Where is it? Find the code you are interested in!

Abstract - The demonstration presents our tool for feature localization and debugging of RTL-designs. Feature localization helps a designer to find the code relevant for a certain feature and, thus, helps him to faster understand a design previously unknown to him. The developer can choose between three basic techniques for feature localization. In the area of debugging the tools allows fault localization, reverse debugging based on dynamic data- and control-flow of the design and dynamic slicing.

Techniques:
- Feature localization:
  - Different basic approaches: statement- and toggle-coverage based, dynamic dataflow based and mutation testing based.
  - Requires use cases which use a feature and use cases which do not use the feature. The system need the information which use cases use which feature.
  - In case of dynamic dataflow the request to use the feature and the result of the feature must be marked
- Debugging:
  - Based on the dynamic dataflow analysis. All the techniques work on the dynamic dependency graph

Results:
- Feature localization:
  - For the evaluated designs more precise localization than the documentation of the designs is achieved.
  - Coverage based feature localization is fast and requires few manual effort.
  - The dynamic dataflow based approach and the mutation testing based approach yields better results than the coverage based approach
  - However, dynamic dataflow based approach requires more manual effort and more simulation time; the mutation testing based approach requires significantly more simulation time.
- Debugging:
  - In the case studies the time to correct errors was reduced by 50% on average.
  - Especially good in case of incorrect operands or operators.
  - The least improvements in case of missing code.

Related Publications:

Contact:
Jan Malburg
University of Bremen
Bibliothekstrasse 1 (MZH)
28359 Bremen, Germany
Email: malburg@informatik.uni-bremen.de
Phone: +49 (421) 218 - 59884