System-level Co-simulation of Integrated Avionics Using Polychrony

Huafeng Yu
Espresso, INRIA Rennes - Bretagne Atlantique / IRISA
The CESAR project

Innovations in

- Requirements engineering
- Formalization of multi viewpoint, multi criteria and multi level requirements
- Component-based engineering
- Application on design space exploration

Reference Technology Platform (RTP)

- Integration or interoperation of existing or emerging technologies

Tools demonstration

- Pilot applications: avionics, automotive, rail, ...
- Tools and tool chain demonstration
- Tools (technologies) integration via Eclipse-based ModelBus
A350 Doors Management System

Flight control systems
- Landing gear system
- Doors management system
  - passenger doors, emergency exits, cargo doors
- Flight warning system
Simplified Doors and Slides Control System (SDSCS)

- Monitor doors status via door sensors
- Control flight lock actuators
- Manage the residual pressure
- Inhibit incorrect cabin pressurization
A350 Doors Management System

Simplified Doors and Slides Control System

IMA (Integrated Modular Avionics)

- Core Processing Input/Output Modules (CPIOM)
- Remote Data Concentrators (RDC)
- Aircraft Full Duplex (AFDX)
- door sensors
- door actuators
- Landing gear system (LGS)
- Differential pressure system (DPS)
- Overflow Valve Control Unit (OCU)
- Flight Warning system (FWS)

A safety-critical system

- High-level modeling
- Early phase validation & verification
- Architecture exploration
The Polychrony approach for CESAR
Architectural modeling based on AADL
Functional modeling based on Simulink/Gene-Auto
Additional models and system integration
VCD-based simulation and profiling
Distribution and scheduling via Syndex

AADL: Architecture Analysis & Design Language
VCD: Value Change Dump
The Polychrony approach for CESAR
The Polychrony approach

Polychrony for CESAR

- Timing analysis
- Formal verification, simulation, synthesis, etc.
- (Partial) specifications
- Incomplete system description
- Parallel development
- GALS design
- Eclipse Integration in the framework of MDE
  - SME/Polychrony
  - Tools connectivity

GALS: Globally Asynchronous Locally Synchronous
MDE: Model-Driven Engineering
SME: Signal Meta under Eclipse
The Polychrony approach

Simulink/Gene-Auto functional model

AADL architectural model

SME model

Signal

C or Java

Simulation

Verification

A simplified view of design process
The Polychrony approach

SME Platform

- SME model
- Simulink/Gene-Auto Functional model
- AADL architectural model

SIGNAL Toolbox

- Compilation
- Code distribution

Polychrony

Fiacre

- XML model

SIGNAL process

C, C++

GCC

Analysis

- Sigali
- Compilation
- Code distribution

GCC

- Signal Library for AADL

C, C++

Test cases

Scheduling

Syndex

- Binaries
- VCD files

Simulation

Polychrony

Design

- Simulink/Gene-Auto Functional model
- AADL architectural model

AADL architectural model

- Compilation
- Code distribution

Polychrony

Scheduling
Architectural modeling
in AADL
Architecture modeling in AADL

AADL (Architecture Analysis and Design Language)

- SAE (Society of Automotive Engineers) standard
- High-level architecture design and evaluation for embedded systems
- Component-based paradigm
- AADL components
  - application software (*process, thread, thread group, subprogram, and data*)
  - execution platform (*processor, memory, device, and bus*)
  - composite (*system, etc.*)

ARINC 653 (Avionics Application Standard Software Interface)

- An API for software of avionics, following the IMA architecture
- APEX (APplication EXecutive) for space and time partitioning
- An ARINC partition is a logical allocation unit
Architecture modeling based on AADL
Architecture modeling in AADL

A complete AADL transformation chain

- AADL textual model
- AADL Ecore model
- SME
- Signal
- C / Java

AADL modeling

- textural AADL model

AADL Ecore model

Polychrony framework

- SME
- Signal

model transformation
Functional modeling in Simulink/Gene-Auto
Functional modeling in Simulink/Gene-Auto

Simulink and Gene-Auto
- Matlab Simulink and Stateflow: wide-spread high-level modeling languages
- Gene-Auto: a safe subset of Simulink/Stateflow for ES design

Synchronous semantics of Gene-Auto
- Logical time
- Synchronized data-flow

A complete transformation chain
Functional modeling in Simulink/Gene-Auto

Simulink point of view of SDSCS
The door handler block
Additional models and system integration
Additional models and system integration

Additional models for **open** system simulation

- Scheduler
  - A simple and static scheduler without preemption
  - Time interval is abstracted
- Simulation clocks
  - Reference clocks
  - Period clocks (for periodical threads)

Additional models for **“almost” closed** system simulation

- A simple environment model
  - Representation of the system outside SDSCS and pilot commands
- System inputs (pilot commands: take_off, open_door, close_door, land)
Simulation
Simulation

VCD visualization

- Traces (changed values) recorded in VCD format
- Global synchronization clock
- Interactive or non-interactive mode
Simulation

Profiling

- Temporal properties
- Temporal homomorphism
- Co-simulation

![Diagram showing simulation process]

Inputs

SDSCS

T(SDSCS)

Observer

Date

date_I

date_O
Distribution and scheduling via Syndex
Distribution and scheduling via Syndex

**Syndex**
- Algorithm, architecture, and adequation
- Heuristic algorithm for adequation
- Automatic code distribution
- Processor-level scheduling and communication
- Synchronization, ...

**Signal to Syndex**
- Endochronous programs transformation
- Algorithm is translated from Signal programs automatically
- Architecture is translated from AADL manually
- Constraints are added for specific binding between software and hardware
Distribution and scheduling via Syndex

Algorithm

Architecture
Distribution and scheduling via Syndex

Syndex simulation results
Conclusion and perspective
Conclusion

- High-level functional and architectural design
- High-level modeling with AADL and Simulink/Gene-Auto
- Polychrony as a common development platform
- Formal polychronous model
- Automatic model transformations
- Good interoperability between tools
- Early phase co-simulation
- Demonstration by VCD viewers
- Profiling
- Syndex
Conclusion and perspective

Perspective

- More simulation with timing analysis
- Sophisticated schedulers, such as Syndex and OS-level scheduler
- Clock constraints in MARTE/CCSL
- RT-Builder

Architecture exploration

- Performance, energy, flexibility, etc.

Formal verification, synthesis, fault modeling and analysis

- Sigali, Fiacre, Altarica, etc.

Automatic test case generation

- GATeL, TGV, etc.
Thank you for your attention