

# The MathWorks® Crossover to Model-Based Design

The Ohio State University  
Kerem Koprubasi, Ph.D. Candidate  
Mechanical Engineering

The 2008 Challenge X Competition



- Model-based design tools allowed Ohio State to:
  - Accurately define vehicle technical specifications,
  - Design and test control strategies in simulation,
  - Predict vehicle dynamic behavior,
  - Generate embedded code automatically.

**Simulink®:** Rapid construction of complex dynamic system models. Graphical environment.

**Stateflow®:** Effective implementation of state machines. Visual support makes code debugging simple.

**Real-Time Workshop®:** One-step automatic code generation directly from model. Relieves requirement for advanced programming skills.

**MATLAB® Toolboxes:** Assist in control design, system optimization, statistical data analysis, and signal processing.



# Summary of Year 4 Improvements

- **Modeling and Data Analysis**

- **CX-Sim:** Quasi-static vehicle simulator for fuel economy and performance evaluation

Verified &  
Validated

- **CX-Dyn:** Dynamic simulator for drivability assessment and control

Merged

- **CX-Trac:** Detailed tire dynamics for traction control.



Verified &  
Validated

- **CX-Start:** Specialized engine-start simulator.

- **CX-DAQ:** Data analysis and post-processing tool.



# Summary of Year 4 Improvements

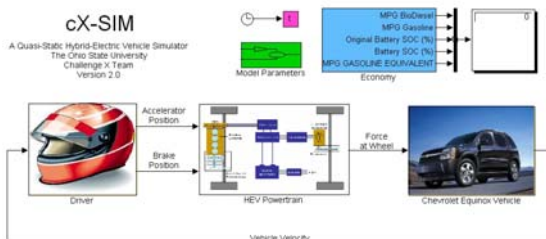
- **Control Strategy Improvements**
  - **Exhaust Aftertreatment Control:**
    - **Local regeneration control** to achieve substantial NOx emissions reduction with a low fuel economy penalty.
  - **Engine Start Control:**
    - **Model-based (LQR) strategy** to reduce speed overshoot with minimal calibration.
  - **Mode-Transition Control:**
    - **Torque blending between mode transitions** to improve drivability.
  - **Active Driveline Control:**
    - Electric motor torque **control during pedal tip-in/tip-out**.
  - **Adaptive Energy Management Strategy:**
    - **From look-up table implementation to calibratable code** using an embedded MATLAB function.



**System Design  
Concept**

**Lessons  
Learned**

## Model Based Design



## Simulation Results

- Predicted VTS
- Control Strategy Parameters

**Lessons  
Learned**

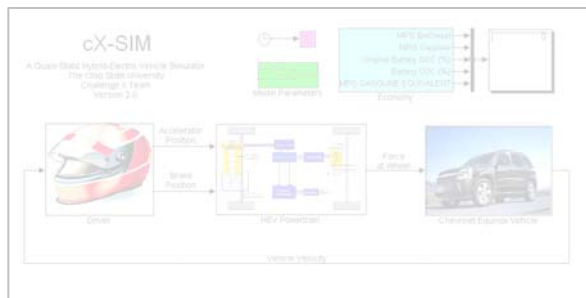
**YEAR 1**



**Lessons  
Learned**

**System Design  
Concept**

## Model Based Design



## Vehicle Integration



## Simulation Results

- Predicted VTS
- Control Strategy Parameters

**Lessons  
Learned**



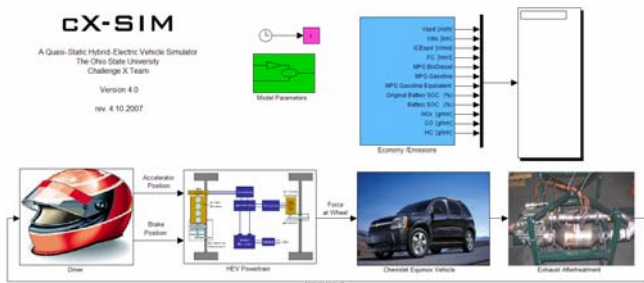
# YEAR 2



Lessons  
Learned

System Design

## Model Based Refinement



## Vehicle Integration



Simulation Results  
- Updated VTS  
- Tuned Control  
Strategy Parameters

Lessons  
Learned

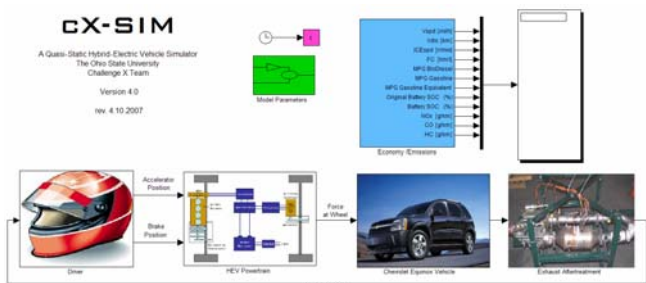
**YEAR 3**



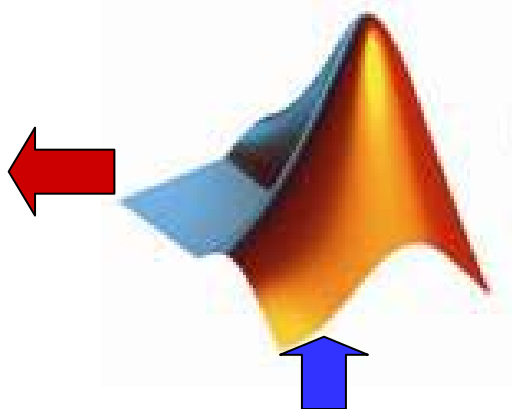
**Lessons Learned**

**System Design**

## Model Based Refinement



## Code Optimization and Verification



## In-Vehicle Results

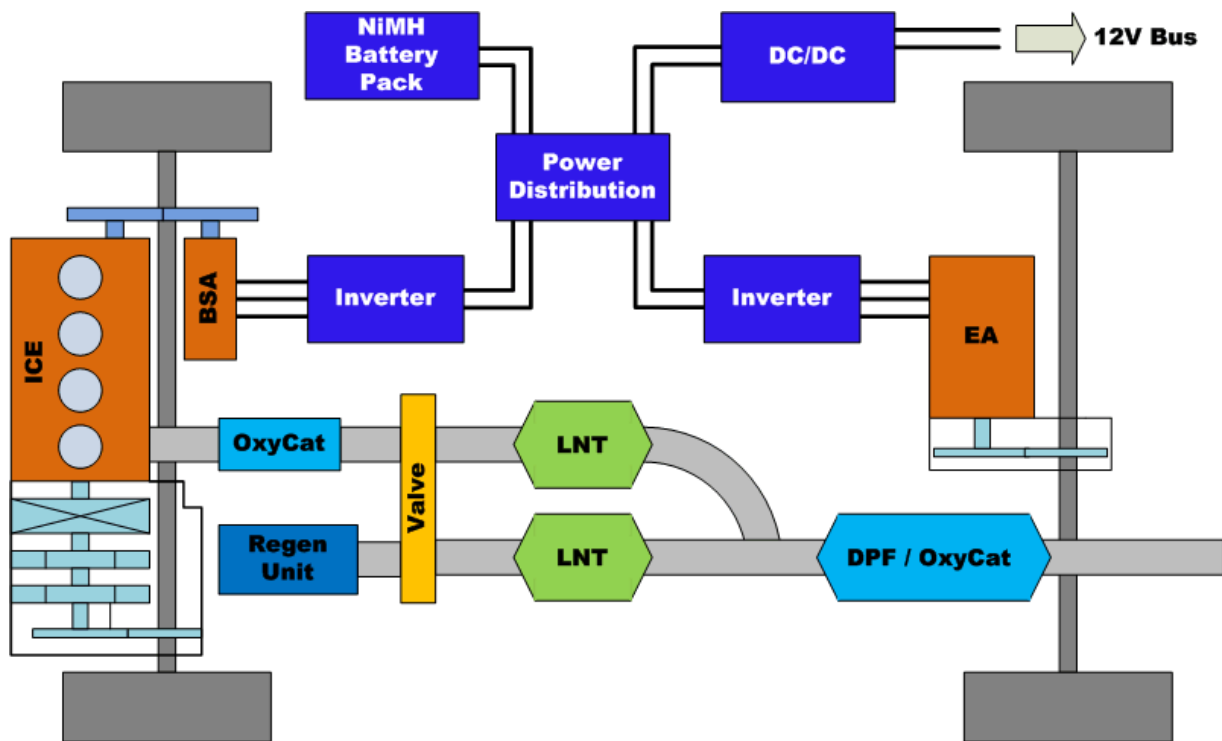
- Use of data for model calibration and validation.
- Control verification and optimization.



