Computation Structures
Interrupts and Real Time Worksheet

Latency (L) = elapsed time before handler code starts to execute
Service time (S) = time required to execute handler code
Deadline (D) = maximum time after request by when handler execution must be complete
Maximum allowable latency (Lmax) = largest L such that Lmax + S = D.

Handler priority: when choosing which handler to run, choose the handler with the highest priority. Priorities are chosen to ensure that the deadlines for all handlers will be met.

Recurring requests: often requests will occur at some fixed interval ≥ deadline.

Earliest deadline is a strategy for assigning priorities that is guaranteed to meet the deadlines if any priority assignment can meet the deadlines:
1. Sort the requests by their deadlines
2. Assign the highest priority to the earliest deadline, second priority to the next deadline, and so on.

Weak (non-preemptive) priorities: after current handler completes, look through the pending requests and execute the handler with the highest priority. Latencies with weak priorities: service of each device might be delayed by (1) service of one other (arbitrary) device whose request was just honored, and (2) service of all higher-priority tasks.

Strong (preemptive) priorities: always run the pending handler with the highest priority, possibly interrupting the execution of a lower-priority handler to do so. Latencies with weak priorities: service of each device might be delayed by service of all (possibly recurring) higher-priority tasks.

Recurring Interrupts
Consider interrupts which recur at bounded rates:

<table>
<thead>
<tr>
<th>Priority</th>
<th>Latency using string priority</th>
<th>Device</th>
<th>Service time (S)</th>
<th>Deadline (D)</th>
<th>Lmax</th>
<th>Max freq</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>900μs</td>
<td>Keyboard</td>
<td>800μs</td>
<td>100μs</td>
<td>100μs</td>
<td>100μs</td>
</tr>
<tr>
<td>2</td>
<td>0</td>
<td>Disk</td>
<td>500μs</td>
<td>800μs</td>
<td>500μs</td>
<td>500μs</td>
</tr>
<tr>
<td>3</td>
<td>500μs</td>
<td>Printer</td>
<td>400μs</td>
<td>1000μs</td>
<td>1000μs</td>
<td>1000μs</td>
</tr>
</tbody>
</table>

Note that interrupt LATTENCIES don't tell the whole story—consider COMPLETION TIMES, e.g., for Keyboard in the example above.

Interrupt Load
How much CPU time is consumed by interrupt service?

<table>
<thead>
<tr>
<th>Priority</th>
<th>Device</th>
<th>Service time (S)</th>
<th>Deadline (D)</th>
<th>Lmax</th>
<th>Max freq</th>
<th>% Load</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>900μs</td>
<td>Keyboard</td>
<td>800μs</td>
<td>100μs</td>
<td>800μs*1/100μs = 8%</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>0</td>
<td>Disk</td>
<td>500μs</td>
<td>300μs</td>
<td>500μs*500μs = 25%</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>500μs</td>
<td>Printer</td>
<td>400μs</td>
<td>1000μs</td>
<td>400μs*1/1000μs = 40%</td>
<td></td>
</tr>
</tbody>
</table>

User-mode share of CPU = 1 - ∑(Smax*max_freq/τ) = 0.27
Also check to see if enough CPU time to meet all deadlines

6.004 Worksheet

Interrupts and Real Time

- 1 of 7 -
Problem 1.

Ben Bitdiddle has designed a wrist device called the BenBit to measure how long you’ve been tooling away without getting up and moving around. The BenBit runs a real-time operating system supporting three tasks whose handlers are executed in response to periodic requests. Each task has a period (time between requests), a service time (time it takes to run its handler), and a deadline (maximum time allowed to elapse between request and completion of the handler).

<table>
<thead>
<tr>
<th>Task</th>
<th>Period (ms)</th>
<th>Service time (ms)</th>
<th>Deadline (ms)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Check Accelerometer (CA)</td>
<td>80</td>
<td>20</td>
<td>40</td>
</tr>
<tr>
<td>Update Display (UD)</td>
<td>200</td>
<td>?</td>
<td>200</td>
</tr>
<tr>
<td>Determine heart rate (DHR)</td>
<td>60</td>
<td>10</td>
<td>50</td>
</tr>
</tbody>
</table>

Ben is trying to figure out whether to use a weak or strong priority system to manage task execution. For each of the questions below, please fill the answers for both types of priority system. For both the weak and strong priority system assume the task priority is CA > DHR > UD, i.e., CA has the highest priority and UD the lowest. If a calculation requires the service time for UD, use your answer for part (A).

