ECU Production Code Generation

15 June 2004

Tom Erkkinen and Steve Toeppe

Simulink 6
for Model-Based Design

Agenda

- Introduction
- New Features
- Success Stories
- Process Capabilities
- Next Steps
Production Code Overview

MathWorks Production Code Focus

- Production code generation is a top corporate initiative
  - Working closely with automotive and aerospace companies
  - Real-Time Workshop® Embedded Coder generates production code
- Recent results:
  - Automatic code smaller than production hand code\(^1\)
  - One million lines of automatic code in safety-critical systems\(^2\)

\(^1\) Visteon, SAE Technical Paper 2004-01-0269, March 2004
\(^2\) Honeywell, FAA Software Tools Forum, May 2004
Agenda

- Introduction
- New Features
- Success Stories
- Process Capabilities
- Next Steps

Production Code User Requests

**Systems Engineering**

*Modeling and ECU prototyping*
- Large scale architecture
- Multiple domains
- Data management

*Software design and ECU code*
- Code efficiency
- Fixed point
- Module packaging
- Multirate systems
- Embedded targets

**Project Management**

*Processes and teams*
- Configuration management
- Interfaces
- Documentation
- Seamless tool chain

*Release 14 provides a complete ECU development environment that maximizes team efficiency*
System Architecture

- Modular Development
  - Model Referencing
  - Incremental and iterative development
  - Configurable items
- Incremental Code Generation
  - Model Blocks
  - Persistent identifiers
- Component Interface
  - Non Virtual Buses
  - C structures

*Simulink 6 has immediate code generation support*

Multiple Domain System Modeling

Systems containing:
- Sensors and Actuators
- Environment or Plant
- Application or Controller

Are naturally modeled with:
- Simulink
- Stateflow
- Signal Processing Blockset
- Embedded MATLAB Fcn
- SimMechanics
- SimPowerSystems
- Much more ...

*Simulink 6 has complete code generation support*
Multiple Domain System Code Generation

Real-Time Workshop
Embedded Coder 4
supports and extends all
Real-Time Workshop 6

One code generator
for multiple uses

Rapid Prototyping

Real-Time Workshop
Embedded Coder (ERT)

Real-Time Workshop
Module packaging

Production Code

Continuous time

Simulink Verification and Validation

- Develop requirement based test
- Validate designs
- Establish coverage
- Detect flaws
- Trace requirements

Requirements appear in code
Simulink Data Management

Model Explorer and Simulink Data Objects
- Create Data Dictionary
- Manage Simulink and Stateflow data
- Use MPT data objects for production data
- Model workspaces

>> daexplr

Code Efficiency

- R14 includes many new optimizations
  - Optimized fixed-point operations (e.g., division)
  - For-loop fusion
  - Enhanced inlining of Stateflow library charts
  - Improved expression folding
  - Improved dead-path elimination
  - Efficient implementation of absolute and elapsed time
  - And much more…
Fixed-Point Optimizations

Multiple operations combined as a single shift right.

```c
/* Outport: '<Root>/Out1' incorporates: 
* Product: '<Root>/Divide' */
rtY.Out1 = (int16_T)(rtU.In1 >> 3);
```

For-loop Fusion

- For-loops are fused inside Simulink and Stateflow
- Complements expression folding

```c
for(i1=0; i1<10; i1++) {
    if(rtU.In1[i1] * 3.0 >= 0.0) {
        rtb_SW2[i1] = rtU.In1[i1] - rtDWork.Delay_DSTATE[i1];
    } else {
        rtb_SW2[i1] = (rtDWork.Delay_DSTATE[i1] - rtU.In1[i1]) * 5.0;
    }
    rtY.Out1[i1] = rtb_SW2[i1];
    rtDWork.Delay_DSTATE[i1] = rtb_SW2[i1];
}
```
Simulink Fixed Point

- Fixed-point Blockset fully consolidated with standard Simulink library
- Unified fixed-point functionality now added to more blocks
- Discrete integrators with fixed-point types now support triggered sample times
- Bit-true simulation and code generation for multiplication and division with non-zero bias
- Optimized fixed-point code generation across products

Engineers want to use standard Simulink blocks

Module Packaging Features

For mass production, company specific needs
Based on production pilot success

Coding standards and file formats -- Unique data definitions -- Application management
Custom Storage Classes

- Create custom storage classes
- Integrate production code data

Multirate Systems

- Rate grouping: Each rate into separate function
- Target independent (priority based) rate transition block for periodic and asynchronous data transfers
- Sample time constraints for models and subsystems
- Enhanced asynchronous event support
- Production quality Absolute and Elapsed time

*Multirate is now much more efficient, clearer, and better for production*
Interfaces

- Simulink-based
  - External Mode
- Target-based
  - C-API
- Host-based
  - ASAP 2

*Simulink/Real-Time Workshop offer a synchronized model/code platform for other technologies to leverage*

Embedded Targets

- Support for any target
  - ANSI-C
  - ISO-C (optimized math)

- More Support for
  - Motorola® MPC555 (incl. 56x)
  - Motorola® HC12
  - Infineon C166® Microprocessor
  - TI C6000™ DSP
  - TI C2000™ DSP
  - OSEK/VDX®
New Demo Suite

- Full suite of demos showing seamless modeling and code generation environment

>> rtwdemos

---

Agenda

- Introduction
- New Features
- Success Stories
- Process Capabilities
- Next Steps
Real-Time Workshop Embedded Coder

**Automotive Highlights**

*Toyota and Denso*

*OEMs and suppliers in mass production*

*Visteon Powertrain*

*Automatic code smaller than hand code*

*Caterpillar*

*Single tool for rapid prototyping and production code*

*Delphi*

*Set & Forget Automatic Climate Control*

*Motorola Powertrain & Chassis Systems*

*Second-generation fixed-point use*

*Jaguar*

*In-vehicle rapid prototyping*

*Siemens VDO*

*“Excellent local support from MathWorks Germany”*

---

**ECU Production Process**

- **Define Requirements**
- **System-Level Specification**
- **Subsystem Design**
- **Subsystem Integration & Test**
- **Vehicle Integration, Test & Calibration**

**Definitions**

- **Sim**: Simulation
- **RP**: Rapid Prototyping
- **OTRP**: On-Target Rapid Prototyping
- **PCG**: Production Code Generation
- **SIL**: Software in Loop
- **PIL**: Processor in Loop
- **HIL**: Hardware in Loop

**Implementation**

- **Version control**
- **Requirements management**
- **Documentation**
Next Steps

- Production code recorded webinar series (R13)
  - [www.mathworks.com/webinars](http://www.mathworks.com/webinars)

- Attend RT04 training class (R14)
  - [www.mathworks.com/training](http://www.mathworks.com/training)

- Contact your local MathWorks representative

- Meet MathWorks developers at IAC
Conclusions

- Production code success is currently being achieved
- MathWorks Release 14 will make more customers, more successful

*MathWorks is deeply committed to satisfying your production code generation needs*