

## Formal Methods

Since the last decade and a half, Formal Methods is receiving a lot of attention. Formal methods are based on rigorous mathematical techniques and allow one to prove or disprove properties of system models. R&D in this rapidly evolving area has resulted in a rich variety of techniques and tools suitable for use at different stages in a system's design cycle.

A formal approach to verifying software / hardware systems consists of the following steps:

- ◆ **Specification:** A precise statement of the intended properties of the system. Two Main approaches are: logic based (e.g. Z, B methods) and automata based (as in Formal Check).
- ◆ **Modeling:** A precise description of the behavior of the system. The description can be at different levels of detail, such as abstract state transition systems (e.g. Statecharts), programs in a high level language (e.g. Promela, Verilog, VHDL) etc.
- ◆ **Verification:** A formal procedure for establishing that the system model satisfies the specified properties.

Two widely used methods for checking properties are :

- ◆ **Model checking:** This is a method for automatically checking whether a given model of the system satisfies a specified property. If so, the model checker returns 'yes'. Otherwise, it returns a 'run' of the system that violates the property.

SMV (from CMU and Cadence-Berkeley Labs, USA), VIS (developed at University of California, Berkeley, USA), FormalCheck (commercial tool from Cadence Inc., USA) are some of the model-checkers that have been used successfully in the verification of large control-intensive hardware designs. These tools have found subtle bugs in designs that were cleared by conventional verification methods.

- ◆ **Theorem Proving:** This is a deductive approach in which logical theories are used to deduce that a property is satisfied by a system model.

PVS (from SRI, USA), STEP (from Stanford University, USA), and HOL (from Cambridge University, UK) are some of the powerful theorem proving tools developed in recent years and used in the verification of microprocessors, programming language implementations etc.



## Ongoing / Completed Projects

- ◆ Verification of Embedded Systems: Industrial Case Studies
- ◆ Static Assertion Checking Tool
- ◆ Tools for Design and Synthesis of Process Control Software
- ◆ Case-Studies in Hardware Verification
- ◆ Probabilistic Interface Timing Verification
- ◆ Timing Analysis and Verification of Gate-level Asynchronous Circuits
- ◆ Verification Environment for Distributed Control Application
- ◆ Efficient Verification of Synchronous Programs
- ◆ Approximate Model Checking



Conference Room

## Principal Investigators

S. Ramesh (IITB)  
 G. Sivakumar (IITB)  
 Supratik Chakraborty (IITB)  
 Sridhar Iyer (IITB)  
 S.D. Dhodapkar (BARC)  
 A.K. Bhattacharjee (BARC)  
 R.K. Shyamasundar (TIFR)  
 P.K. Pandya (TIFR)

## For More Information Contact:

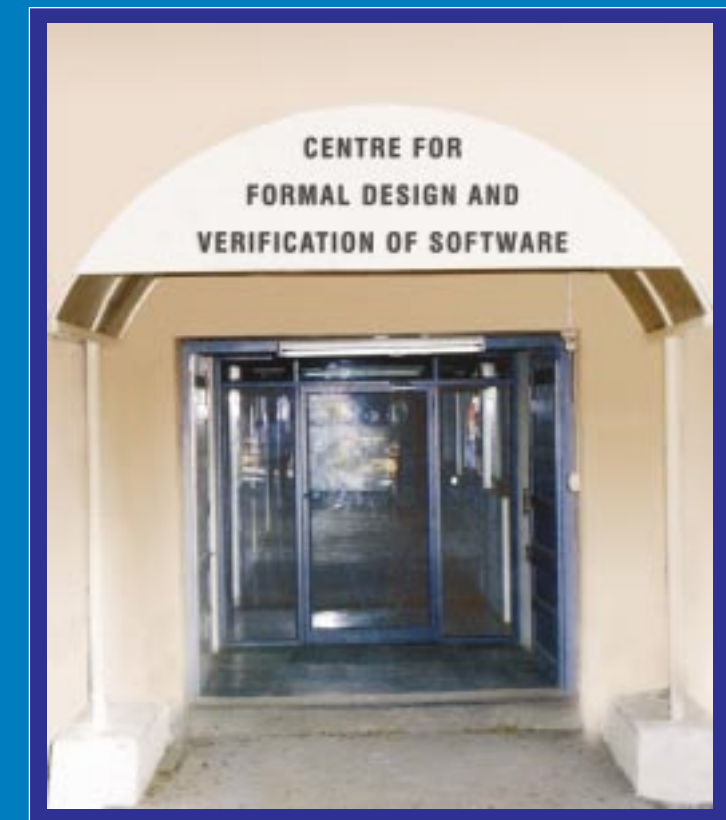
Prof. S.Ramesh, Head  
 C.F.D.V.S  
 I.I.T., Bombay  
 Mumbai - 400 076  
 Phone : +91 - 22 - 576 8700  
 email : head@cf dvs.iitb.ac.in

## INDIAN INSTITUTE OF TECHNOLOGY, BOMBAY

Email: office@cf dvs.iitb.ac.in URL: http://www.cf dvs.iitb.ac.in  
 Phone: +91 - 22 - 576 8701  
 Fax: +91 - 22 - 572 0196



# Centre for Formal Design and Verification of Software



Indian Institute of Technology, Bombay  
 Mumbai 400076, India.