<table>
<thead>
<tr>
<th>Question</th>
<th>Using Weak Priorities</th>
<th>Using Strong Priorities</th>
</tr>
</thead>
<tbody>
<tr>
<td>(A) What is the maximum service time for UD handler that still allows all deadlines to be met (in ms)?</td>
<td>20 ms [otherwise CA will miss deadline]</td>
<td>100 ms [= 200 - 3 \times 20 - 4 \times 10]</td>
</tr>
<tr>
<td>(B). What fraction of the time will the processor spend idle? (%)</td>
<td>[48.33%] = 1 - \left(\frac{20}{20}\right) - \left(\frac{20}{60}\right) = \left(\frac{10}{60}\right)]</td>
<td>[8.33%] = 1 - \left(\frac{20}{60}\right) - \left(\frac{100}{200}\right) - \left(\frac{10}{60}\right)]</td>
</tr>
<tr>
<td>(C) What is the worst-case completion time for the CA task (in ms)?</td>
<td>40 ms [= UD_{st} + CA_{st}]</td>
<td>20 ms [= CA_{st}]</td>
</tr>
<tr>
<td>(D) What is the worst-case completion time for the UD task (in ms)?</td>
<td>50 ms [= DHR_{st} + CA_{st} + UD_{st}]</td>
<td>200 ms [= 3 \times CA_{st} + 4 \times DHR_{st} + UD_{st}]</td>
</tr>
<tr>
<td>(E) What is the worst-case completion time for the DHR task (in ms)?</td>
<td>50 ms [= UD_{st} + CA_{st} + DHR_{st}]</td>
<td>30 ms [= CA_{st} + DHR_{st}]</td>
</tr>
</tbody>
</table>
Problem 2.

A particular real-time system has three interrupt handlers. The following table shows the maximum rate at which each interrupt occurs (rate), the time taken to execute each handler (service time), and the maximum allowable interval between the interrupt and completion of the handler (deadline). In your analysis, assume that A, B, and C interrupts can arrive at any time.

<table>
<thead>
<tr>
<th>Task</th>
<th>Rate</th>
<th>Service Time</th>
<th>Deadline</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>1/20ms</td>
<td>10ms</td>
<td>20ms</td>
</tr>
<tr>
<td>B</td>
<td>1/80ms</td>
<td>10ms</td>
<td>80ms</td>
</tr>
<tr>
<td>C</td>
<td>1/25ms</td>
<td>5ms</td>
<td>25ms</td>
</tr>
</tbody>
</table>

(A) What is the percentage idle time for this system?

\[ \text{Percentage Idle Time (\%)} = \frac{100\% - \left( 50 + 12.5 + 20 \right)}{50} \]

\[ \text{Percentage Idle Time (\%)} = 17.5\% \]

(B) Assuming a weak priority system. Can the priority ordering C > A > B satisfy all the constraints of the system?

\[ \text{C > A > B (Yes/No): No} \]

(C) Assuming a strong priority system with priority C > A > B, what is the worst-case delay between the task’s interrupt and completion of the task’s interrupt handler?

Worst-case delay for task A (ms): 15

Worst-case delay for task B (ms): 40

Worst-case delay for task C (ms): 5
Problem 3.

A particular real-time system has three interrupt handlers. The following table shows the maximum rate at which each interrupt occurs (rate), the time taken to execute each handler (service time), and the maximum allowable interval between the interrupt and completion of the handler (deadline). In your analysis, assume that A, B, and C interrupts can arrive at any time.

<table>
<thead>
<tr>
<th>Task</th>
<th>Rate</th>
<th>Service time</th>
<th>Deadline</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>1/40ms</td>
<td>5ms</td>
<td>30ms</td>
</tr>
<tr>
<td>B</td>
<td>1/100ms</td>
<td>?ms</td>
<td>100ms</td>
</tr>
<tr>
<td>C</td>
<td>1/50ms</td>
<td>10ms</td>
<td>25ms</td>
</tr>
</tbody>
</table>

Please answer the following two questions assuming the system has a weak priority system.

(A) (1 point) What is the maximum service time for task B that still allows all constraints to be met?

\[ \text{Maximum service time for task B (ms)} = \begin{cases} 15 & \text{if } A > B \text{ in port}(B) \\ 10 & \text{if } A > C \text{ in port}(B) \end{cases} \]

Task C has max latency of 15 ms \( \Rightarrow \) limits service time of B.

(B) (1 point) Assuming a suitable service time for B, give a weak priority order for the tasks that meets the constraints. List the task with the highest priority first, the task with the lowest priority last.

\[ \text{Weak priority order for the tasks: } C > A > B \]

Please answer the following two questions assuming the system has a strong priority system where task C has the highest priority and task B has the lowest priority.

