Homework 3
Analysis of Software Artifacts
Due Date: Nov 8, 2002 (Fri)

Note: This homework has to be done in *teams of at most two students.* As in the earlier homework, before you start changing the NuSMV model, write a *design document.* Outline your design strategy in the design document. Please be as detailed as possible. Show the design document to a fellow student, i.e., the *reviewer.* Mention the reviewers name on the document. Keep the design document *consistent* with your specification. Please submit the *modified NuSMV specification AND the design document* with the homework. This will also help in grading your homework and will help you with completing your homework as well.

Note: Please use the "base" design as the NuSMV file provided with the homework. This will keep the design consistent as everybody will start from the same code base. Follow the same procedure for question number two.

Question 1 (Modeling 70 points): You will enhance the specification in the following way:

(Part A):
Multiple Priorities
Users have two levels of priority: *high* and *low.* Assume that we have two users of each priority. Within users of the same priority use the round-robin scheduling policy.[1] If two users of different priorities are waiting, the user with the higher priority gets the machine.

(Part B):
More coins
Assume that each drink costs *fifty cents.* User deposits *dimes* or *quarters.* As soon as the user (who has the vending machine) has deposited coins worth greater than or equal to fifty cents, they can get the drink. Of course the vending machine will have to return the spare change.
(`Hints:`) You will probably have to keep an extra variable called `amount-accrued`

[1] We will discuss round-robin scheduling in the lectures.

which keeps track of how much the user has deposited. In the vending machine, there should be a variable called `amount-returned` which is equal to the spare change. First you should figure out how many possible values could the variables `amount-accrued` and `amount-returned` take. Do not use arithmetic in `NuSMV` to keep track of `amount-accrued` and `amount-returned`. Use explicit transitions to simulate addition, e.g., if `amount-accrued` is `twenty` and the user deposits a dime, then `amount-accrued` becomes `thirty` in the next step.

Question 2 (Modeling in SPIN 30 points): This question consists of the following three parts:

Part A: Explain the *Mars Pathfinder glitch* in detail. Use a diagram (similar to the one we used in the lecture) to illustrate the priority inversion problem.

Part B: Construct a toy model of the relevant portions of the Mars Pathfinder in PROMELA, the input language of SPIN. Model check your design using the specifications corresponding to absence of priority inversion. You should be able to find the *LTL* specifications in one of the documents that I provided. As usual, write a detailed design document.

Part C: Change the model constructed in part (b) to implement the *basic priority inheritance* protocol (or *BIP*). The *LTL* specification should now become true.