Future Microprocessor Verification Issues

Warren A. Hunt, Jr.

Department of Computer Sciences
1 University Station, M/S C0500
The University of Texas
Austin, TX 78712-0233

E-mail: hunt@cs.utexas.edu
TEL: +1 512 471 9748
FAX: +1 512 471 8885

Overview of the Talk

- Introduction and Position
- Future Challenges
- Research Directions

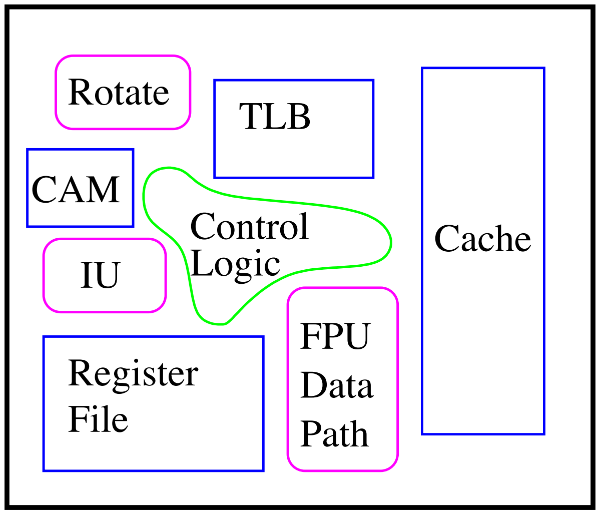
Introduction and Position

- The application of formal methods to design projects varies widely, for many practical reasons:
  - size and importance of project,
  - available tools and people,
  - confidence of architects and managers in the available technology,
  - degree of integration of formal methods tools in the tool flow,
  - size of the company, and
  - duration of the project.
- The use of formal methods represents a vexing management challenge, due to the lack of available metrics to know when its use is efficient and sufficient.

Cooperating FSMs and "Embedded" Specifications

Microprocessors are designed as many cooperating FSMs.

- Each FSM generally works as fast (eagerly) as possible.
- Specifications are annotated with all kinds of different information.



RTL Description & Annotations

```
reg1 <= b_bus + a_bus;
n = max( a, b );
cache[n] <= mem_bus;
enable = a & cntl[3];
reg[i] <= sel( enable, i_bus, data_in);
```

```
//  b_bus[0] ~= a_bus[0];
//  b > 0;
//  Ax( cntl[1] ∨ cntl[3]);
//  fp_value(reg1) == fp_value(b_bus) + fp_value(a_bus)
//  C1( reg1 );
```

- Many tools (compilers, synthesis, simulators) operate on the RTL.
- Many scripts (Perl, TCL, sed) strip out pieces for other tools.
- There is no uniform, consistent, RTL/annotation-language.

A Possible Future

We envison a system called **FMaAT** (Formal Modeling and Analysis Tool).

- **FMaAT** needs to be able to read, compile, and "model build" the entire design specification.
- The **FMaAT** system must, in all respects, operate in a hierarchical manner.
- **FMaAT** needs to contain all embedded annotations.
- **FMaAT** needs to be able to act as a database engine that allows every design module and interface to be identified.
- **FMaAT** needs to be able to compute cones-of-influence, bus conflicts, improper connections, and other user-definable queries.
- **FMaAT** must have a command-line interface. In a big design project, tools are always "taped" together with scripting languages to overcome deficiencies in the design flow. **FMaAT**'s command-line interface should itself be described formally.

A Possible Future, continued

- There has to be a way to re-run all checkers, simulators, etc., automatically whenever there is any change to the design. Automatic regression verification is a must.
- There must not be any way to get a false positive. There must be provisions for ensure vacuity checking for analysis requests.
- If possible, **FMaAT** should have some kind of analysis "coverage metrics".
- A purely functional verification system is not sufficient. **FMaAT** must a way to specify and verify non-functional properties such as:
  - power requirements,
  - circuit sizes,
  - wire types,
  - physical location data,
  - environmental requirements, and
  - other critical design properties.

A Possible Future, 2nd continued slide

- **FMaAT** must provide a means to write a truly rigorous high-level specification.
- **FMaAT** must deal with a distributed design process. No project of a significant size is all done in a single place.
- Finally, **FMaAT** must safely extensible; that is **FMaAT** should be no more difficult to extend than Emacs, but **FMaAT** should impose a discipline that ensures that extensions do not render existing checker and verifiers unsound.

A tool like **FMaAT** will require a much more general language than those commercially available, such as Verilog and VHDL and their derivatives.

The limitations of the available languages are causing the specification problem to actually become worse because designers are forced to record their design properties as comments or in external files. System-C and System Verilog are not a long-term answer; in fact, these languages are creating yet more problems.

Research Problems

To make a system like **FMaAT** will require fundamental changes to the infrastructure of commercial design environment.

- New formally specified design and annotation languages need to be defined that
  - provide a semantically unified framework for designs and all associated specifications; and
  - provide mechanisms to represent all of the design "meta" data directly as a part of the design specification.
- Typical two- and four-valued simulators need to be extended to symbolic simulators.
- All of the analysis tools (e.g., equivalence and model checkers, (G)STE engines, reachability analysis, theorem provers) must all read the same design data and all follow the same semantics.
- A new suite of non-functional checkers (e.g., power) need to be developed.

Research Problems, continued

- Post-silicon design approaches need to be integrated into the formal design process.
- Autonomic systems that automatically (re-)run all checkers and provers should be automatically started any time any part of a design is changed.
- Formal approaches to (microprocessor) security need to be developed.