**YEAR 4**



# Ohio State Vehicle Architecture

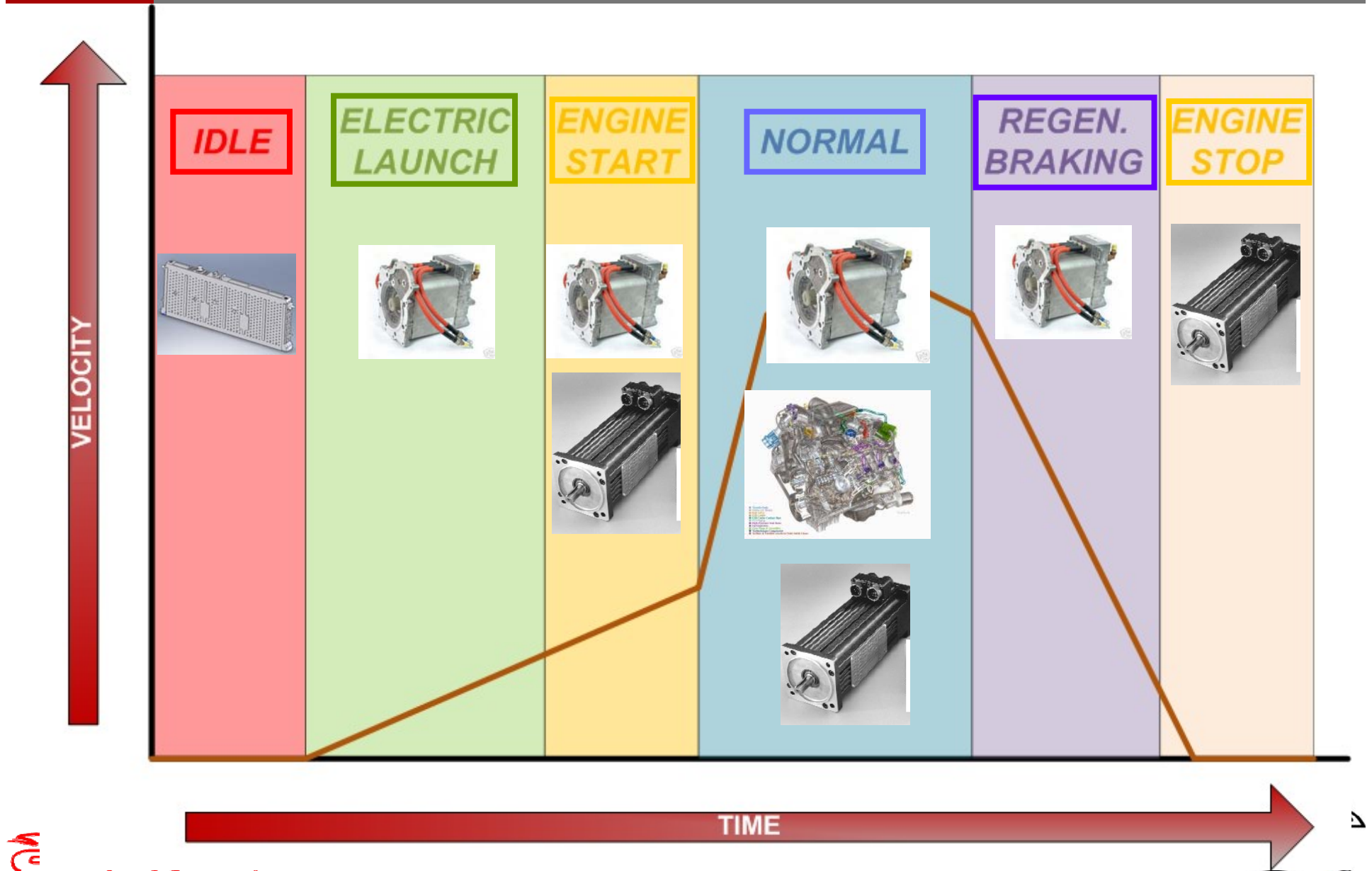


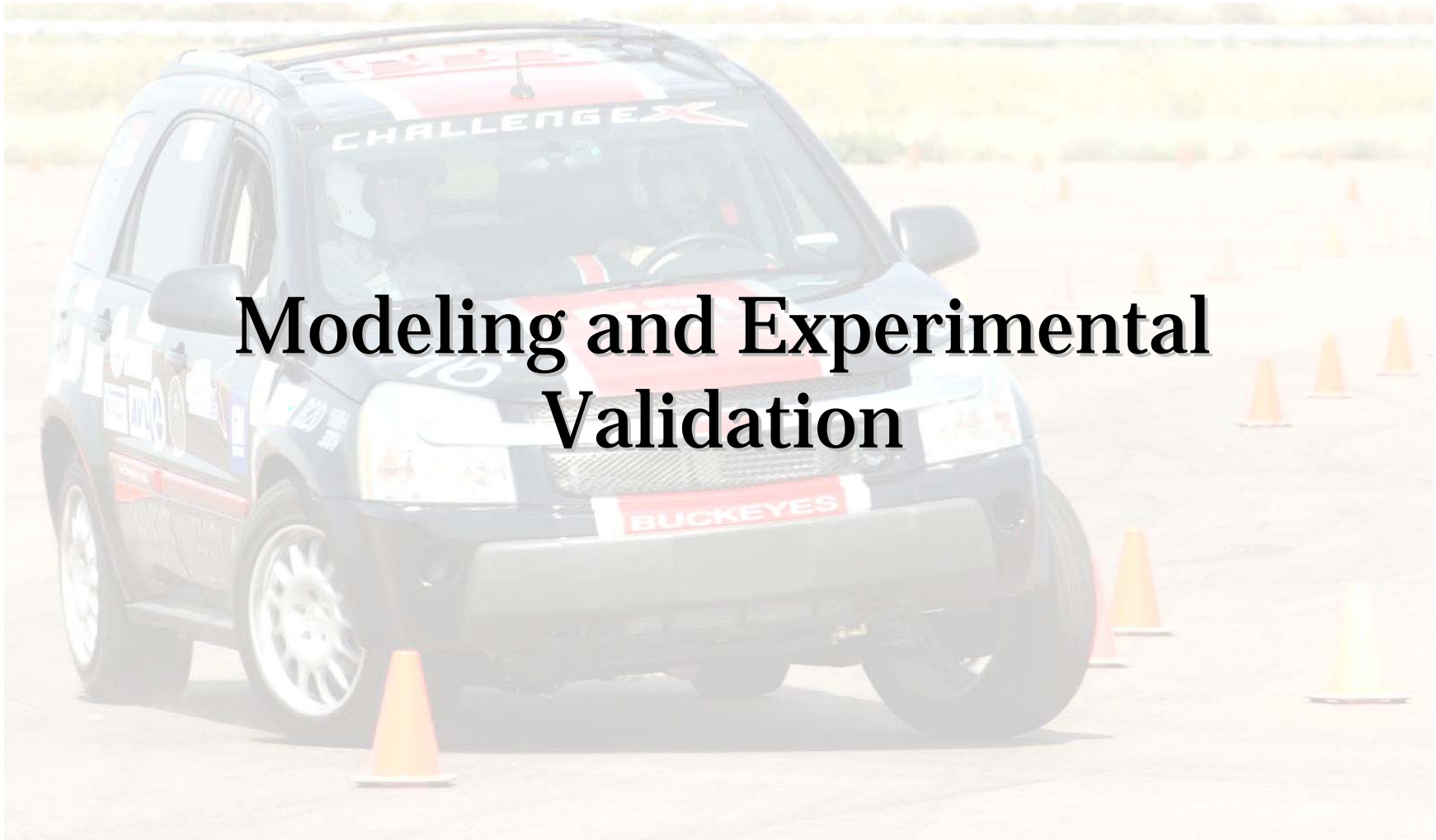
- 1.9L (110 kW) Diesel Engine (ICE)
- 6-Speed Automatic Transmission
- 32 kW Traction Electric Machine (EA)
- 10 kW Belted Starter Alternator (BSA)

