Problem 1. Petri Nets [15 pts]

Consider this specification of a Petri net, with $P$ places, $T$ transitions, $F$ flows and $W$ arc weights:

$P = \{P_1, P_2, P_3, P_4, P_5\}$

$T = \{T_1, T_2, T_3, T_4\}$

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

$W(P_2, T_2) = 2$

$W(T_2, P_2) = 2$

... all other arc weights are 1.

Given initial marking $M_0 = [1, 2, 0, 0, 0]$:

a. Draw the Petri net. [5]

b. Draw the reachability tree/graph. [3]

c. What is the liveness of transition $T_1$? [1]

d. What is the liveness of transition $T_2$? [1]

e. What is the liveness of transition $T_3$? [1]

f. What is the liveness of the net? [2]

g. Is the net pure? [1]

h. Is the net conservative? [1]
Problem 2. Synchronous Data Flow [10 pts]

Consider this SDF graph and answer the following short questions [2 pts]:

- a. Each time node A executes, it **produces** / **consumes** _____ token(s).
- b. Each time node B executes, it **produces** / **consumes** _____ token(s).
- c. There is/are _____ token(s) stored on the arc at start up.
- d. The sample produced at node A’s 1st invocation is consumed at node B’s ____st/th invocation.

Consider another SDF graph [8 pts]:

- e. Write down the incidence matrix. Use columns to represent nodes and rows to represent edges.
- f. What is the rank of the matrix?
- g. How can we infer the existence of a periodic admissible sequential schedule (PASS) based on this rank?
- h. Show the number of invocations for each node in 1 cycle of the schedule. (E.g. The schedule "AABC" would give answers A=2, B=1, C=1).
  - A:
  - B:
  - C:
  - D:
- i. Find the initial number of elements in each edge buffer for your PASS schedule (Hint: execute one cycle of your PASS schedule manually to find the number of initial elements necessary in each buffer.)
- j. Find the minimum size required of each edge buffer.
Problem 3. Esterel [10 pts]

Examine the following Esterel code snippet:

```esterel
await A emit Allo!
present B then
    emit Beep!
end present
await C do
    emit Cheep!
end await
pause;
weak abort
loop
    present D then emit E else emit F;
    pause;
end loop
when Q
emit G
sustain H
emit I
```

a. Given the following timeline and input sequence, list all expected outputs at each time point. [5]

Hint: You may find this extensive reference paper useful:
https://www.researchgate.net/publication/242374294_The_Esterel_v5_Language_Primer_Version_v5_91

```
B C
A B D D D D Q
```

b. Draw the equivalent finite state machine (FSM). [5]

Hint: Use 'x' as the notation for a "don't care" signal value in FSM'
Problem 4. Verilog [5 pts]

For each subsection, assume that the initial values on each signal are:

\[
a = 1; \quad b = 2; \quad c = 3; \quad d = 4; \quad e = 5;
\]

In VHDL, signals (not variables) are analogous to wires in hardware - they may contain different delays when new values are assigned. Consider these non-blocking statements in VHDL:

\[
a <= b+c;
b <= d+e;
c <= a;
\]

a) In a world with no wire delays, there may be non-determinism of the final values. List two possibilities for the set of final values of \(\{a, b, c\}\). [2]

b) Now add a delta delay of 1 ns to each assignment. What are the final values of \(\{a, b, c\}\)? [1]

In Verilog, non-blocking statements may be executed in parallel, but the assignments do not happen until all right-hand-side statements have been evaluated. Consider this Verilog snippet:

\[
a <= b+c;
b <= d+e;
c <= a;
\]

c) What are the final values of \(\{a, b, c\}\)? [2]
Problem 5. StateCharts / SDL [15 pts]

Consider this description of a single node in a distributed sensing network:

1. At startup, the node waits to receive configuration parameters, like the required sampling period and sensor trigger threshold. It waits until a configuration message is received ("Conf").
2. The node will start sampling from its attached sensors.
3. After sample collection, the data is processed.
4. After processing, the node checks for any anomalies detected on board or from neighboring sensors (collectively represented as an "Outlier" signal).
5. If an outlier is detected, the sensor emits an alert ("Alert"), then immediately goes back to sampling.
6. If no outliers are detected, the sensor emits a confirmation ("Typ"). It then calculates any remaining time until the next sampling period and sets a timer for that length "T". The node will sleep until the timer goes off, then return to sampling (step 2).
7. At any point during steps 2-6, a low-battery ("Low") signal may be received. This will send the node into low power operating mode.
8. In low power mode, the node immediately goes to sleep.
9. In low power mode, sleep will only be woken up by a hardware message from the sensors if their readings pass a certain threshold ("Thres").
10. When the node wakes in low-power mode, it takes a sample from the sensors and goes back to sleep without processing the raw data.
11. At any point in low-power mode, if a charging cable is supplied to the node (indicated by a signal "Chrg"), it will return to normal operations and take a new sample (step 2).

a. Draw a StateChart that accurately represents the functionality of a single node. You should use superstates where appropriate to make your state diagram more readable. [10]

b. Now assume that the node simply has one normal operation mode (no low power mode). Draw the equivalent SDL representation. Apart from the low power mode, your answer must be consistent with the StateChart shown in part a). [5]