



New Developments in Model-Integrated Development of High-Confidence Software

Joe Porter, Graham Hemingway, Nicholas Kottenstette,
Harmon Nine, Chris vanBuskirk,
Gabor Karsai, and Janos Sztipanovits

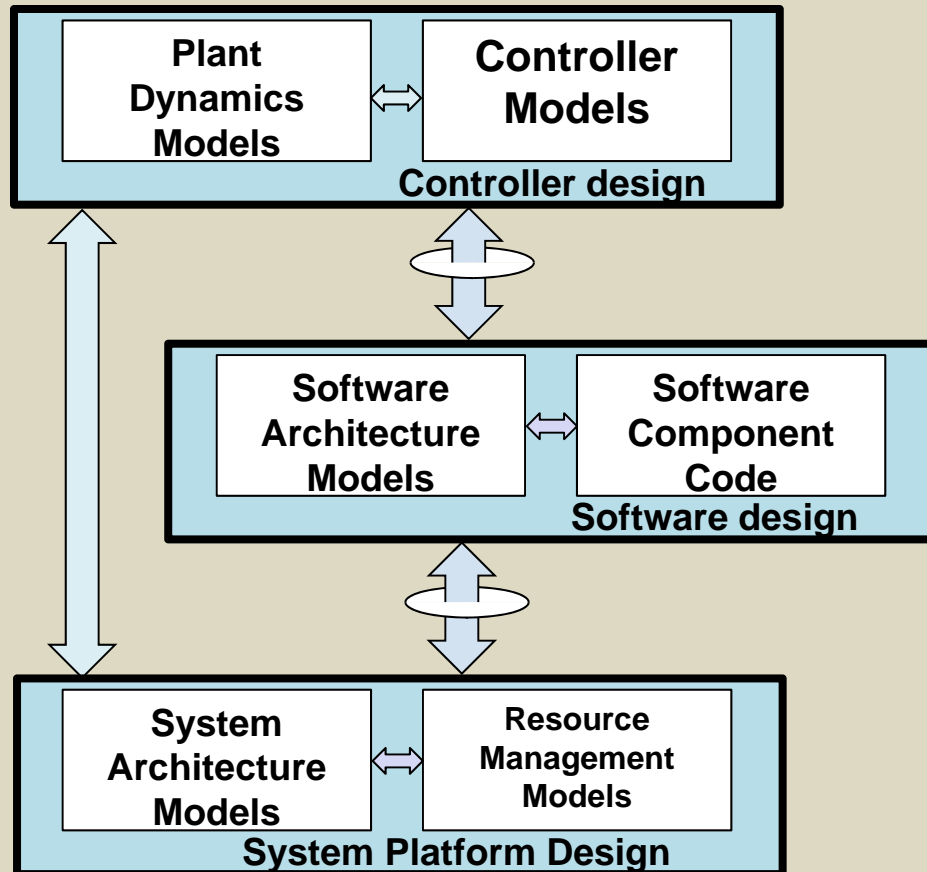
Institute for Software Integrated Systems
Vanderbilt University
Nashville, TN 37205



Big Picture: High-Confidence Embedded Software Design



Layered Design Concerns



Our Goals

Certifiable implementations

- Model-based Design Flows (GME)
- Domain-Specificity (ESMoL)
- Support certification processes

Integrating verification and validation into tools

- Concurrency/Deadlock
- Schedulability
- Value domain issues

Exploring compositional techniques

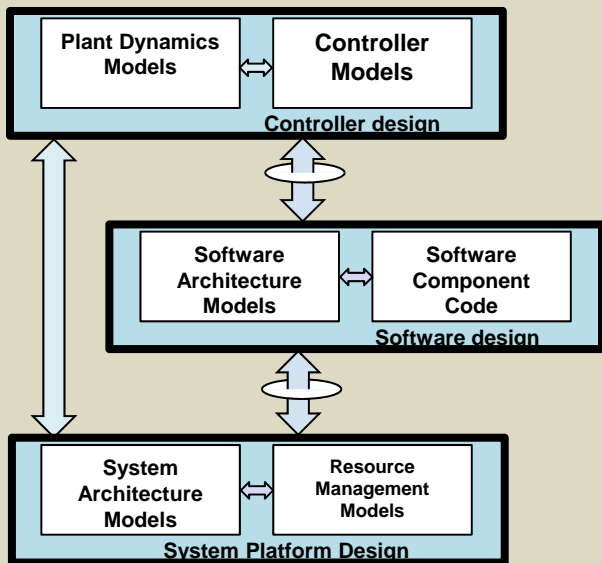
- Emphasize correct-by-construction
- Passive control design
- Verification
- Scheduling



Big Picture: High-Confidence Embedded Software Design



Layered Design Concerns



AFOSR HCDDDES MURI Team

The screenshot shows the main page of the HCDDDES MURI team website. The page features the ISIS logo, the Vanderbilt University logo, and logos of partner institutions: The University of California, Air Force Office of Scientific Research, The Ohio State University, and Carnegie Mellon University. The page title is "Main Page" and the content includes:

- Frameworks and Tools for High-Confidence Design of Adaptive, Distributed Embedded Control Systems**
- MURI Project on High-Confidence Design for Distributed Embedded Systems**
- Vanderbilt:** Graham Hemingway, Gabor Karsai, Nicholas Kottenstette, Harmon Nine, Joe Porter, [Janos Sztipanovits](#) (PI), Chris vanBuskirk and [alumni](#)
- UC Berkeley:** Claire Tomlin (Co-PI), [Edward Lee](#), George Necula, Shankar Sastry
- CMU:** Bruce Krogh (Co-PI), [Edmund Clarke](#)
- Stanford:** Stephen Boyd (Co-PI)

Navigation links include: Main Page, Collaborations, Current events, Meetings/Workshops, Publications, Project Documents, Tools, and Help. A search bar and a toolbox with links like "What links here" and "Upload file" are also visible.

<https://wiki.isis.vanderbilt.edu/hcddes>

Sponsored by the AFOSR MURI
FA9550-06-0312
and by NSF CPS contract
NSF-CCF-0820088



Development Workflows for Modeling Tools



CONTROL
DESIGN

SOFTWARE
IMPLEMENTATION

SOFTWARE
ANALYSIS

GENERATION
& EXECUTION

Simulink
Simulation

Software Modeling

Scheduling

Platform/HIL
Simulation

Requirements

Platform Design

Deadlock

Testing

High-confidence development involves many activities in different domains of expertise.



