Problem 1 Petri Nets [15 pts]
Assume you have a Petri Net $N$, with $P$ places, $T$ transitions and $F$ flows:

$P = \{P_0, P_1, P_2, P_3, P_4\}$
$T = \{T_0, T_1, T_2, T_3, T_4\}$
$F = \{(P_0, T_0), (P_0, T_1), (P_1, T_2), (T_2, P_1), (T_1, P_2), (P_2, T_3), (T_3, P_3), (T_0, P_4), (T_2, P_3), (P_4, T_4), (T_3, P_4)\}$

Initial marking $M_0 = [1, 2, 0, 0, 0]$

a) Draw the specified Petri Net.

b) Draw the reachability tree. Are states $[0,1,1,1,0]$ and $[1,1,0,0,0]$ reachable?

c) Is $N$ pure?

d) What degree of liveness does $N$ have?

c) Yes

d) L1
Problem 2 SDL [5 pts]
Convert the following FSM into an equivalent SDL representation.

Solution:
Problem 3 Finite State Machines and StateCharts [10 pts]

Consider the given StateChart with a superstate and shallow history.

a) Assume the system is restarted before each following question. Describe the system's state(s) at the end of each input sequence.
   i)  $y \rightarrow z \rightarrow u \rightarrow w$
   ii) $x \rightarrow y \rightarrow v$
   iii) $x \rightarrow y \rightarrow w \rightarrow z \rightarrow v \rightarrow x \rightarrow y$
   iv) $x \rightarrow y \rightarrow w \rightarrow v \rightarrow x$
Solution [1 pt each]

ai) A
ii) C/E
iii) C/F
iv) C/F

b) Draw the equivalent Finite State Machine. [6 pts]
Problem 4 Periodic task scheduling [15 pts]

You are given 4 periodic tasks to run on a single processor.

Task T_1 has execution time C_1 = 2 and period P_1 = 8
Task T_2 has execution time C_2 = 1 and period P_2 = 10
Task T_3 has execution time C_3 = 3 and period P_3 = 10
Task T_4 has execution time C_4 = 7 and period P_4 = 25

a) Are the four tasks EDF schedulable? You may use the Utilization Bound Test. Show your work.

Solution:
2/8 + 1/10 + 3/10 + 7/25 = 0.93 < 1
→ Yes they are EDF schedulable

1 - tried to use utilization bound
1 - correct answer

b) Are the four tasks RM schedulable? You may need to use both Theorem 1 (Guaranteed Schedulability) and Theorem 2 (Exact Schedulability). Show your work. [5]

Solution:
Theorem 1:
Schedulability bound: 4(2^(1/4) - 1) = 0.7568
Task 1-4 Utilization = 0.93 (same as above) → Not guaranteed…

Theorem 2:
Task 1-3 Utilization = 2/8 + 1/10 + 3/10 = 0.65 → Guaranteed schedulable
Task 1, 2, and 3 are guaranteed RM schedulable (according to Theorem 1). Can we add Task 4?
w0 = 2+1+3=7 = 13
w1 = 7 + 2*cei(13/8) + 1*cei(13/10) + 3*cei(13/10) = 19
w2 = 7 + 2*cei(19/8) + 1*cei(19/10) + 3*cei(19/10) = 21
w3 = 7 + 2*cei(21/8) + 1*cei(21/10) + 3*cei(21/10) = 25
w4 = 7 + 2*cei(25/8) + 1*cei(25/10) + 3*cei(25/10) = 27
w5 = 7 + 2*cei(27/8) + 1*cei(27/10) + 3*cei(27/10) = 27

The minimal period for RM schedulability is 27. P_4 = 25 is not RM schedulable.

1 - use theorem 1 correctly
1 - correct interpretation from theorem 1 (inconclusive)
1 - attempt theorem 2
1 - use theorem 2 correctly
1 - correct conclusion from theorem 2
c) Task 1 is modified to occur less frequently, such that $P_1 = 20$. Other tasks remain unchanged. Assume all tasks arrive at $t=0$. Show the EDF schedule for Tasks 1-4 until $t=39$. [4]