- 300V Nominal Ni-MH Battery Pack
- Dual LNT Exhaust After-treatment System
- DC/DC Converter for 12V accessories



# Hybrid-Electric Vehicle Modes

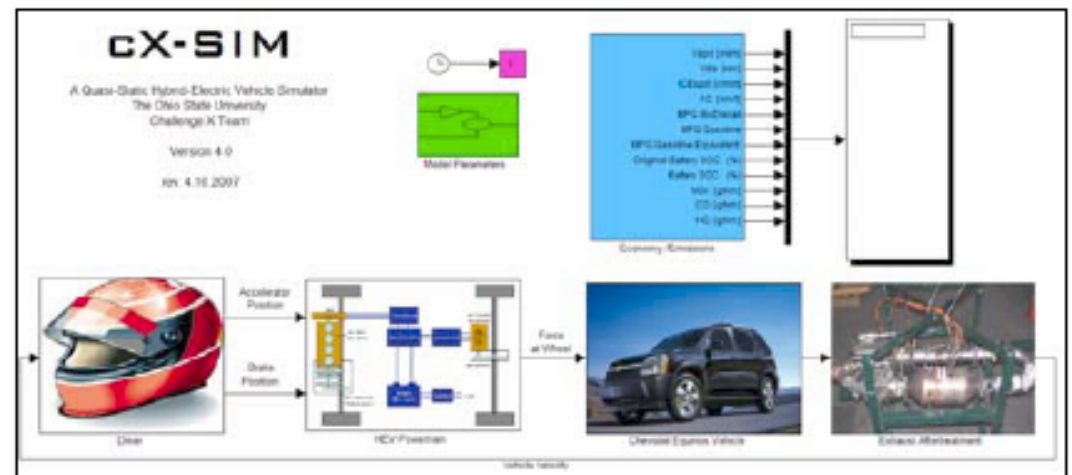




# Modeling and Experimental Validation

## CX-SIM

- Quasi-Static Vehicle Simulator
- Fuel Economy and Basic Performance Evaluation
- Control Strategy Development and Tuning



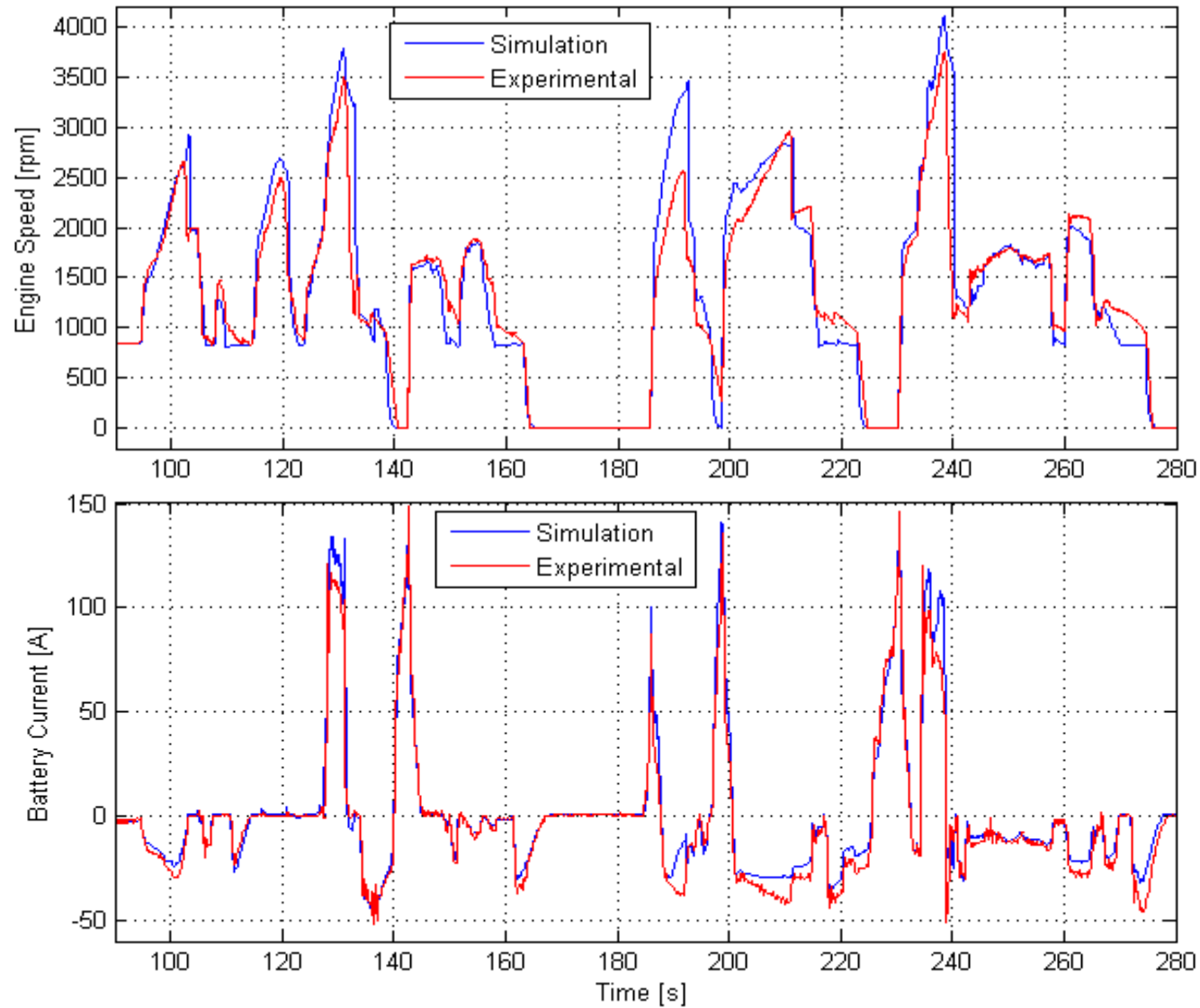
## Year 4 Improvements:

- Model is validated (no driver feedback) with experimental data.