Workflow: Control Design

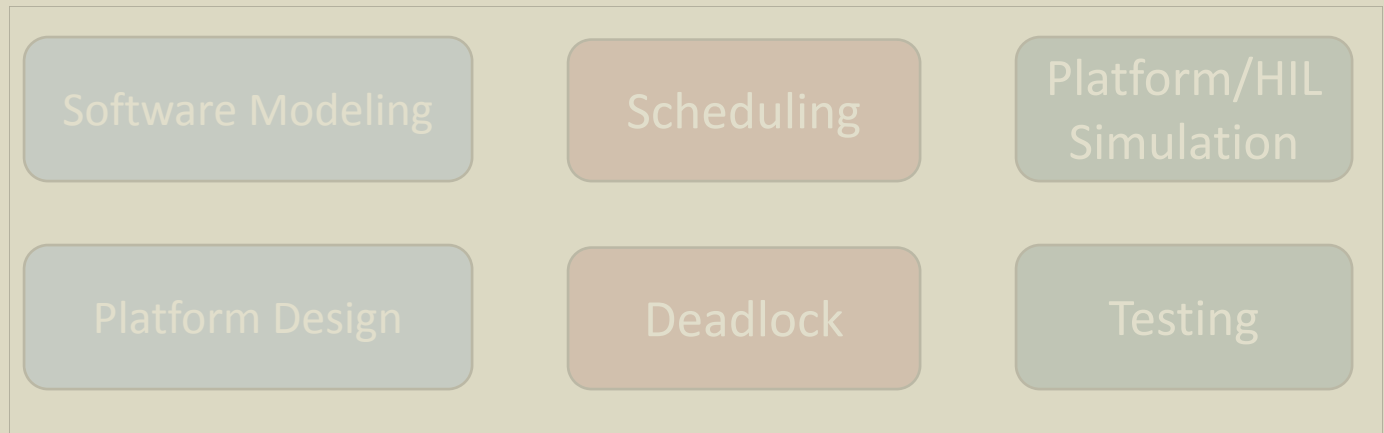
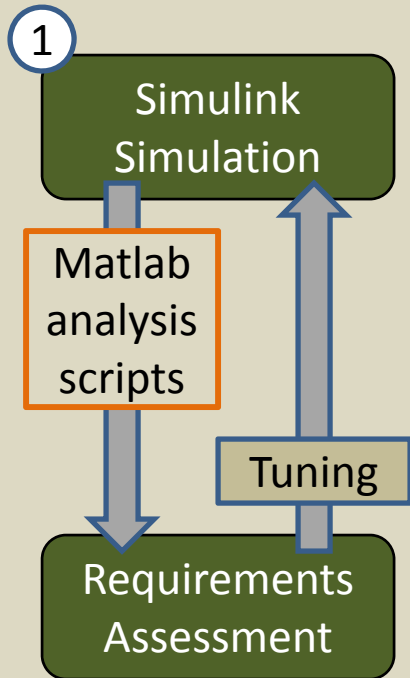


CONTROL
DESIGN

SOFTWARE
IMPLEMENTATION

SOFTWARE
ANALYSIS

GENERATION
& EXECUTION



Control designers create Simulink and Stateflow models to capture and simulate the physical behavior as well as the engineering design. Design verification takes the form of scripts to assess controller performance (e.g. stability, settling time, overshoot) and adjust controller gains.

Workflow: Import Control Design to ESMoL



CONTROL DESIGN

SOFTWARE IMPLEMENTATION

SOFTWARE ANALYSIS

GENERATION & EXECUTION

Software Modeling (Arch/Deployment)

Scheduling

Platform/HIL Simulation

Deadlock

Testing

Simulink Model Files (.mdl)

2

Importer

ESMoL Modeling Language

Requirements

Platform Design

The ESMoL domain-specific modeling language (DSML) includes a sublanguage which fully represents Simulink and Stateflow model structures. The tools include a fully automated model importer.



Workflow: Software and Hardware Design



CONTROL DESIGN

SOFTWARE IMPLEMENTATION

SOFTWARE ANALYSIS

GENERATION & EXECUTION

Simulink Simulation

Requirements

Software Modeling (Arch/Deployment)

3

ESMoL Modeling Language

Platform Design

Scheduling

Deadlock

Platform/HIL Simulation

Testing

Software and hardware designers manually enter software designs in GME to describe the software architecture of the Simulink design models, network topology, and deployment of the software components to the hardware.



Workflow: Software Analysis



CONTROL DESIGN

SOFTWARE IMPLEMENTATION

SOFTWARE ANALYSIS

GENERATION & EXECUTION

Simulink Simulation

Requirements

Software Modeling (Arch/Deployment)

Scheduling

Platform/HIL Simulation

Testing

4

ESMoL Modeling Language

Model Transformations

Sched Spec

Release Times

BIP Spec

Esched Tool

DFinder

Round-trip analysis is integrated into the tool environment.

Platform Design

Deadlock

Model interpreters generate scheduling specifications and import results. BIP analysis is under development.



Workflow: Generation & Execution



CONTROL DESIGN

SOFTWARE IMPLEMENTATION

SOFTWARE ANALYSIS

GENERATION & EXECUTION

Simulink Simulation

Requirements

Model interpreters synthesize C code for controller functions and for platform-specific task/messaging wrappers.

Software Modeling (Arch/Deployment)

5

ESMoL Modeling Language

Software Generator

Platform Design

A platform-independent time-triggered virtual machine provides a synchronous distributed execution environment.

TrueTime provides platform-specific simulation, and the xPC target enables hardware-in-the-loop.

Platform/HIL Simulation

Control Functions

Task/Msg Wrappers

FRODO VM

TrueTime (Simulink) xPC Target (HIL)

Testing



Workflow: Assessment & Refinement (in progress)



CONTROL
DESIGN

SOFTWARE
IMPLEMENTATION

SOFTWARE
ANALYSIS

GENERATION
& EXECUTION

Control designers can use the same tests to assess controller stability and performance, closing the loop on the design flow.

In the TrueTime and HIL execution environments we can measure the effects of platform uncertainty on controller performance.

