Embedded System Design and Modeling

EE382N.23, Fall 2019

Homework #1

Models of Computation

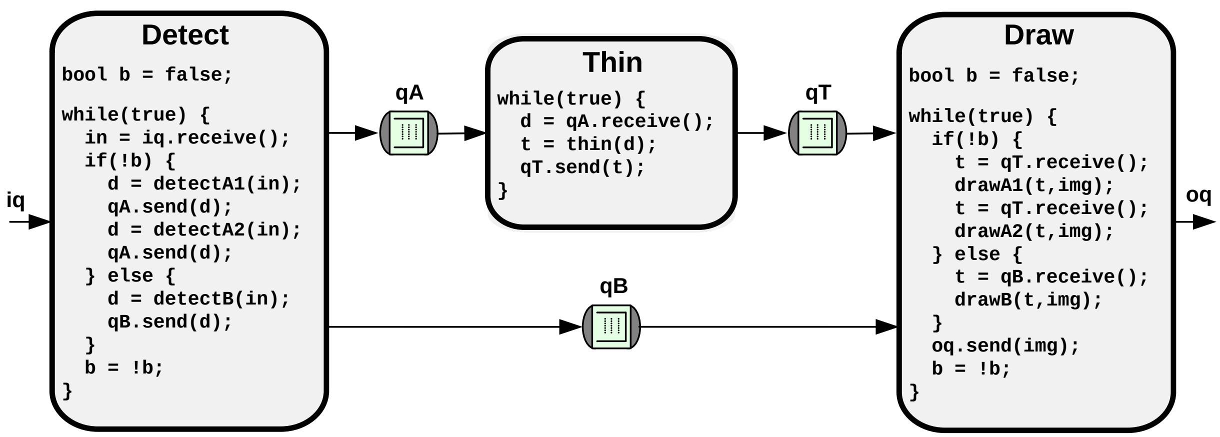
Assigned: September 12, 2019
Due: October 3, 2019

Instructions:

- Please submit your solutions via Canvas. Submissions should include a single PDF with the writeup and a single Zip or Tar archive for any supplementary files (e.g. source files, which has to be compilable by simply running 'make' and should include a README with instructions for running each model).
- You may discuss the problems with your classmates but make sure to submit your own independent and individual solutions.
- Some questions might not have a clearly correct or wrong answer. In general, grading is based on your arguments and reasoning for arriving at a solution.

Problem 1.1: KPN

Given the following KPN model:



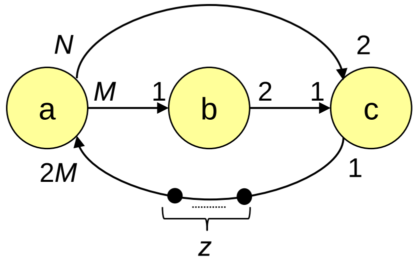
(a) Can this KPN be run in a complete, bounded and non-terminating manner? Show a data-driven schedule (as the sequence of *detectX()*, *thin()* and *drawX()* calls made) for processing of two *iq* tokens on a single processor with minimal number of context switches between processes. What is the buffer size needed for each queue?

(b) Show the schedule for processing of two *iq* items using Park's algorithm. What buffer size is needed? What is the minimum buffer size needed to run this model?

(c) Can this KPN be converted into an equivalent dynamic, Boolean, cyclo-static or synchronous dataflow model? If so, show the conversion into the most restrictive dataflow variant possible.

Problem 1.2: Synchronous Dataflow (SDF)

Given the following SDF graph, where *N*, *M* and *z* are integer parameters:



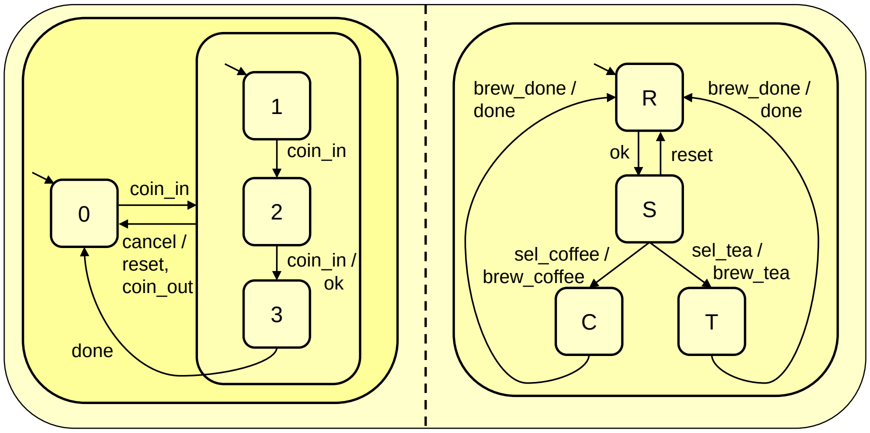
(a) Write down the balance equations and determine for which integer values of *N* and *M* the graph is consistent. Give the repetition vector for a minimal (least integer) schedule as a function of *N* and *M*.

(b) What is the minimal number of initial tokens *z* for which the graph is schedulable (as a function of *N* and *M*)? Set *N*, *M* and *z* to their minimal integer values that ensure consistency and schedulability, and write down a periodic single-processor schedule that minimizes buffer sizes. What is the minimal memory requirement to execute the graph?

(c) Now assume that *z* = 0. Is there an assignment of initial tokens to other channels that makes the graph schedulable? If so, what are the minimal values of initial tokens needed on other arcs? Set *N* and *M* to their minimal integer values that ensure consistency and find an assignment of initial tokens and a periodic single-processor schedule that minimize buffer sizes. What is the minimal memory requirement to execute the graph?

Problem 1.3: HCFSM

Given is the following HCFSM model of a simple vending machine. The machine has 5 external inputs (a *coin_in* slot, a *cancel*, *sel_tea* and *sel_coffee* button, and a *brew_done* signal), 3 external outputs (*brew_tea* and *brew_coffee* signals to trigger the brew unit operation, and a *coin_out* signal to release inserted coins), and 3 internal signals for communication between state machines (*ok*, *reset*, *done*). Unless specified otherwise, signals are by default absent and states have implicit self-transitions.



(a) Demonstrate the operation of the vending machine for a customer ordering tea. Show the sequence of events and state transitions.

(b) Do you see any way to further simplify and reduce complexity (e.g., number of states or number of transitions) of the model using any of the state machine concepts we discussed in class? If so, just sketch where and how the model could be changed, you don't need to show the complete simplified state machine.

(c) Convert the HCFSM into an equivalent single, flat FSM.

(d) The state machine has a bug that allows the user to cheat. Show a trace of input events and transitions that demonstrates the bug. Show how the original HCFSM and the flattened FSM need to be modified to fix the bug (you can indicate changes directly in the graphs above).