case timing diagram is shown below. The worst-case is when all 4 interrupts happen simultaneously and then subsequent interrupts happen their maximum frequency. Since this is a strong priority system, the handlers for lower priority devices are interrupted to service higher priority interrupts.

time	interrupt	running
0	A,B,C,D	D (10)
10		C (50)
20		"
30		"
40		"
50	D	D (10); interrupting C
60		C (resume with 10 left)
70		B (20)
80		"
90		A (30)
100	D	D (10); interrupting A
110		A (resume with 20 left)
120		"
130		idle...
...		

Looking at the table, the maximum time between interrupt and completion is:

- A: 130 ms
- B: 90 ms
- C: 70 ms
- D: 10ms