Platform/HIL
Simulation

Control Functions

Task/Msg
Wrappers

FRODO VM

TrueTime (Simulink)
xPC Target (HIL)

Testing

Requirements
Assessment

Matlab analysis scripts

Tuning

6

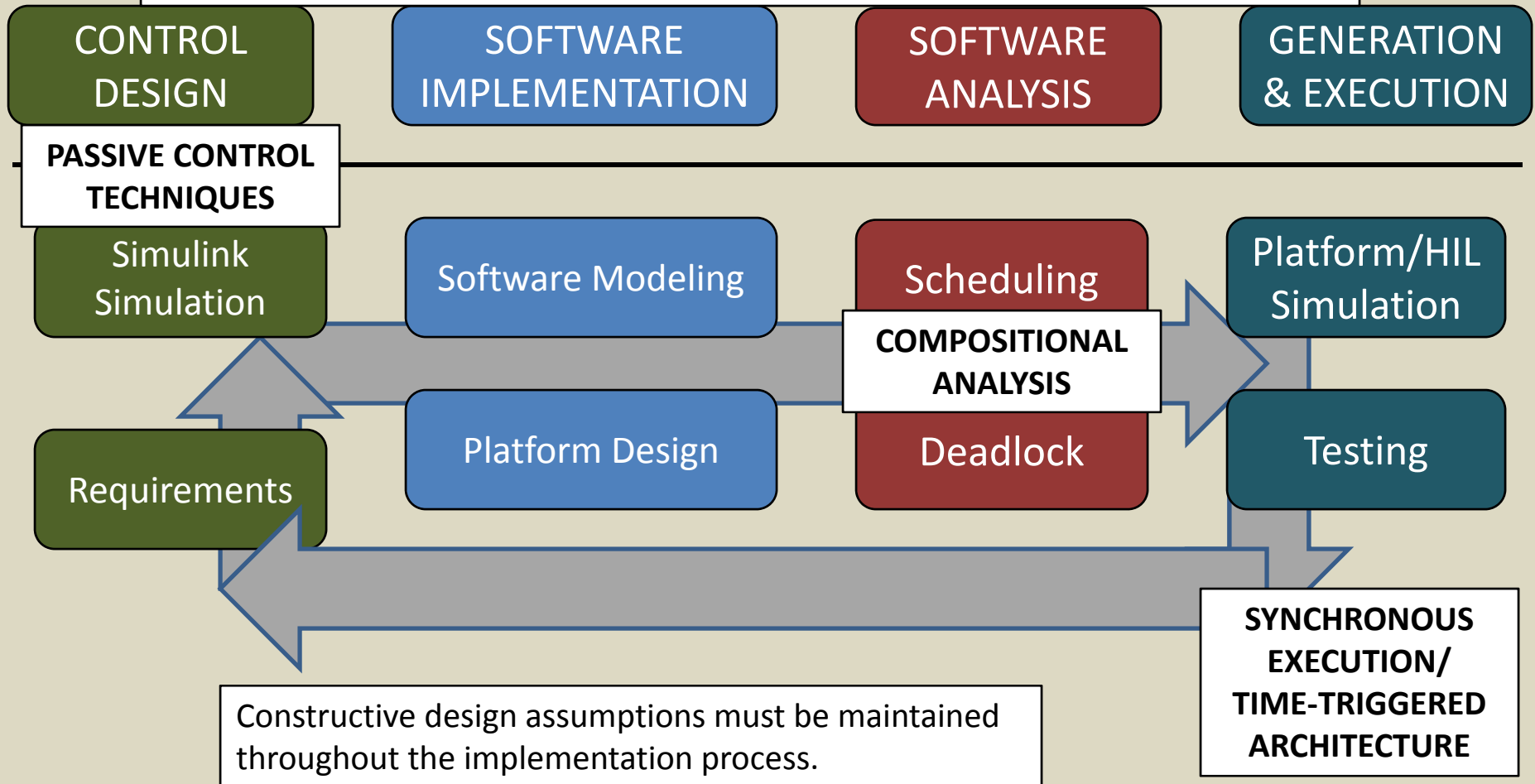




Workflow: Constructive Methods



Constructive methods rely on component behavior conditions and interconnection rules to guarantee correct behavior, reducing verification burdens.

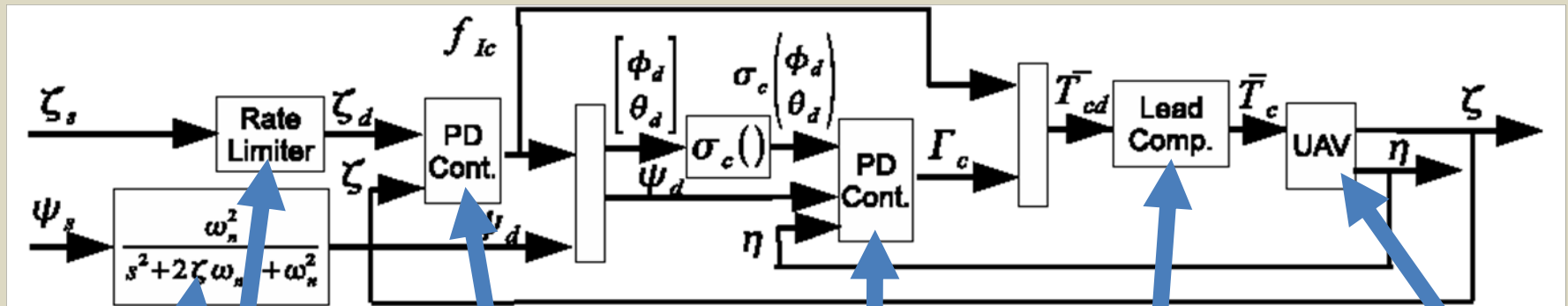




Quadrotor Design Example



Quadrotor Control Architecture



Trajectory Generator

Outer Loop Controller (Inertial Position)

Inner Loop Controller (Attitude)