Control Commands In → Vehicle States Out → Compare with Data

- Battery, engine and torque converter models are revised as a result of the experimental validation.

# CX-SIM: Experimental Validation



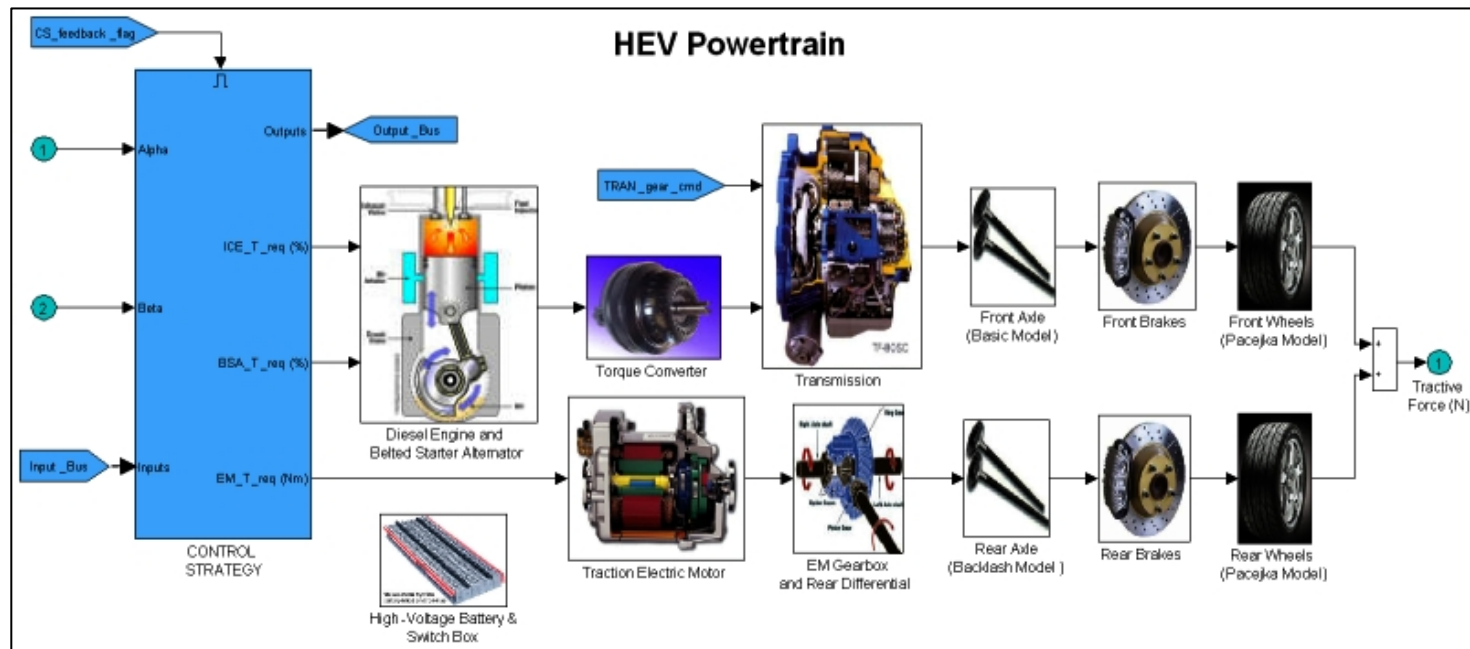
# CX-DYN: Dynamic Drivability Model

## CX-DYN

- Low frequency dynamic model suited for drivability and fuel economy evaluation.
- Gear shift transients, driveline shuffle, actuator delays.