(C) (2 points) What is the maximum service time for task B that still allows all constraints to be met?

\[ \text{Maximum service time for task B (ms)} = 65 \]

In 100ms worst case:
3 task A requests (15ms total)
2 task C requests (20ms total)
\[ \frac{35 \text{ms service time}}{35} \Rightarrow 100 - 35 = 65 \text{ms available for B} \]

(D) (3 points) Assume B’s service time is 10ms. For each task, what is the worst-case delay between the task’s interrupt and completion of the task’s interrupt handler?

\[ \text{Worst-case delay for task A (ms)} = 15 \]

\[ \text{Worst-case delay for task B (ms)} = 25 \]

\[ \text{Worst-case delay for task C (ms)} = 10 \]

\[ \text{starts immediately} \]

\[ = C \text{ service time} \]
Problem 4.

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

<table>
<thead>
<tr>
<th>Device</th>
<th>Service time</th>
<th>Interrupt Frequency</th>
<th>Deadline</th>
</tr>
</thead>
<tbody>
<tr>
<td>D1</td>
<td>400us</td>
<td>1/(800us)</td>
<td>800us</td>
</tr>
<tr>
<td>D2</td>
<td>250us</td>
<td>1/(1000us)</td>
<td>300us</td>
</tr>
<tr>
<td>D3</td>
<td>100us</td>
<td>1/(800us)</td>
<td>400us</td>
</tr>
</tbody>
</table>

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 end of the execution of the interrupt handler is indicated by the deadline.

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

   **Approximate time for P to run with interrupts enabled (seconds):** \( \frac{8 \cdot 100}{800} = 800 \) seconds

   \( D1: \frac{400}{800} = 50\% \) of CPU time
   \( D2: \frac{250}{1000} = 25\% \) of CPU time available for running P
   \( D3: \frac{100}{800} = 12.5\% \)

   Only 12.5\% = 100us of CPU time available for running P.

B. Can the requirements given in the table above be met using a weak priority ordering among the interrupt requests? If so give priority ordering for D1, D2, D3 or list device(s) whose deadlines cannot be met.

   **Weak priority ordering or list device(s) with missed deadlines:** D2, D3

   If D1 is running, D2 and D3 may miss deadlines since D2 max latency is 500us and D3 max latency is 300us.

(C) Can the requirements given in the table above be met using a strong priority ordering among the interrupt requests? If so give priority ordering for D1, D2, D3 or list device(s) whose deadlines cannot be met.

   **Strong priority ordering or list device(s) with missed deadlines:** D2 > D3 > D1
   
   Use earliest deadline strategy.
Problem 5.

A real-time operating system with priority interrupts has three interrupt handlers – A, B, C – each running at a different priority level. The handlers are invoked by the A, B, and C interrupts, marked as ↑ in the execution timelines. For example, the following execution timeline shows the A handler running to completion after an A interrupt request, followed by execution of the B handler, which is interrupted by execution of the C handler.

For each of the following execution timelines, please indicate whether the system is using a WEAK or STRONG priority scheme, or CAN’T TELL if the timeline is consistent with either WEAK or STRONG. Also, if WEAK or STRONG, indicate any relative priorities that can be deduced from the timeline (there may be more than one), expressed as inequalities. For example, A > B indicates A has a higher priority than B.

(A)

Circle one: STRONG ... WEAK ... CAN'T TELL, Priorities: A > B, C > B

(B)

Circle one: STRONG ... WEAK ... CAN'T TELL, Priorities: C > A

(C)

Circle one: STRONG ... WEAK ... CAN'T TELL, Priorities: __________________________

(D)

Circle one: STRONG ... WEAK ... CAN'T TELL, Priorities: C > B > A
Problem 6.

A real-time operating system with priority interrupt has three interrupt handlers (A, B, C), each of which, when invoked by the appropriate interrupt request (marked as ↑ in the execution timelines), takes 11 time units to execute. For example, the following execution timeline shows the A handler running to completion after an A interrupt request, followed by execution of the B handler, which is itself interrupted by execution of the C handler.

(A) The execution timeline below shows the arrival times of interrupt requests for A, B, and C. Diagram the execution of the A, B, and C handlers assuming a weak priority system with the priorities C > B > A. Remember to show the complete execution (all 11 time units) for each handler, labeling each block with the handler that is running.

(B) The execution timeline below shows the arrival times of interrupt requests for A, B, and C. Diagram the execution of the A, B, and C handlers assuming a strong priority system with the priorities C > B > A. Remember to show the complete execution (all 11 time units) for each handler, labeling each block with the handler that is running.