Computational Systems are becoming pervasive in every aspect of human life from day-to-day household appliances, banking and trading sectors, aircraft control to mission critical satellite launchers. The quality of these systems has a direct bearing on the quality of human life. Many of these applications are complex, making the development of high quality (dependable) software or hardware for these applications a very challenging task. Further, some of the applications of computers in nuclear reactors, space, avionics, industrial process-control and robotics are safety-critical or mission-critical in the sense that failures can result in loss of lives, mission failure and in some cases, economic loss.

Dramatic examples of dire consequences of even minor bugs in such applications are many ( see "*Computer Bugs are Costly*" ). Sound methods of design and rigorous Verification & Validation (V&V) are essential for safety-critical or mission-critical systems, also known as High Integrity Systems.

Current industrial practices for development of high integrity systems are far from satisfactory. They employ traditional V&V methods like testing and simulation which are



Projects Laboratory

inadequate for many reasons: they detect presence of bugs rather than guaranteeing their absence, late detection of bugs leads to costlier design cycles, large amount of resources are spent on verification of complex systems, etc.

The Centre for Formal Design and Verification of Software (CFDVS) has been set up in IIT Bombay with the mission of promoting, enabling, supporting and generating trained human resource for rigorous design and verification of software and hardware systems. The focus of the Centre is on the use of formal techniques for design and verification of high integrity safety-critical industrial systems ( see "*Formal Methods*" ).

#### Objectives of the Centre

- ◆ Advance the state-of-the-art in development of high integrity software and hardware systems, and develop environments and tools based on sound methodologies for industrial scale applications and practice.
- ◆ Effectively contribute towards formulation of standards and guidelines for the development of high integrity software and hardware.
- ◆ Create infrastructure and develop expertise to carry out Verification & Validation (V & V) at different stages of system development and emerge as a leading agency to provide know-how and consultancy for Independent V & V (IV & V)

for high integrity systems.

- ◆ Provide education and training in design, verification & validation of high integrity software and hardware systems.

Over the last few years, the Centre has acquired the necessary infrastructure and resources for fulfilling its objectives. The Centre has well-furnished premises consisting of two laboratories, a conference room, a few faculty rooms and a library. Its hardware resources include high-end servers, many workstations and access stations with Internet and ISDN connectivity. It has built up a unique collection of a large number of state-of-the-art commercial and public domain verification tools for hardware and software systems, advanced static & dynamic analysis and testing tools, modeling languages and related environments for real-time software, UML tools and general programming environments. The Centre has an Endowment Fund to support Masters and Doctoral students with fellowships.

Progress has also been made in R&D activities over these years. A unique tool developed in the Centre in collaboration with BARC, is the Static Assertion Checking tool, that enables formal verification of programs compliant to the MISRA subset of C against their specifications. This tool has been extensively used in the verification of some industrial embedded systems. Case studies in software and hardware verification are also being conducted as part of the R&D efforts ( see "*Ongoing/Completed Projects*" ). The Centre has also been carrying out projects for agencies like BARC (Mumbai), ADA (Bangalore) and Texas Instruments (Bangalore).



During its few years of existence the Centre has done extremely well in the area of education and training. It has attracted a large number of students in IIT Bombay to work on various projects. The Centre organized a Workshop on Formal Methods in Safety Critical and Industrial Applications in November 2000, which was well attended and appreciated. The Centre has plans to conduct such workshops regularly to promote formal design and verification practices among people involved in development of high integrity applications.

The activities of the Centre are planned and carried out by the faculty members of IIT Bombay and scientists from BARC and TIFR ( see "*Principal Investigators*" ) and students. A large number of students from various disciplines like Computer Science & Engineering, Electrical Engineering, Information Technology and Reliability Engineering are currently working in the Centre.

#### Computer Bugs are costly!

- ◆ The Intel Pentium chip, released in 1994, produced errors in floating point division. The error was later identified using a Theorem Prover. The "Intel Pentium Bug" cost Intel \$475 million.
- ◆ In December 1996, the Ariane 5 rocket was destroyed 40 seconds after take off. One of the two software components controlling the rocket threw an exception causing the program to crash, taking with it the vehicle and its \$500 million payload.
- ◆ The Therac-25 radiation therapy system was controlled by a faulty software that caused the machine to deliver high radiation doses to patients under certain conditions. This eventually led to many deaths.
- ◆ In October 1996 a British Airways Boeing 777-200A was forced to cancel its trip to Jeddah due to uncommanded movement of the rudder and rudder pedals. This was caused by a fault in the autopilot and flight-director computers.
- ◆ The planned lift-off of the space shuttle Columbia was delayed due to a fuel spill. The crew decided to use the extra time to simulate a "trans-atlantic abort sequence". On issuing the abort command, all four flight computers were locked and could not be revived. This was because the code contained a jump on an uninitialized counter to an empty memory location.
- ◆ In 1992, the London Ambulance Service introduced a computer controlled dispatch system. The system was unable to cope with real-time data and generated exception messages. This eventually caused the relevant information to scroll off the controller screens in ambulances, leading to around 20 deaths.



Instructional Laboratory