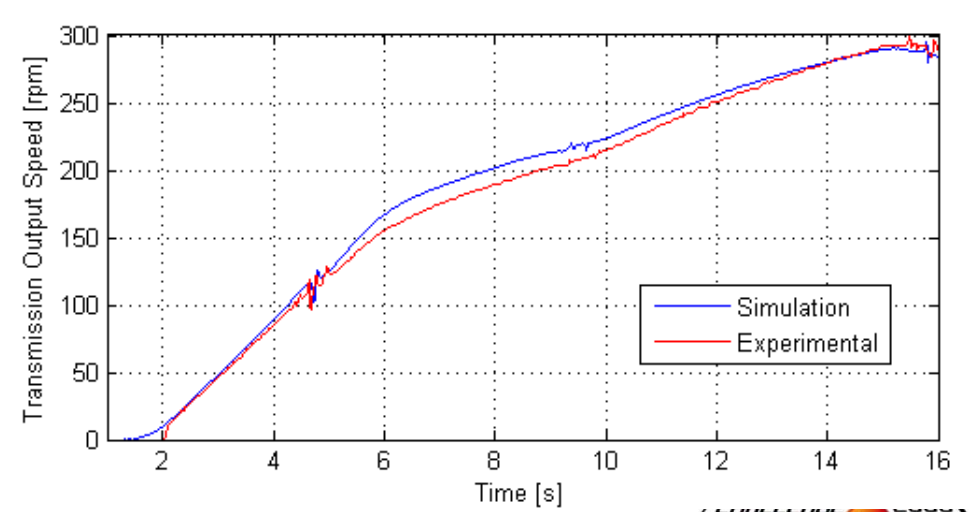
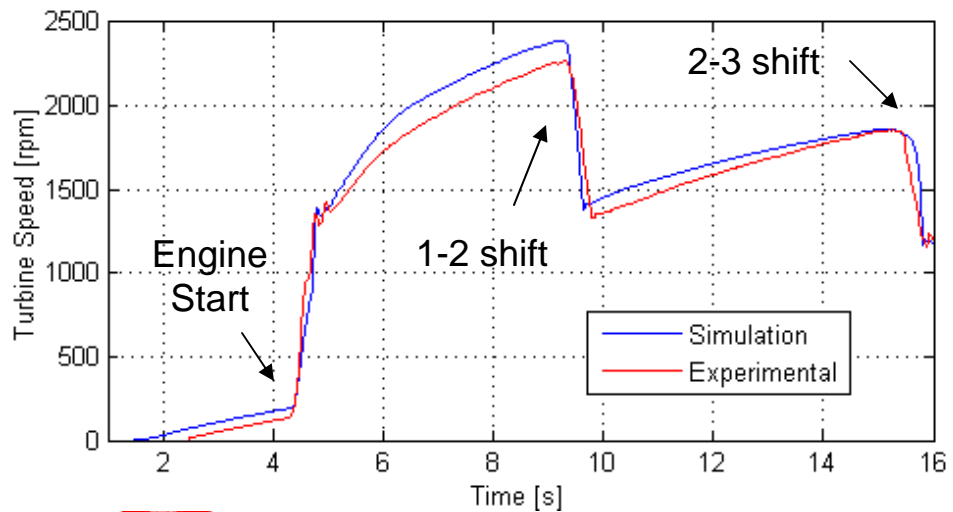
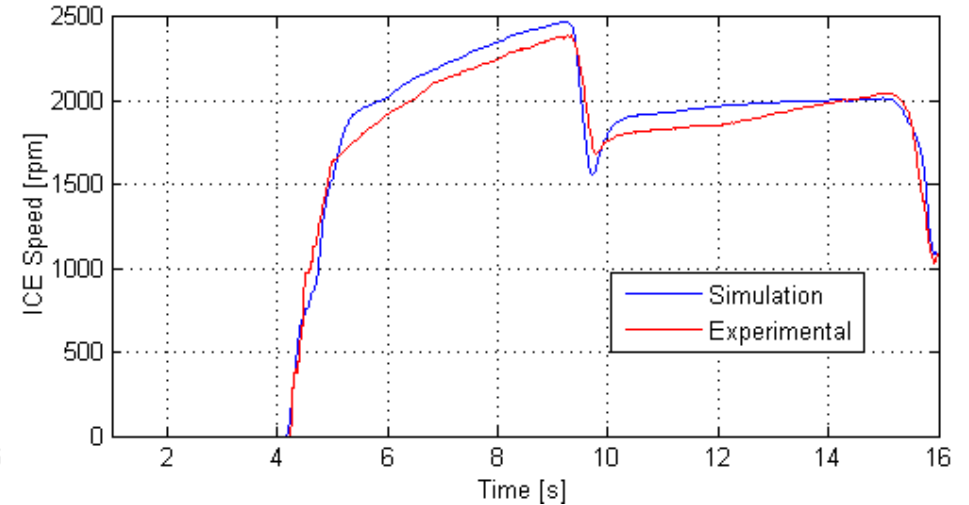
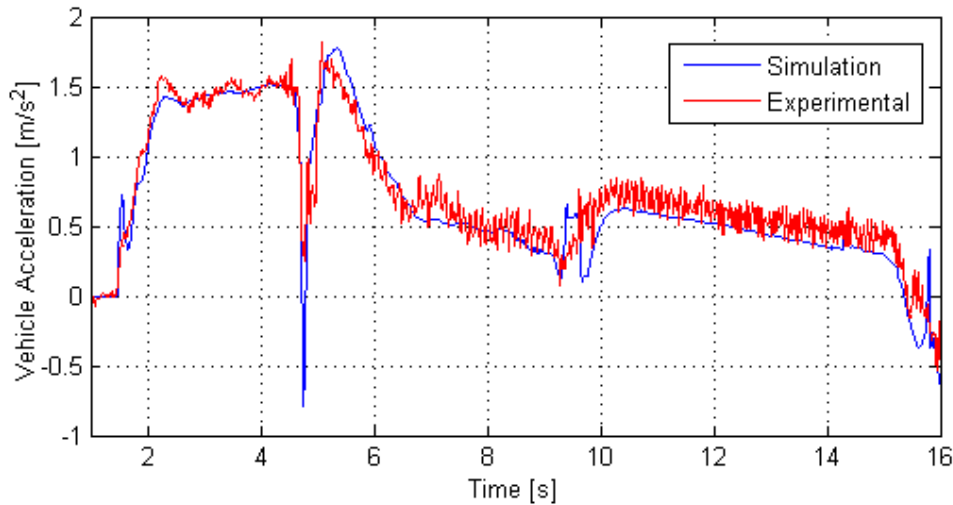
## Year 4 Improvements:

- Improved automatic transmission model (including engine start-stop).
- New Pacejka tire model.
- New driveline model with backlash.
- Model calibration.

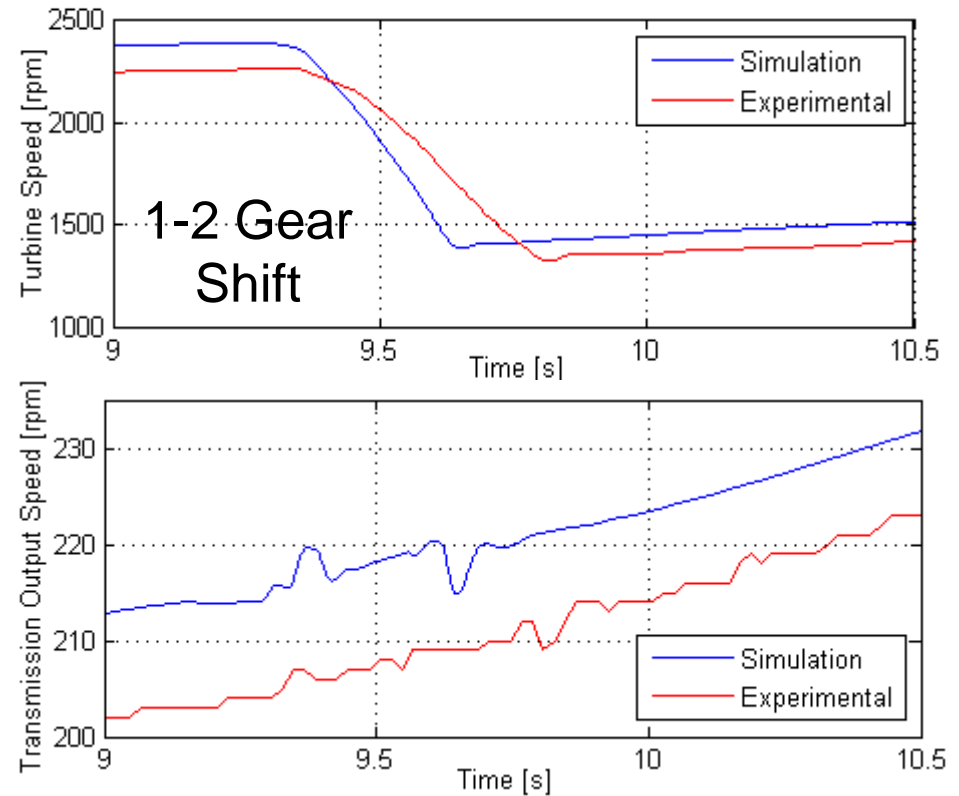
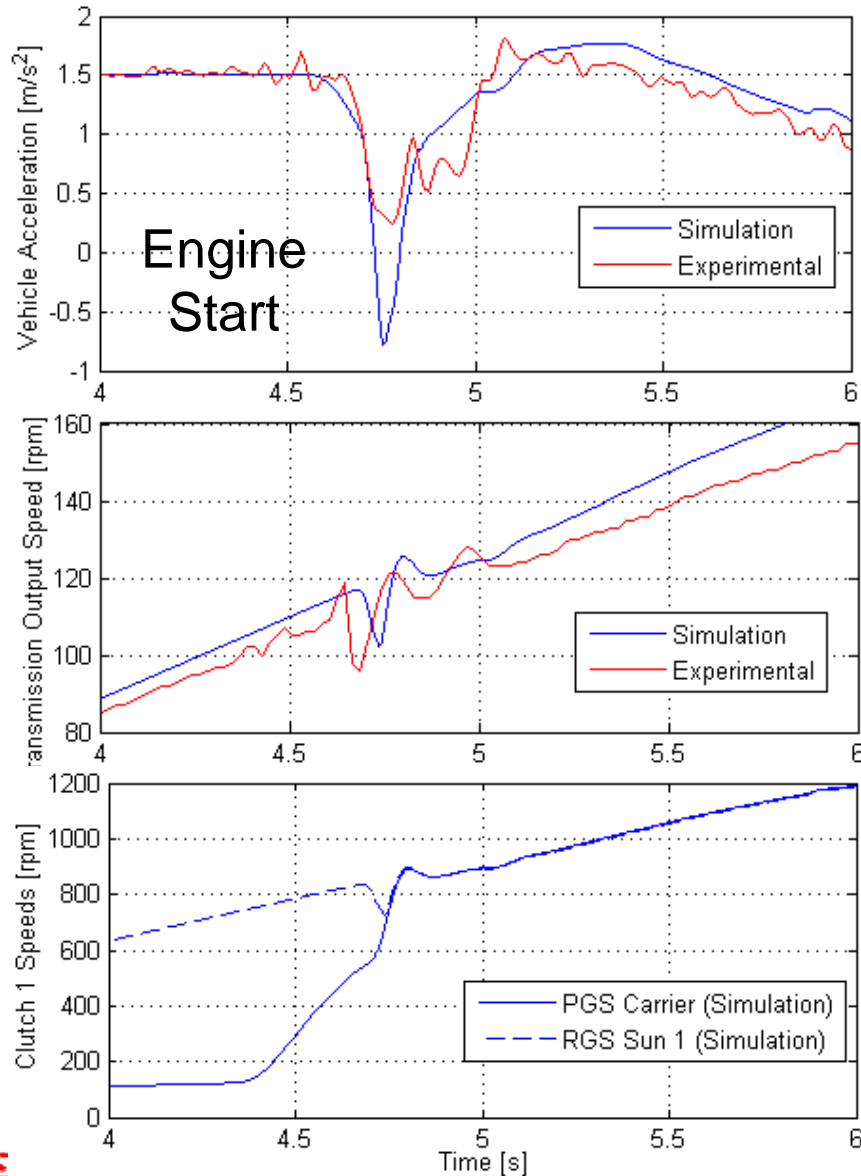