<table>
<thead>
<tr>
<th>Time</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
<th>11</th>
<th>12</th>
<th>13</th>
<th>14</th>
<th>15</th>
<th>16</th>
<th>17</th>
<th>18</th>
<th>19</th>
</tr>
</thead>
<tbody>
<tr>
<td>Task</td>
<td>T2</td>
<td>T3</td>
<td>T3</td>
<td>T3</td>
<td>T1</td>
<td>T1</td>
<td>T4</td>
<td>T4</td>
<td>T4</td>
<td>T2</td>
<td>T3</td>
<td>T3</td>
<td>T3</td>
<td>T4</td>
<td>T4</td>
<td>T4</td>
<td>T4</td>
<td>T4</td>
<td>T4</td>
<td>T4</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Time</th>
<th>20</th>
<th>21</th>
<th>22</th>
<th>23</th>
<th>24</th>
<th>25</th>
<th>26</th>
<th>27</th>
<th>28</th>
<th>29</th>
<th>30</th>
<th>31</th>
<th>32</th>
<th>33</th>
<th>34</th>
<th>35</th>
<th>36</th>
<th>37</th>
<th>38</th>
<th>39</th>
</tr>
</thead>
<tbody>
<tr>
<td>Task</td>
<td>T2</td>
<td>T3</td>
<td>T3</td>
<td>T3</td>
<td>T1</td>
<td>T1</td>
<td>T4</td>
<td>T4</td>
<td>T4</td>
<td>T2</td>
<td>T3</td>
<td>T3</td>
<td>T3</td>
<td>T4</td>
<td>T4</td>
<td>T4</td>
<td>T4</td>
<td>T4</td>
<td>T4</td>
<td>T4</td>
</tr>
</tbody>
</table>

d) Assume all tasks arrive at $t=0$. With $P_1 = 20$, show the RM schedule for Tasks 1-4 until $t=39$. [4]

<table>
<thead>
<tr>
<th>Time</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
<th>11</th>
<th>12</th>
<th>13</th>
<th>14</th>
<th>15</th>
<th>16</th>
<th>17</th>
<th>18</th>
<th>19</th>
</tr>
</thead>
<tbody>
<tr>
<td>Task</td>
<td>T2</td>
<td>T3</td>
<td>T3</td>
<td>T3</td>
<td>T1</td>
<td>T1</td>
<td>T4</td>
<td>T4</td>
<td>T4</td>
<td>T2</td>
<td>T3</td>
<td>T3</td>
<td>T3</td>
<td>T4</td>
<td>T4</td>
<td>T4</td>
<td>T4</td>
<td>T4</td>
<td>T4</td>
<td>T4</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Time</th>
<th>20</th>
<th>21</th>
<th>22</th>
<th>23</th>
<th>24</th>
<th>25</th>
<th>26</th>
<th>27</th>
<th>28</th>
<th>29</th>
<th>30</th>
<th>31</th>
<th>32</th>
<th>33</th>
<th>34</th>
<th>35</th>
<th>36</th>
<th>37</th>
<th>38</th>
<th>39</th>
</tr>
</thead>
<tbody>
<tr>
<td>Task</td>
<td>T2</td>
<td>T3</td>
<td>T3</td>
<td>T3</td>
<td>T1</td>
<td>T1</td>
<td>T4</td>
<td>T4</td>
<td>T4</td>
<td>T2</td>
<td>T3</td>
<td>T3</td>
<td>T3</td>
<td>T4</td>
<td>T4</td>
<td>T4</td>
<td>T4</td>
<td>T4</td>
<td>T4</td>
<td>T4</td>
</tr>
</tbody>
</table>
Problem 5 Dependent task scheduling [10 pts]

You are given a code segment as shown below:

1. \( x = i2 \times i3 \times i4; \)
2. \( y = i5 + i6; \)
3. \( z = x \times y; \)
4. \( w = i1 + i8; \)
5. \( o1 = w + z; \)
6. \( o2 = y - i7; \)
7. \( o3 = i8 \times i9; \)

a) Draw the task graph representing the dependencies in the code.
b) Show how the code can be scheduled with:
   i) ASAP (no resource constraint)
   ii) ALAP (no resource constraint)
   iii) List (Assume there are only 2 adders and 1 multipliers)
**Problem 6 Real Time I/O [10 pts]**

The Time-Triggered Protocol (TTP) is a high-speed, fault-tolerant communication network intended for safety critical applications.

a) The TTP/C variant is claimed to be more "fault tolerant" than TTP/A. **List** three differences between TTP/C and TTP/A and **explain** how each one supports this characterization. [4]

**Solution:**
Features missing in TTP/A:
- distributed vs. master slave
- replicated nodes ("Fault Tolerant Units")
- duplicated buses
- data integrity with CRC field vs. parity bit

b) Is the Controller Area Network (CAN) bus event-triggered or time-triggered? **Explain.** [2]
Event-triggered

c) Is FlexRay event-triggered or time-triggered? Explain. [2]
Time-triggered with a dedicated TDMA slot to handle events

d) In general, what types of system/applications benefit from an event-triggered protocol over a time-triggered one? [2]
Tight resource constraints requiring a simple implementation and low bandwidth requirements. Some low-priority events can tolerate slower/unpredictable response times.
Problem 7 RTOS [10 pts]

A real-time operating systems aims to remove many sources of unpredictability in terms of hardware and software interactions.

  a) List 3 ways real-time tasks and interrupts are handled differently in RTLinux than in regular Linux. [3]
  b) Consider message passing vs. shared memory for processors that need to share data. Does one method guarantee a faster response time over the other? Why or why not? [3]
  c) What is priority inversion? What are two possible solutions for it? Briefly explain each one. [4]

Answers vary