Formal Verification

General Introduction to the Course

Jacques Fleuriot
jdf@inf.ed.ac.uk
Overview

- Lecturers: Jacques Fleuriot (Part 1) and Paul Jackson (Part 2).
- When: 16.10-17.00 on Tuesdays and Fridays.
- Labs: Forrest Hill 1.B31 between 9:00 and 10:50 on Fridays (Week 2 onwards, TBC).
- Web page: http://www.inf.ed.ac.uk/teaching/courses/fv
- Note: This is a new course so some aspects may change as the semester progresses. Constructive feedback is welcome (as always).
Entry Requirements

- Students are expected to be familiar with discrete maths at a level similar to our Discrete Mathematics and Mathematical Reasoning (INFR08023) course.
- Prior exposure to first-order logic is also expected.
- Programming experience in an imperative language such as Java, C or C++ is also essential for handling the material related to software verification.
Contents Overview

- Part 1: Model checking:
  - Temporal logic, CTL, BDDs, etc.
  - Tool: NuSMV, a mature, free tool that illustrates a range of concepts.

- Part 2: Several topics (TBC, under development) e.g.
  - Operational semantics of a (simple) imperative programming language, weakest precondition operators and verification condition generation.
  - Assertion-based software verification.
  - Tools: The SPARK Ada toolset and Why3 from INRIA.
Assessment

- Assessment is based on a final exam only i.e. there is no assessed coursework.
- Practical exercises: At least 2 (1 for each part):
  - Aims: To gain Familiarity with formal verification tools and help you understand formal verification techniques.
  - Formative feedback will be provided through peer reviews and demonstrators/TAs/lecturers will also review your solutions to practical exercises.
  - Practicals will be introduced during lectures and notes on the solutions will be provided.