# CX-DYN: Experimental Validation

- Mild acceleration from 0 to 25 mph



# CX-DYN: Detail of Results

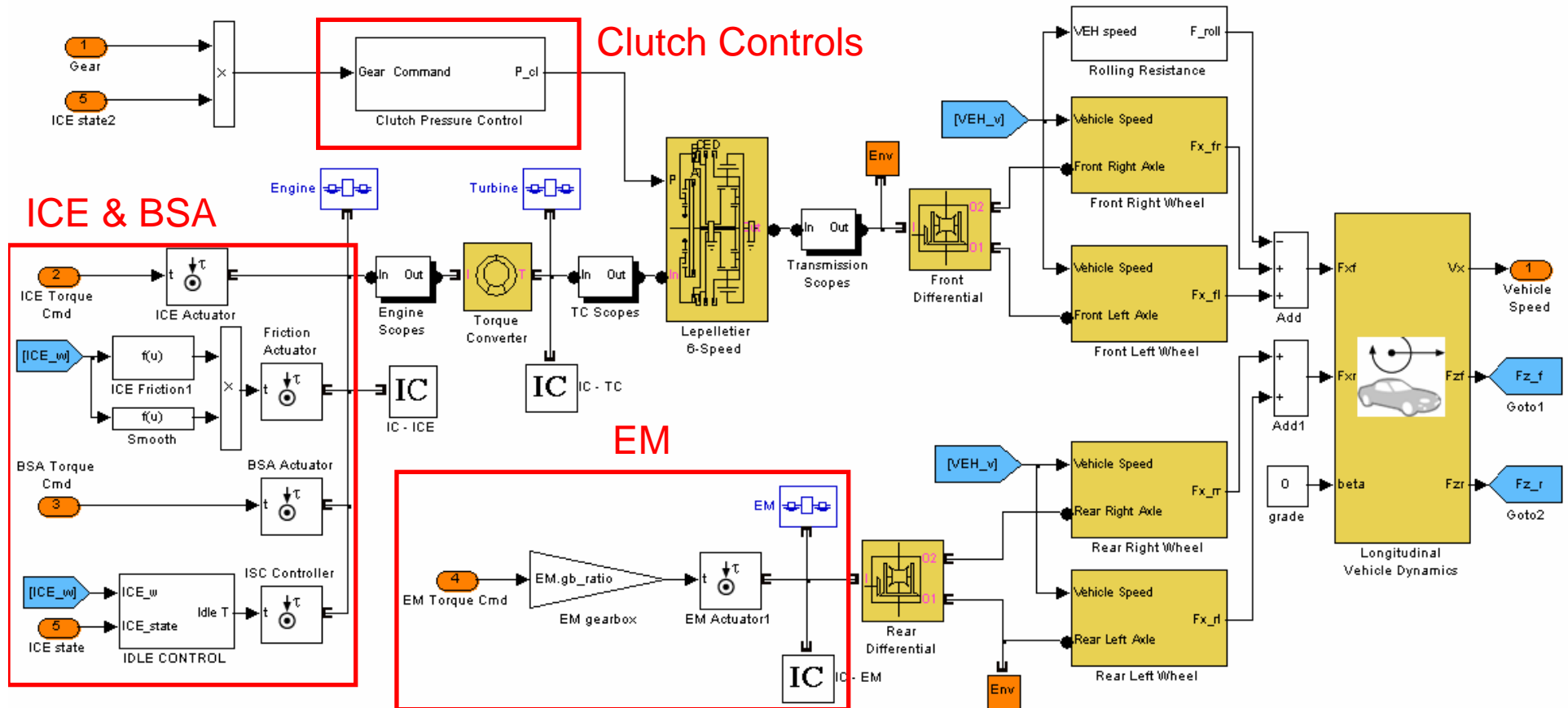


1 <sup>st</sup> gear	C1 engaged	OWC engaged
2 <sup>nd</sup> gear	C1 engaged	B2 engaged