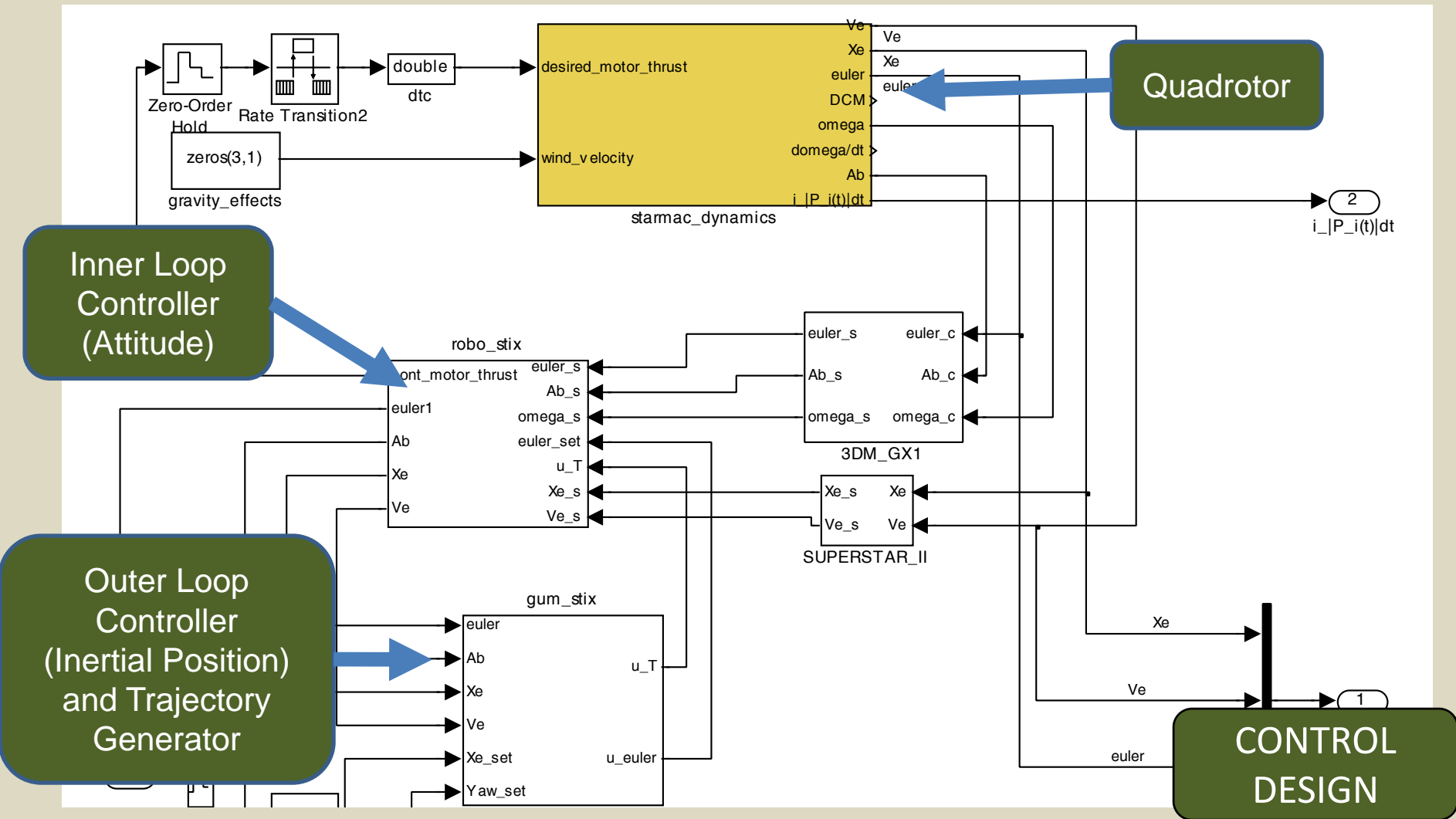
Motor Compensator

Quadrotor

CONTROL DESIGN

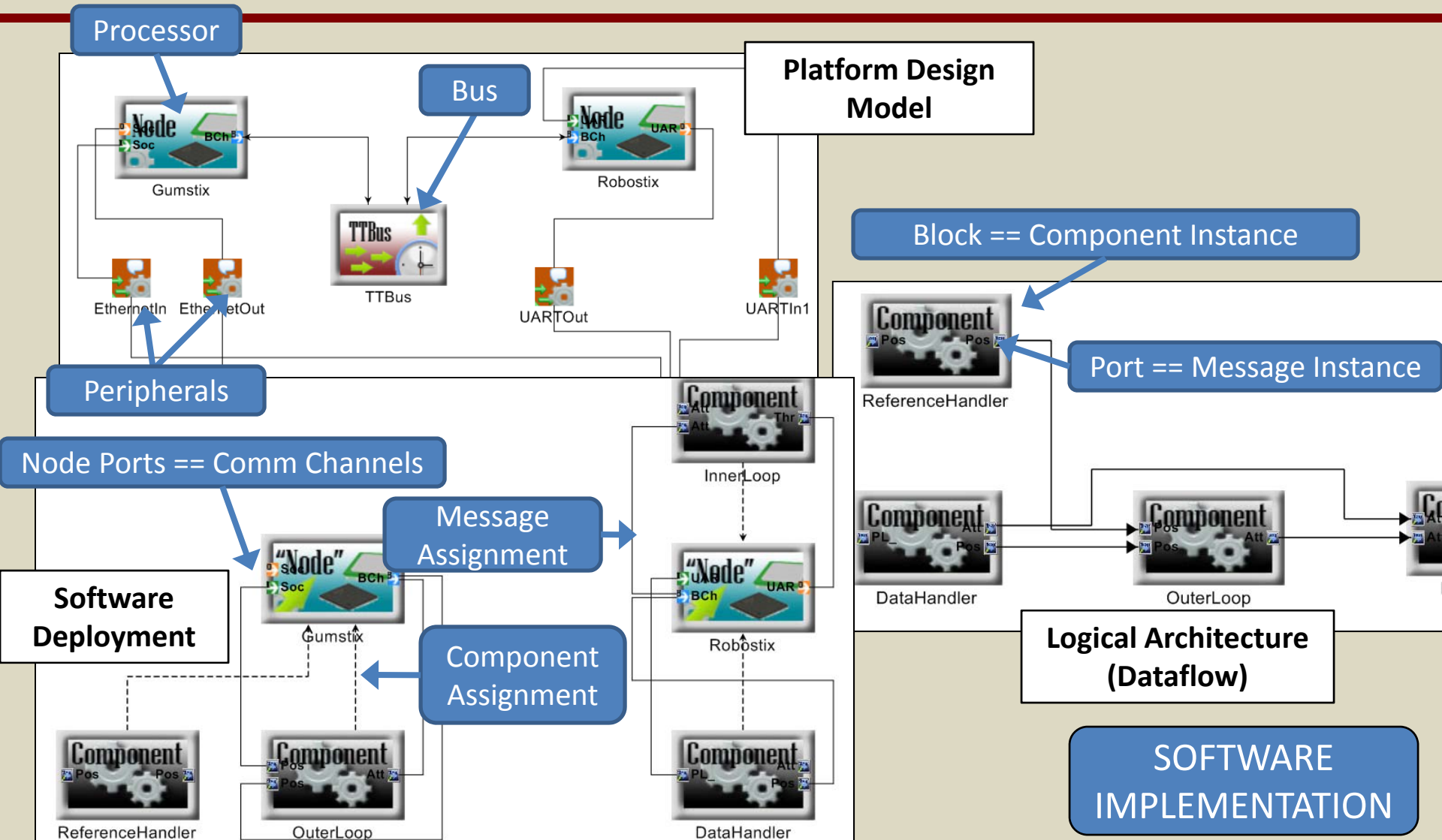


Quadrotor: Simulink





Quadrotor Software Design: GME & ESMoL

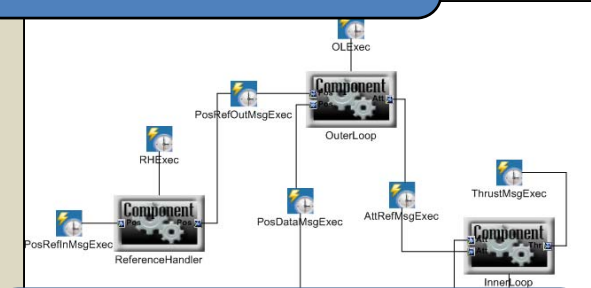




Quadrotor Platform Simulation with TrueTime



SOFTWARE IMPLEMENTATION



Component Timing Parameters:

- TTSchedule – start times
- ExecPeriod
- WC Duration

OLExec	
Attributes	Preferences
TTSchedule	0.015
ExecPeriod	20ms
WCduration	1.5ms

SOFTWARE ANALYSIS

Schedule Spec

Resolution 5us

Proc RS 4MHz 0s 0s
 Comp InnerLoop =50Hz 1ms
 Comp DataHandling =50Hz 1ms
 Comp ADC =50Hz 1us
 Comp SerialIn =50Hz 1ms
 Comp SerialOut =50Hz 1ms
 Msg DataHandling.sensor_data 8B RS/ADC RS/DataHandling
 Msg DataHandling.pos_ref 8B RS/SerialIn RS/DataHandling
 Msg InnerLoop.thrust_commands 8B RS/InnerLoop RS/SerialOut
 Msg LocalOrder 1B RS/DataHandling RS/InnerLoop

Proc GS 100MHz 0s 0s
 Comp OuterLoop =50Hz 1ms

Bus TT_I2C 100kb 1ms
 Msg OuterLoop.ang_ref 8B GS/OuterLoop RS/InnerLoop
 Msg DataHandling.pos_msg 8B RS/DataHandling GS/OuterLoop

Calculated Schedule

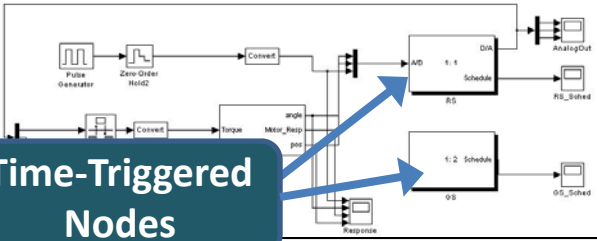
Hyperperiod 20ms

GS/OuterLoop_0 9.495
 RS/SerialIn_0 7.75
 RS/ADC_0 8.995
 RS/InnerLoop_0 9.495
 RS/DataHandling_0 10.495
 RS/SerialOut_0 11.865

TT_I2C/OuterLoop.ang_ref_0 9.175
 TT_I2C/DataHandling.pos_msg_0 10.815

GENERATION & EXECUTION

Time-Triggered Network



Time-Triggered Nodes

Generated Task Execution Code

```

in the start of a hyperperiod ...
// Determine start of current hyperperiod
kernelData->hyperperiodStart = ttCurrentTime();
}
// Otherwise we should schedule a task
else {
  // We are woken up, now schedule the task
  // Create task
  next task
  task++;
  end of hyperperiod
  nextTask = kernelData->tasks.end();
  k list pointer
  nextTask = kernelData->tasks.begin();
  hyperperiod count
  period++;
  % of here
}
}
// Determine time of the next task to be executed
double taskTime = kernelData->currentTask->first +
kernelData->hyperperiodStart;
// Sleep until that time
ttSleepUntil( taskTime );
// Micro step in time
return 0.001;
}

```

ESMoL Model File (mga)

Generator

Scheduler Spec (.scs)

Scheduler

TrueTime Simulink Model (mdl)
Task Execution Code (C)

Importer

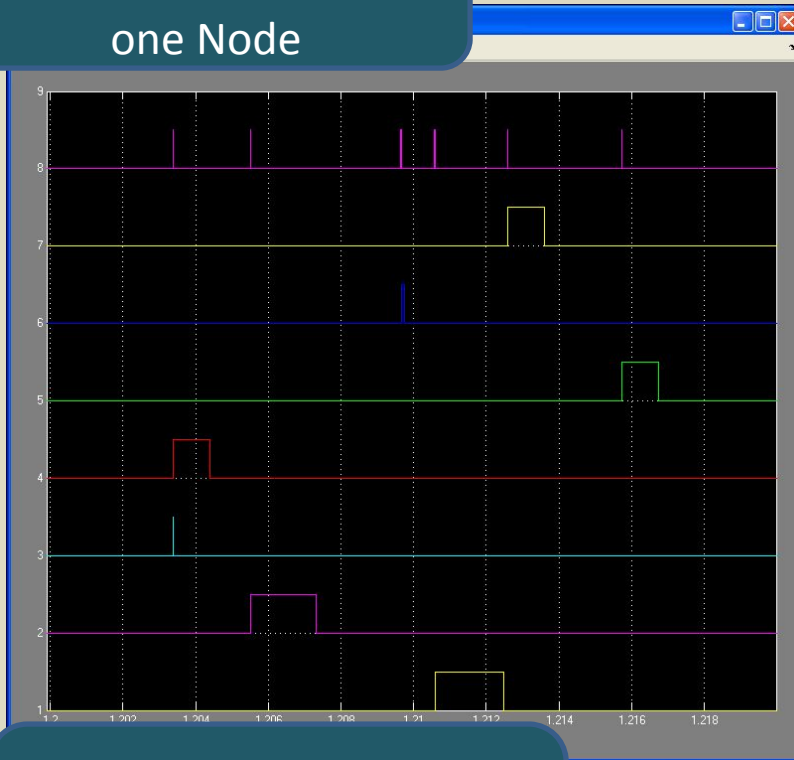
Scheduler Result (.rslt)



