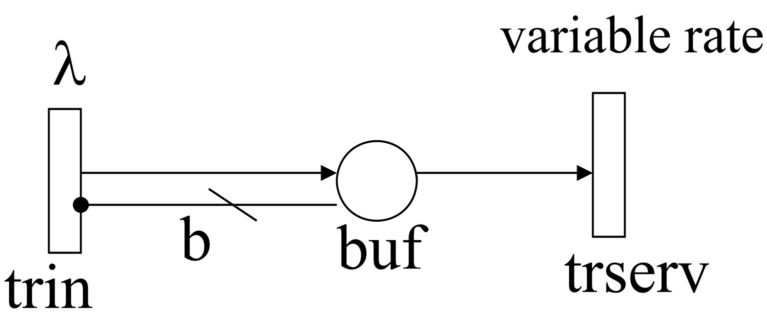
HW#3, Problem #1

Can make use of the SPN model and code on slide #175 for M/M/m/b (with m=3 and b=10)



HW#3, Problem #2

Reliability Modeling and Analysis of a 3P2M system using SPNP

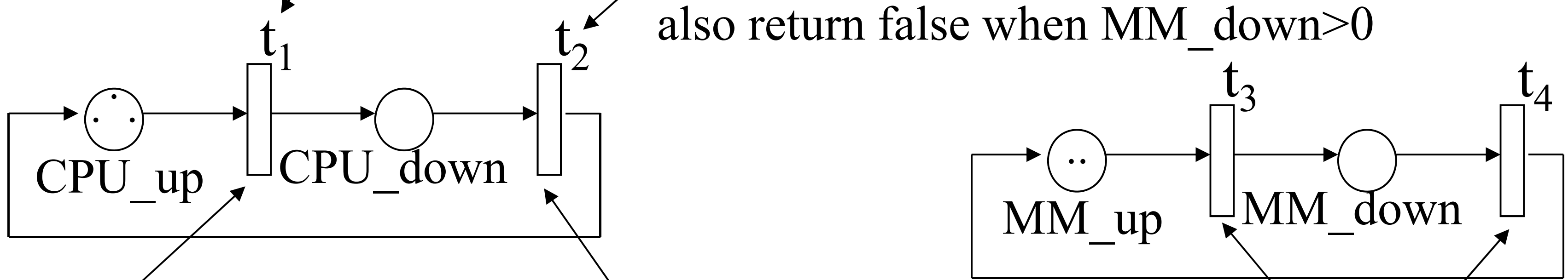
(a) Each component (CPU or memory) has an independent repair facility

(b) Each subsystem (CPU or memory) has an independent repair facility that can repair failed components within the subsystem one at a time.

(c) The whole system shares a repair facility which repairs failed components one at a time with the repair priority of memory modules over CPUs.

Enabling function: return false when CPU_up==0 or MM_up==0

Enabling function: return false when CPU_up==0 or MM_up==0; for case c only: also return false when MM_down>0



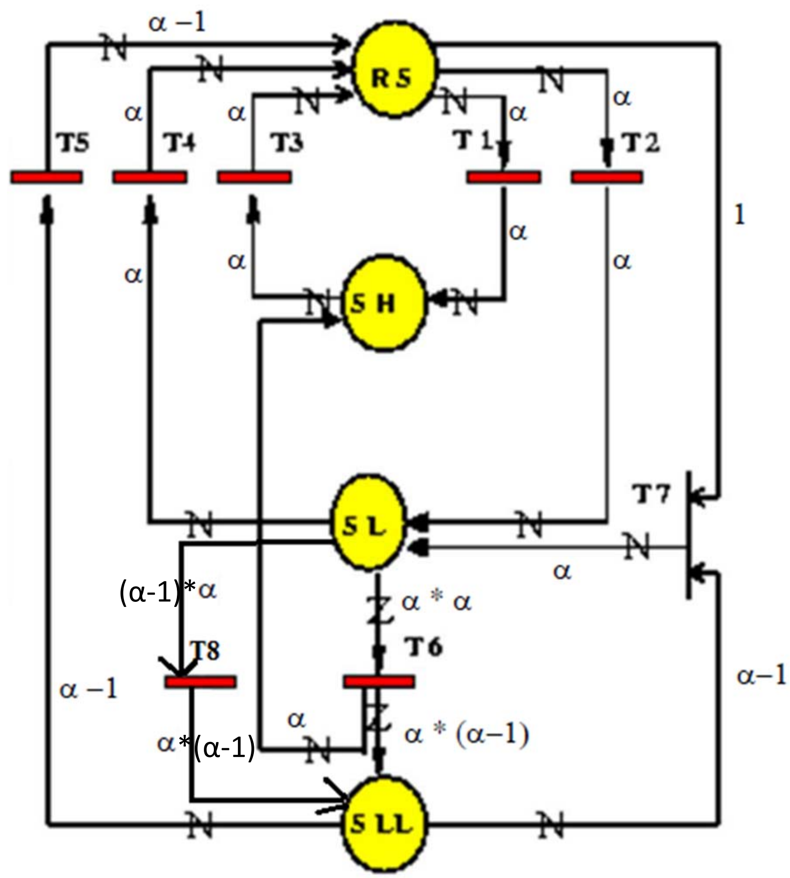
variable rate: mark("CPU_up")*per-CPU failure rate

variable rate: For case (a) it is mark("CPU_down")*per-CPU repair rate; for cases (b) and (c) it is just per-CPU repair rate

t_3 and t_4 each have a variable rate function and an enabling function.

HW#3, Problem #3

(α = 2, so a token represents one-half slot)



A new transition "T8" with rate λ_L is added to account for subrating for low-priority clients

For the homework in which only the middle partition exists, the enabling condition for "T8" is mark("RS")==0

For case study #2 in which all three partitions exist, the enabling condition for "T8" is mark("RL"==0) && mark("RS")==0

Ex1: Reward assignment function for calculating the population of low-priority clients:

reward_type population_low_priority() {return (mark("SLL")/(α -1) + mark("SL")/α);}

Ex2: Reward assignment function for calculating the throughput of low-QoS, low-priority clients:

reward_type throughput_low_QoS_low_priority() {return rate("T5");}

same as

mark(SLL) /(α-1) * μ

HW#3, Problem #4