BSC-1

Autocoding for Verifiability

Tim Smith
Vdot Santhanam

timothy.a.smith4@boeing.com – (314) 232-4417
vdot.santhanam@boeing.com – (316) 523-2014

Session Number/Name

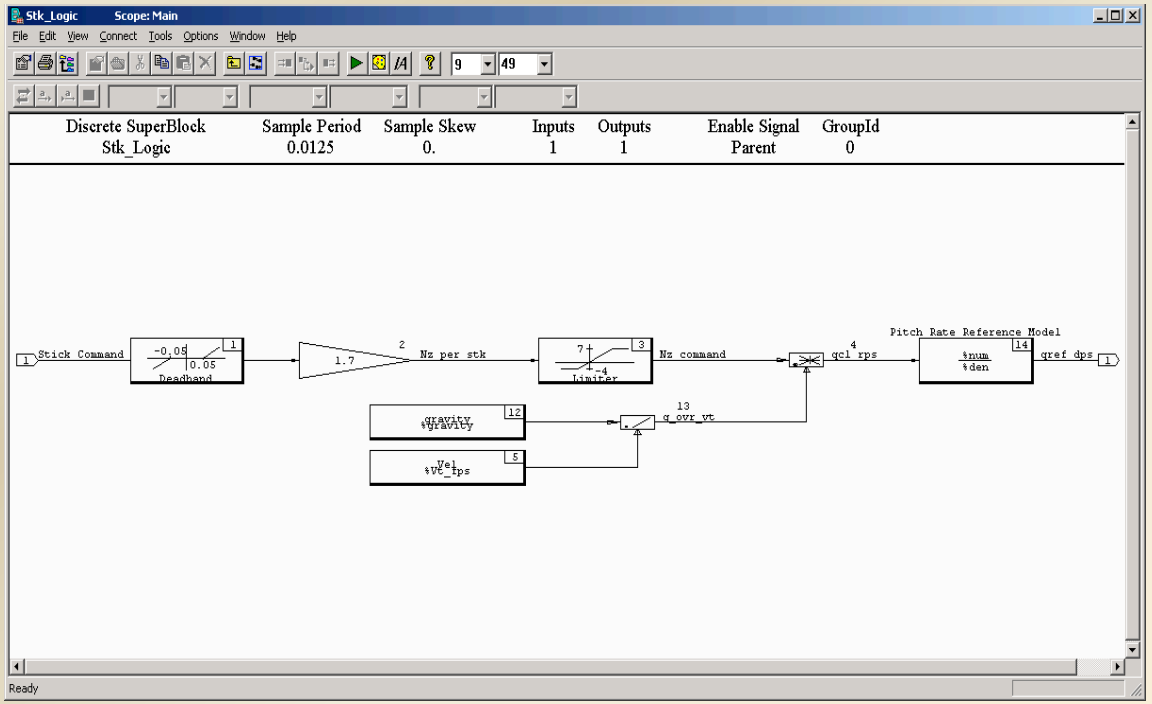
Boeing Software Conference - 1
February 6-7, 2006 – Long Beach, CA

Overview

- Model-Based Control Law Development with Automatic Code Generation
- Source Verification Issues of Automatically Generated Code
- MXZ Code Generator
- Benefits and Limitations of MXZ
- Current State of the Tool

Model-Based Control Law Development

- Cost effective
- Easily traceable to software requirements
- Automatic code generation



NASA / Boeing Software Certification Study

- Identify and address hurdles to transitioning Intelligent Flight Control System (IFCS) technology to the civil arena
- Focus was on the verifiability (DO-178B Level A) of IFCS neural/adaptive control software

MATRIXx Autocode

- It was quickly realized that the software implementation of the neural/adaptive algorithms did not present any unique verification issues.

  BUT...

- The Autocode contained many constructs that hamper source verification.