Quadrotor: TrueTime Execution and Schedule

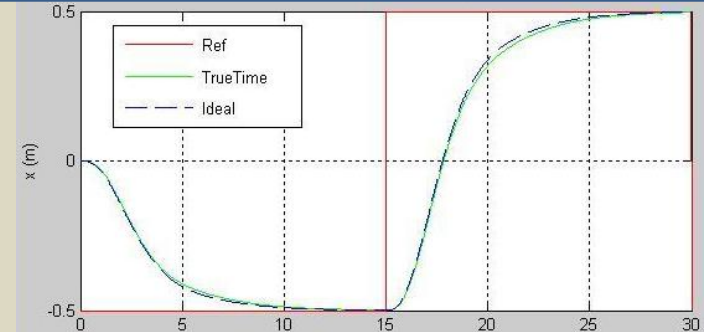


Time-triggered execution schedule for one Node

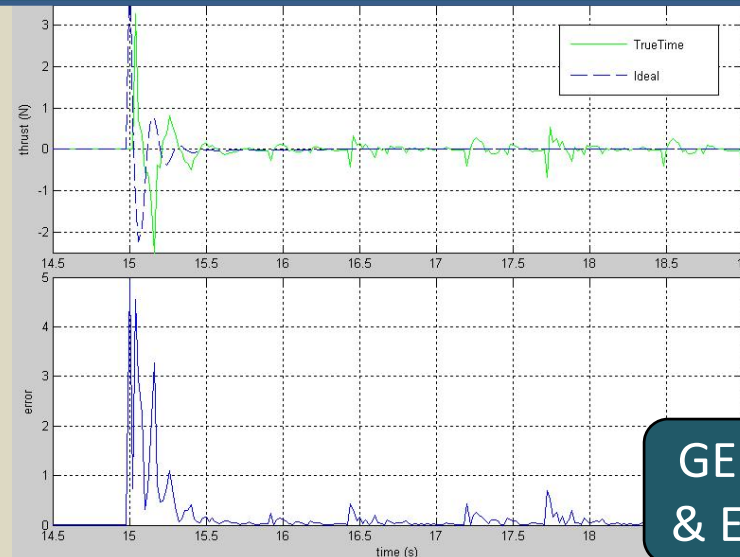


Primary execution loop is an embedded Time-Triggered scheduler

Tracking performance is close to ideal...



...but platform effects are successfully simulated



GENERATION & EXECUTION

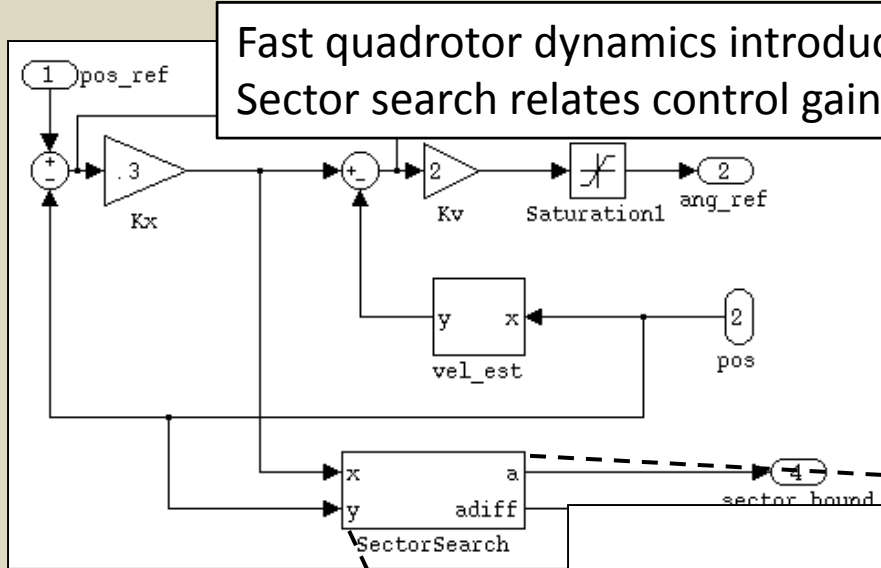


Work in Progress: Online Sector Search

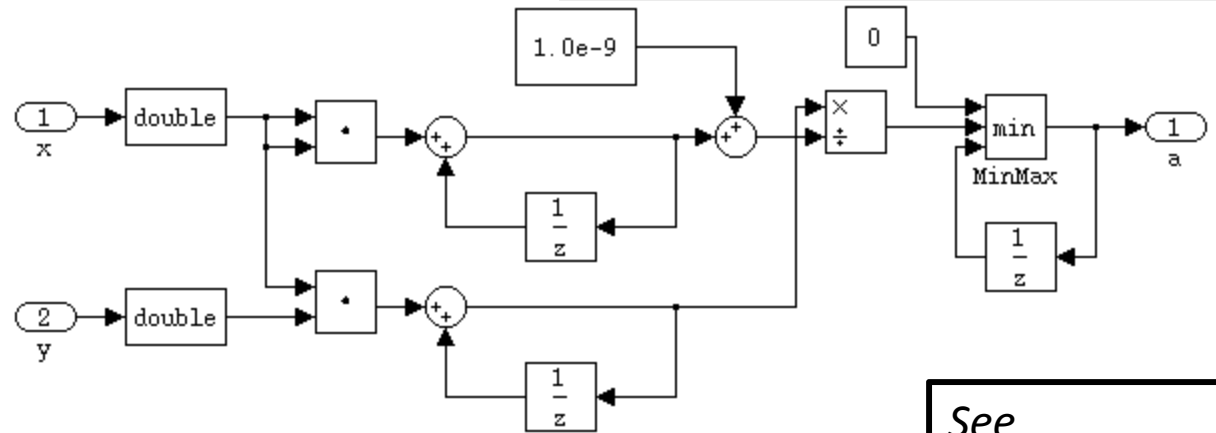


Fast quadrotor dynamics introduce a small amount of active behavior. Sector search relates control gain to an interval behavior bound abstraction.

Example: Quadrotor position tracking uses a passive PD controller, and we validate the position gain using sector search around the gain loop ($K_x < -1/a$).



$$\|y_T\|_2^2 - (a + b)\langle y, u \rangle_T + ab\|u_T\|_2^2 \leq 0$$



- CONTROL DESIGN
- SOFTWARE ANALYSIS
- GENERATION & EXECUTION

Sector search works in simulation as well as in software, so we can adjust gains for platform effects and iterate the design evaluation process.

