Simulation Techniques in Metropolis

Claudio Passerone

Outline

◆ Simulation in the Metropolis Infrastructure
◆ Simulation strategy and algorithm
◆ Current status
◆ What's missing?
◆ Demo
Simulation in the Metropolis Infrastructure

- Design methodology
- Base tools
  - Design imports
  - Simulation
- Meta model of computation

Metropolis Formal Methods:
- Synthesis/Refinement
- Analysis/Verification

Simulation strategy and algorithm

- Multithreaded concurrent simulation
  - a thread for each process
  - function calls for communication media interfaces
- Managing the simulation
  - process threads run until constraints or await need to be evaluated
  - a manager allows only selected processes to proceed
    - high abstraction level: satisfy constraints (using schedulers as help)
    - low abstraction level: verify properties
- Prototype written in Java 1.2
Simulation strategy and algorithm

- Processes
  - instances of a specialized class of *process*
  - thread \( \Rightarrow \) run function of a thread
  - port \( \Rightarrow \) variable (type is an interface)

- Communication Media
  - instances of a specialized class of *medium*
  - implement interfaces using member functions

- *await*
  - translated into a simulation model, interacting with the manager

Simulation strategy and algorithm

MAKE REQUEST

PROCESS

WAIT (evaluate conditions)

PROCEED

START

CHECK PRECONDITIONS FOR AWAIT

run list

enabled list

RUN SCHEDULERS

Satisfy (Check) Constraints

RUN PROCESSES SET POSTCONDITIONS

running list

END

MANAGER

SCH1

SChn

SCHEDULERS
Simulation strategy and algorithm

```java
await {
    (cond₁) [iflist₁] {stmt₁;}
    ...
    (condₙ) [iflistₙ] {stmtₙ;}
}
```

```java
Lock[][] lock = {iflist₁}, ..., {iflistₙ};
setProgramCounter(pc);
myManager.addRequest(pc, lock, await);
do {
    thread.wait();
    boolean[] cond = {cond₁, ..., condₙ};
    myManager.notify();
} while (state != RUNNING);
selected = myManager.getSelected();
switch (selected) {
case 1: stmt₁; break;
    ....
case n: stmtₙ; break;
}
setProgramCounter(pc);
myManager.endRequest(pc);
```

Simulation strategy and algorithm

```java
block(label) {
    stmtₙ;
}
```

```java
setProgramCounter(pc);
myManager.addRequest(pc, constraint);
do {
    thread.wait();
} while (state != RUNNING);
stmtₙ;
setProgramCounter(pc);
myManager.endRequest(pc);
```
Simulation strategy and algorithm

June 18, 2001 9

Simulation strategy and algorithm

June 18, 2001 10
Current status

- works for wait
- not implemented for constraints

WAIT
(evaluate conditions)

PROCESS

MAKE REQUEST

SCHEDULERS

MANAGER

MANAGER

PROCEED

END

CHECK PRECONDITIONS
FOR AWAIT

SCH1

SCHn

RUN SCHEDULERS

SATISFY (CHECK)
CONSTRAINTS

RUN PROCESSES
SET POSTCONDITIONS

What's missing?

- Automatic translation from meta-model to simulation
  - meta-model parser, macro expansion, ...

- Support for refinement
  - generate entire new netlist and simulate

- Support for architecture
  - architectural elements are specified in the meta-model
    - processes (RTOS schedulers)
    - media (Bus)
  - architecture introduces time
    - need for synchronization
What's missing?

Two primitives for architectures

- delay(\text{annotation})
  - interface between functional elements and architectural elements
  - may carry any information, but ultimately it's time

- synch()
  - synchronize all annotations with a global time

Demo

- write(x, 10);
- read(y, 10);
- loose synching
- hard synching

June 18, 2001