MATRIXx Autocode

- Readability
  - Package Proliferation
  - Package USE Clauses
  - Unnecessary With's
  - Frequent Use of Temporaries
  - I/O Variable Bundling
  - Prolific Use of Pointers

MATRIXx Autocode

- Testability / Traceability
  - Statement Coverage, Exception Handlers
- Type Safety
- Performance

The Zbra Subset

- Boeing developed, ‘safe subset’ of Ada 95
- Associated compiler – direct source to object mapping
- Translating MATRIXx autocode to Zbra
  - Refactoring – SAGA based tool
  - MATRIXx Template Programming Language
  - A new automatic code generation tool MXZ

Initial Attempts

- Using MATRIXx Template Programming Language
  - Unable to alter the interface convention
  - National Instruments showed no interest in modifying the internals of code generator
- Refactoring (source to source conversion)
  - Some of the clutter was eliminated
  - I/O unbundling still posed a challenge
- A new autocoder seemed easier to build

MXZ

- MATRIXx script extracts model properties
- MXZ converts properties text file to Zbra compliant Ada
- Zbra compliant code addresses the shortcomings cited above

MXZ Performance

- The study found both object code size and execution time improved on the host (Windows) platform and the 68K target

		MATRIXx	MXZ
Host (Windows, gnat compiler)	Memory	292KB	47KB
	Exec Time	73-76 µs	42-49 µs
Target (68040, Tartan Ada compiler)	Memory	42KB	17KB
	Exec Time	7.3-7.6 ms	6.7-7.2 ms

MXZ Limitations

- Only block types used in the IFCS model are currently implemented.
- Fixed point arithmetic not implemented.
- Some blocks have restrictions placed on them.
  - e.g.: Vector inputs are not permitted to Waveform
  - e.g.: Sequential execution Condition blocks are not supported

What About MATLAB?

- MXZ generates code directly from a text file containing the model properties. Independent of the application that generated the properties file.
- A *.m file that can produce the model properties text file for input to MXZ is all that is needed to bridge the gap.

What About Other Languages?

- There are no inherent limitations in MXZ that will inhibit it from being adapted to generate C code or code in any contemporary programming language.
- Zbra is a very limited subset of Ada 95 and can be readily mapped to C, FORTRAN or Java.

Disclaimer

- Many I/CRAD and production programs have used MATRIXx's Auocode capability to develop highly reliable software that functions flawlessly in deployment.
- The authors believe that MXZ has the potential to provide cost savings in the source code verification process.

Summary

- Boeing/NASA study to determine the certifiability of neural/adaptive flight control laws.
- One focus was on source code verification
- MATRIXx autocode deemed inadequate for certification to DO-178B Level A, independent of the neural architecture
- Zbra compliant code greatly improves certifiability
- MXZ automatically generates Zbra compliant code from the model properties description.

Name: Tim Smith
Key Technical Field: Systems And Flight Engineering
Phone number: 314-232-4417
Fax number: 314-232-4141
E-mail: timothy.a.smith4@boeing.com

Biography:

Tim Smith is a Guidance, Navigation and Controls engineer. In his five years with the Boeing Company, Tim has helped design and test the flight control laws for the Intelligent Flight Control System (IFCS) for the NASA NF-15B aircraft. Tim designed the pre-trained neural networks for the C-17 IFCS application. Tim recently participated in the NASA sponsored project concerning the verification and validation of adaptive software for the F-15 IFCS.

Name: Vdot Santhanam
Key Technical Field: High Integrity Software Engineering
Phone number: 316-523-2014
Fax number: 316-526-2105
E-mail: vdot.santhanam@boeing.com

Biography:

Vdot is a Technical Fellow in the area of software with emphasis on software architecture for high-integrity applications. During his eighteen years with Boeing, Vdot has supported a number of crucial programs with software technology that has led to significant cost savings and improved integrity. He assisted the 777 primary flight computer program with compiler selection, certification testing, code optimization and automated filtering for error-prone constructs. Vdot is the principal architect of the aerial refueling software for the 767 tanker. Vdot was the principal investigator on the NASA/FAA contract to study the tool qualification criteria for automated software verification tools and most recently led the verification and validation study of adaptive software for the F-15 Intelligent Flight Control System project, also sponsored by NASA.

Autocoding for Verifiability

BSC Session: ?

Program implementation: F-15 Intelligent Flight Control Systems, Adaptive Software Verification & Validation Study

Abstract: This presentation concerns an improved process for automatically generating source code that is more easily verifiable to DO-178B Level A standards. MXZ automatically generates a safe-subset compliant Ada source code for a model-based design directly from a text file describing the properties of the model. MXZ was prototyped on a Boeing/NASA collaborative project involving a MATRIXx/Simulink representation of the software requirements implementation. MXZ generated code showed improvements in verifiability and performance over the MATRIXx generated code.