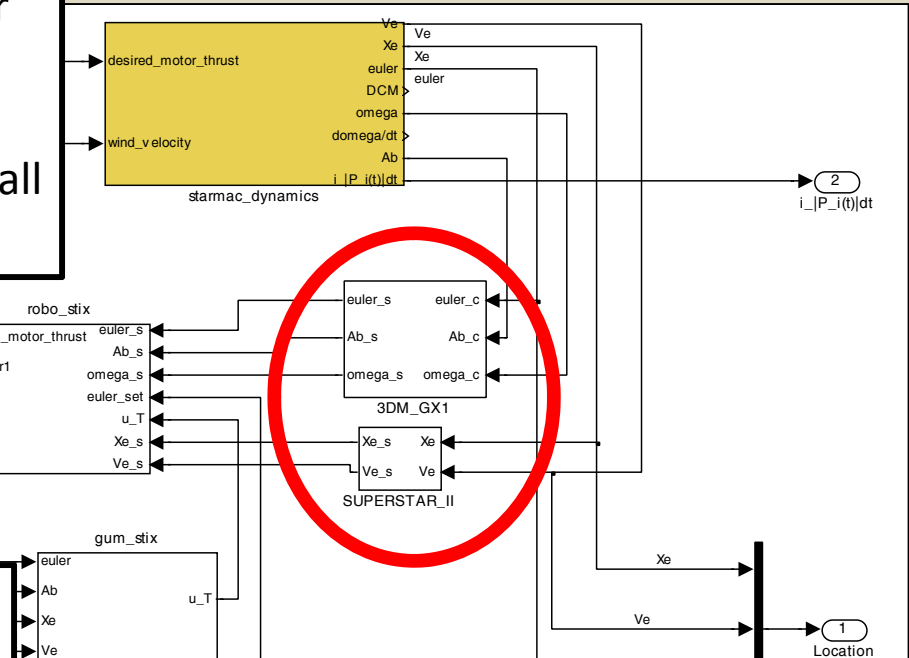
See *Kottenstette [1] for details.*



Work in Progress: Statistical Model Checking



We extended the Simulink Quadrotor model with a random (Poisson-distributed in time and duration) sensor fault. When the fault occurs, all of the sensor outputs return zero.



CONTROL
DESIGN

Requirements
Assessment

Statistical model checking sees the system as a black box, so we used an LTL error condition on the measured trajectory. "For 500 seconds of the trace, it will never be true that the error of either x, y, or z will exceed the specified bound for more than 50 seconds."

$\neg F[500] G[50] (ex > 400 \mid ey > 400 \mid ez > 100)$

How did we do? Well, it shouldn't crash more than 10% of the time under these fault conditions...

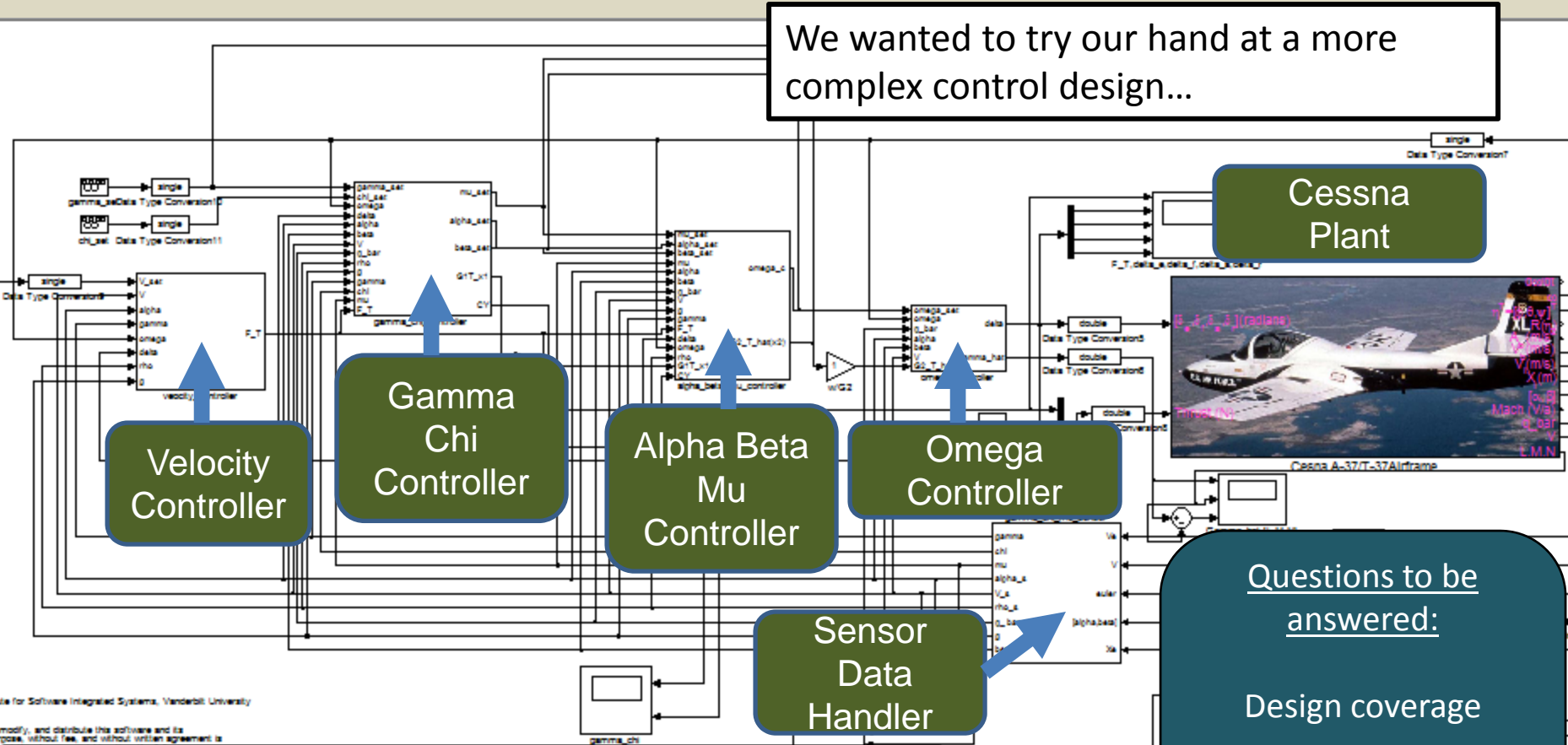
See Zuliani [5]
for details.



Work In Progress: Fixed Wing Aircraft Example



We wanted to try our hand at a more complex control design...



Velocity Controller

Gamma Chi Controller

Alpha Beta Mu Controller

Omega Controller

Sensor Data Handler

Cessna Plant

Questions to be answered:
Design coverage
Online stability assessment for backstepping designs

Multicore Experiment: Deploying multiple control loops to the Cell processor.

