VIS Technology Transfer Course

University of California, Berkeley

Lab 8 : Fairness Constraints and Monitors

Serdar Taşıran

5/15/96

1 Introduction

The first part of this lab session will focus on using fairness constraints inside **VIS**. We will look at the non-deterministic client module from the previous labs, and will see why it does not model an actual client adequately. We will then add a fairness constraint to this description and see that we are able to prove more properties with this refined description.

In the second part we will work on writing monitors describing properties. We will start with a description of a property in English. This will tell us what the undesired behavior is. From this we will derive a monitor that captures this undesired behavior.

2 Hands on Experience

2.1 Fairness Constraints

1. Consider the system described in "ex3.v". Write a CTL property that says "It is always the case that if client A requests the bus, it eventually gets it."

2. Check this property using VIS. Does it pass? If it does, go to section 2.2. If not, go on to 3 below.

3. Examine the error trace. Why does the property fail? Write down in English a fairness constraint on the clients that would remove this undesirable behavior. How would you write this constraint in the format $F^{inf}S_1$ & $F^{inf}S_2$ & ... & $F^{inf}S_n$?

4. Write this fairness constraint in a file called "ex3.fair". After the "read_blif_mv ex3.mv" and "init_verify" commands, read in the fairness constraint using the command "read_fairness ex3.fair". Now check the property.

2.2 Monitors

5. Recall the *Equal Opportunity* property from Lab 6: "If A is holding the bus and B is requesting, then it is not possible for A to release control of the bus and get it back, before B gets control of the bus ". Write down what "undesired behavior" would look like in this case. Then draw a state transition diagram for a monitor capturing this bad behavior. Convince yourself that the monitor can satisfy its fairness constraint if and only if the system has a trace violating this property.

6. Look in the file "ex4.v". This file contains the system description from "ex3.v" and a template for writing a monitor. Write verilog code for the monitor in 5. Also create a file called "ex4.fair" which should consist of "ex3.fair" augmented with the fairness constraint for the monitor.

   The **lang_empty** command checks if the system (including the monitor, if there is one) has any legal traces at all, i.e., if the "language" of the system is empty. We will use this command to see if our system-monitor combination has any traces that satisfy the fairness constraints. The desired result is that the language is empty, i.e., the language emptiness check passes. If not, we will have a trace that satisfies the fairness constraints of the monitor, which means that the system can generate undesired behavior.

7. Now check the property using the read_blif_mv, init_verify, read_fairness and lang_empty commands.

(Bonus problem) 8. Prove the following property using the approach described above: "If A requests the bus and the bus is free, then A gets control of the bus within 5 clock cycles unless B or C request the bus in the meantime".