Abstraction of Concurrency and Communication in VHDL

Peter J. Ashenden
University of Adelaide

Philip A. Wilsey
University of Cincinnati

This work was partially supported by Wright Laboratory
under USAF contract F33615-95-C-1638.

VHDL in the design flow

- At the system level, specify aspects of the system
  - structure (architectural partitioning)
  - behaviour (data transforms, reactions)
  - . . .
- VHDL allows description of structure and behaviour
- Prefer single language approach
  - system level to detailed design (hardware/software)
  - aids refinement
  - avoid semantic mismatch

Problem with VHDL

- Behaviour includes concurrency and communication
- Communication performed using signals
  - assignment, resolution, waits
- Signals do not express synchronization
  - *cf.* system description and programming languages
  - assignment schedules a message for a particular time
  - receiver may miss the message
    - if it does not respond at that time

Ad hoc solutions

- Communication using handshaking protocol
  - requires extra signals
  - requires protocol implementation
- Explicitly instantiated message queues
  - extra components
- Both detract from abstraction of communication
  - introduction of extraneous artefacts

Previous extension proposals

- Vista OO-VHDL (Swamy *et al*)
- LaMI (Benzakki & Djaffri)
- Bournemouth/IBM-UK (Cabanis *et al*)

➨ Based on OO extensions to VHDL

  – entity/architecture as a class, components as objects
  – entity specifies operations invoked by other processes
  – architecture encapsulates state

⇨ Monitor semantics with concurrency control

Other description languages

for describing behaviour, eg:

- StateCharts
- Estelle
- SDL
- CSP
- . . .

➨ Concurrent processes with mess