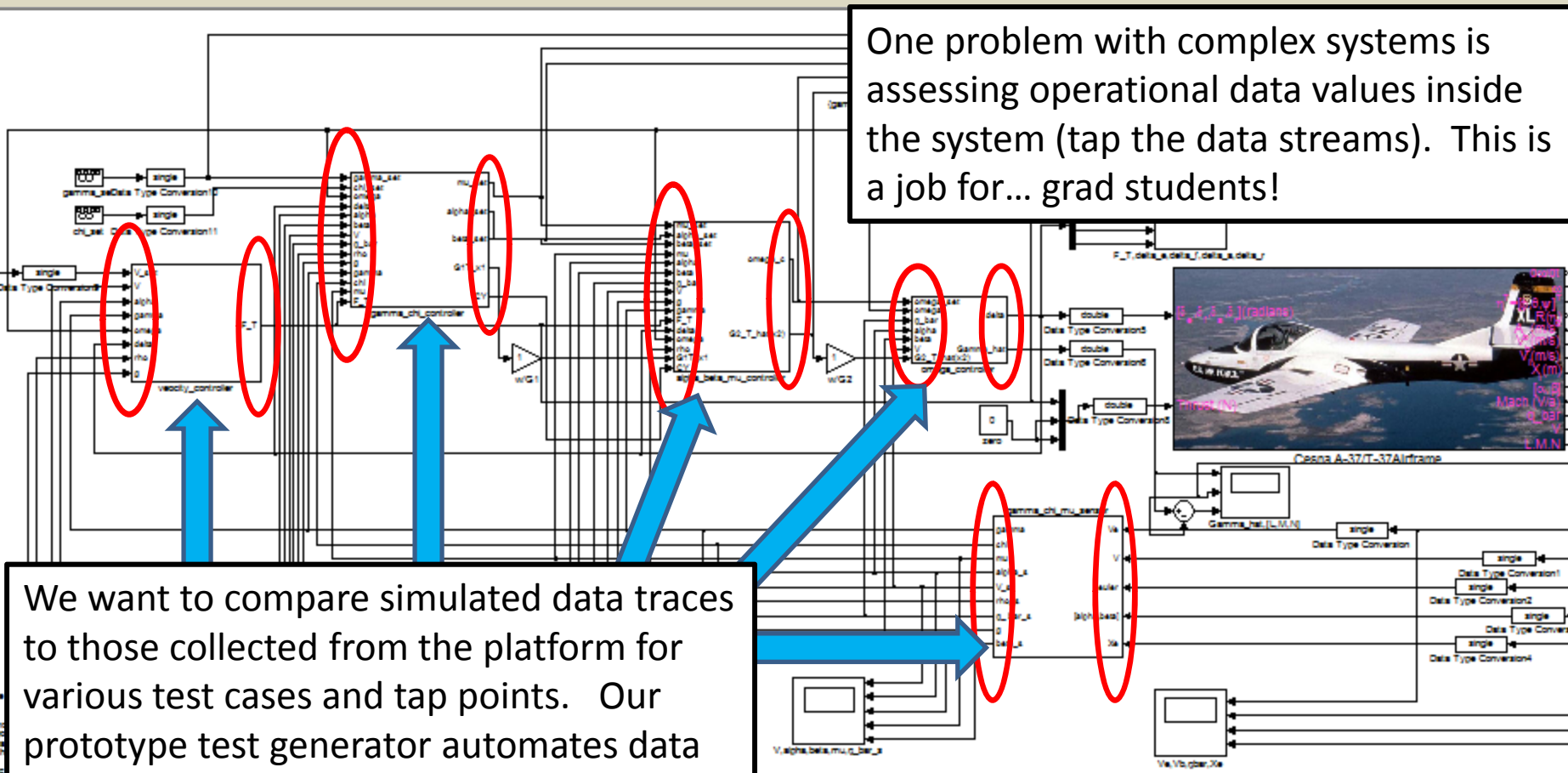
© 2012 Institute for Software Integrated Systems, Vanderbilt University
All rights reserved. No part of this document may be reproduced, stored in a retrieval system, or transmitted, in any form or by any means, without the prior written permission of the copyright holder.



Work in Progress: Test and Tap



One problem with complex systems is assessing operational data values inside the system (tap the data streams). This is a job for... grad students!



We want to compare simulated data traces to those collected from the platform for various test cases and tap points. Our prototype test generator automates data generation, execution, data collection, and comparison. It's still fairly limited...





Where would we like to go?



- Modeling Concepts
 - Requirements modeling and management
- Design examples
 - Coordinated flight
- Synthesis tools
 - Automated controller design (IDA-PBC for Port Hamiltonian systems)
 - Platform-specific function templates
- Modeling Semantics
 - Asynchronous execution models
 - Characterization of the interactions and conflicts between layered design domains
- Analysis
 - Reducing response time
 - Performance optimization



Questions?

Those interested in trying our tools can visit

https://wiki.isis.vanderbilt.edu/hcddes/index.php/The_ESMoL_Tool



References



- [1] N. Kottenstette, J. Porter. Digital Passive Attitude and Altitude Control Schemes for Quadrotor Aircraft. IEEE 7th Intl. Conf. on Control and Automation (ICCA 2009). Christchurch, New Zealand, Dec. 2009.
- [2] Porter, J., P. Volgyesi, N. Kottenstette, H. Nine, G. Karsai, and J. Sztipanovits, "An Experimental Model-Based Rapid Prototyping Environment for High-Confidence Embedded Software", Rapid System Prototyping (RSP'09), Paris, France, Jun. 2009.
- [3] J. Porter, G. Hemingway, C. vanBusKirk, N. Kottenstette, G. Karsai, J. Sztipanovits. Online Dynamic Stability Verification Using Sector Search. ACM Intl. Conf. on Embedded Software (EMSoft) Grenoble, Oct. 2010.
- [4] G. Hemingway, J. Porter, N. Kottenstette, H. Nine, C. vanBuskirk, G. Karsai, and J. Sztipanovits. Automated Synthesis of Time-Triggered Architecture-based TrueTime Models for Platform Effects Simulation and Analysis. Rapid Systems Prototyping (RSP), Jun. 2010.
- [5] P. Zuliani, A. Platzer, E. M. Clarke. Bayesian Statistical Model Checking with Application to Stateflow/Simulink Verification. HSCC 2010 (Hybrid Systems: Computation and Control), Apr. 12-16, 2010, Stockholm, Sweden.
- [6] LeBlanc, H., E. Eyisi, N. Kottenstette, X. Koutsoukos, and J. Sztipanovits, "A Passivity-Based Approach To Deployment In Multi-Agent Networks", Informatics in Control, Automation and Robotics (ICINCO 2010), Funchal, Madeira - Portugal, Jun. 2010.
- [7] R. Thibodeaux. The Specification and Implementation of a Model of Computation. M.S. Thesis. Vanderbilt University, May 2008.
- [8] N. Kottenstette, "Constructive Non-Linear Control Design With Applications to Quad-Rotor and Fixed-Wing Aircraft", Tech. Rpt., Inst. for Software Integrated Systems, Vanderbilt Univ., Jan. 2010. Nashville.
- [9] S. Bensalem, M. Bozga, T. Nguyen, J. Sifakis: D-Finder: A Tool for Compositional Deadlock Detection and Verification. CAV 2009: 614-619