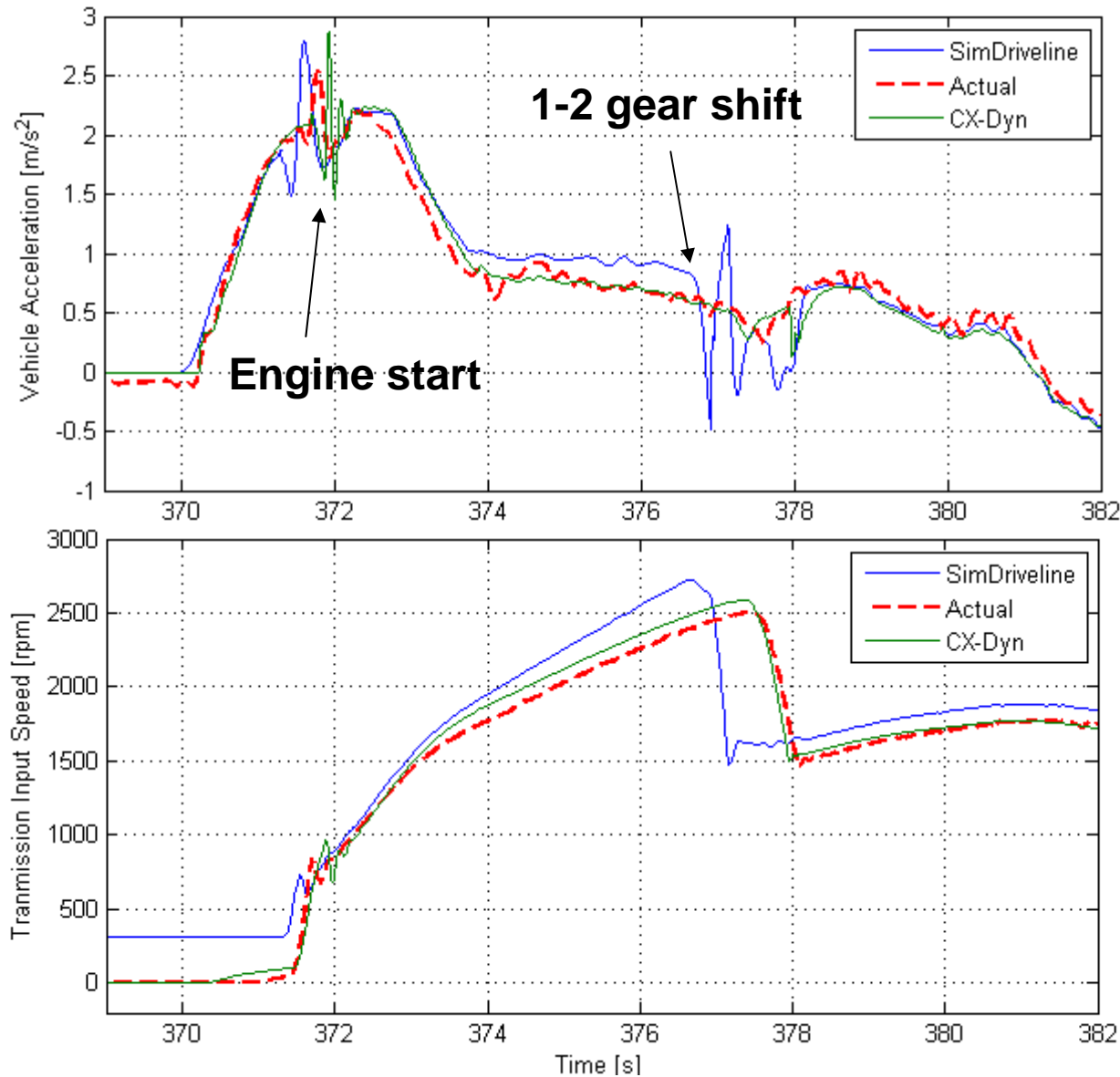
- Open-loop pressure profiles are used to represent simulated clutch pressure commands during gear shifts.

# SimDriveline Equivalent Model

- A similar dynamic HEV model can be built using **SimDriveline** thus achieving substantial time savings.
- This simulator consists of a combination of custom built components and SimDriveline blocks.



# CX-DYN – SimDriveline Comparison



- The SimDriveline model gives similar results as CX-DYN despite the low level of model calibration.

- 1-2 gear shift is captured more accurately with CX-DYN.

(same clutch pressure profiles are used in both models)



# CX-DYN – SimDriveline Comparison

	<b>SimDriveline simulator</b>	<b>CX-Dyn simulator</b>
<b>Pros</b>	Complex vehicle models can be quickly built.	Requires considerable effort and know-how to build.
	Simulations can be run in a matter of seconds.	Time-consuming to run simulations.
<b>Cons</b>	Not trivial to troubleshoot errors during model development.	Easier to troubleshoot.
	Some components need to be customized for application.	All components are custom built.

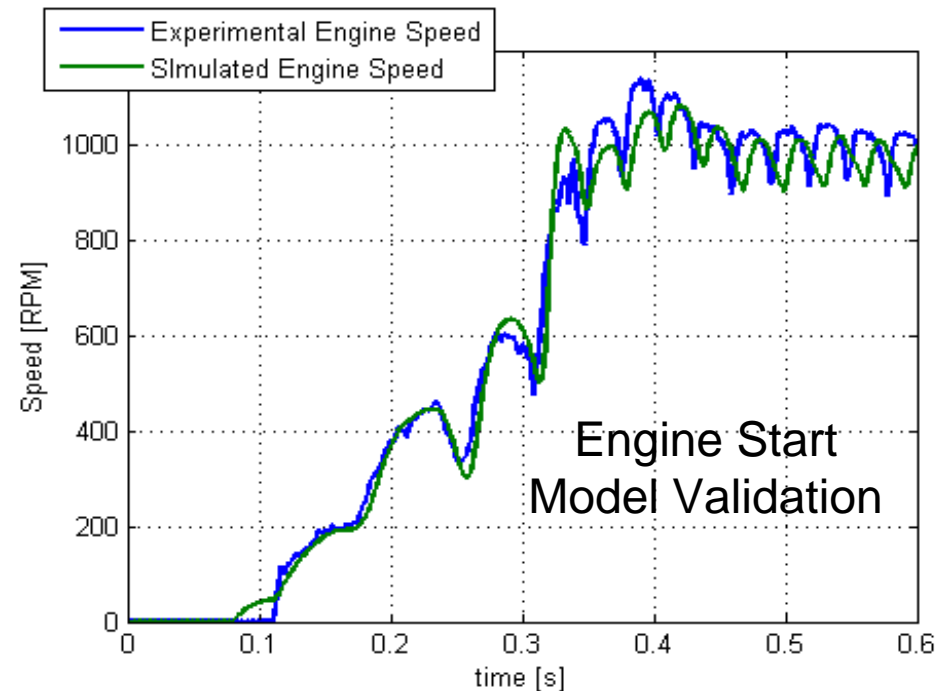
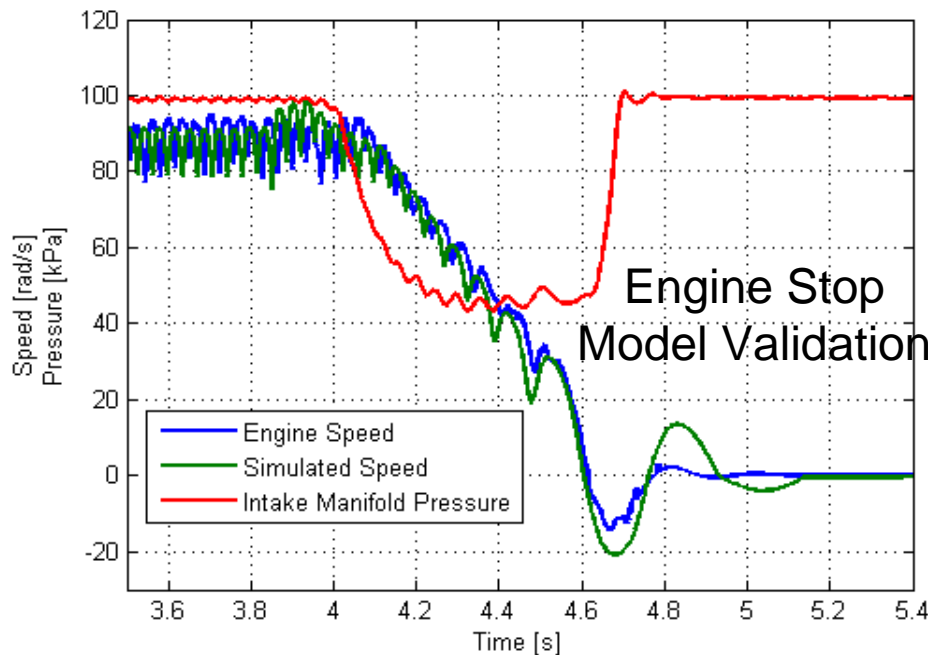
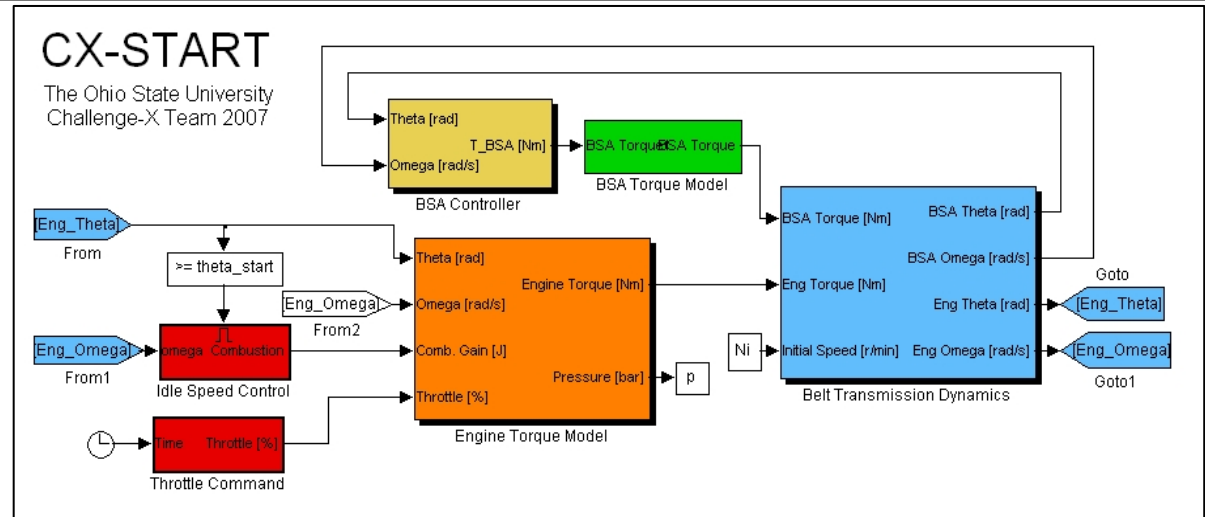


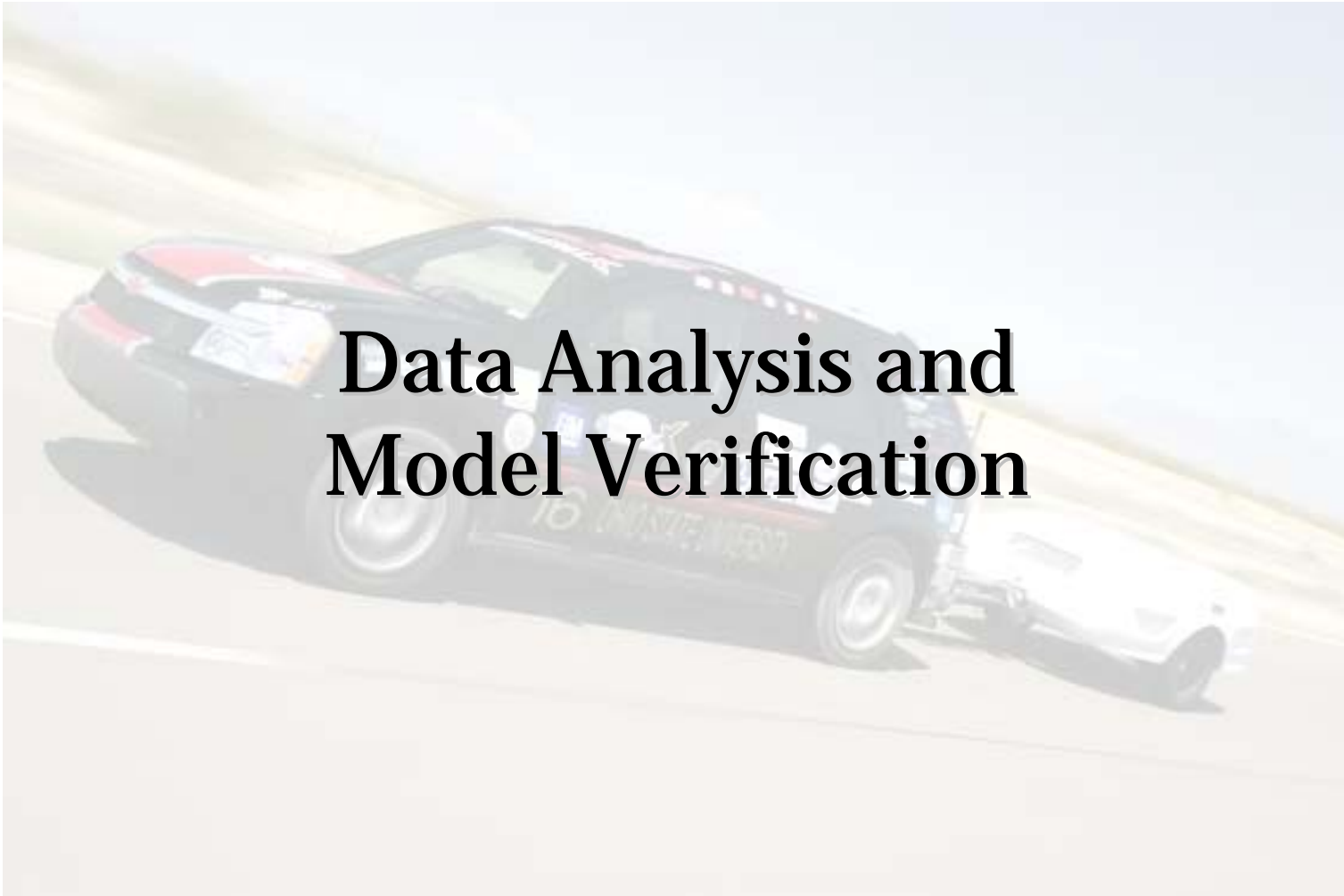
# CX-START: Engine Start Model

## CX-START

- Stand-alone simulator
- Detailed models of
  - diesel engine
  - starter alternator
  - belt dynamics

for engine start-stop control.











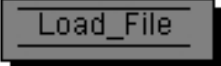
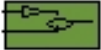


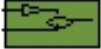

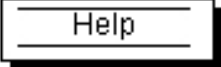




## CX-DAQ

- Data analysis and post-processing
- Simple handling of CAN data
- Supporting tool for model validation

## Data Acquisition

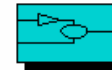
The Ohio State University  
ChallengeX Equinox CAN signal analysis

 Plot variables for TIM	 Select variables for TIM	 Plot All Selected Variables on Single Plot
 Plot variables for Engine	 Select Variables for Engine	 Choose settings and units for plots
 Plot variables for Transmission	 Select Variables for Transmission	 Load a *.mat data file
 Plot variables for Vehicle	 Select Variables for Vehicle	 Close all figures opened
 Plot variables for Battery	 Select Variables for Battery	 Double-click the box above for help and instructions
 Plot variables for Controls	 Select Variables for Controls	

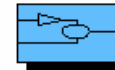
## CX-Graphics

- Developed in Year 1
- For use with developed vehicle simulators.
- Analyze simulation results.

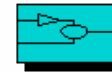
## cX Graphics



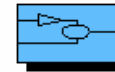
Plot the Results for Driver



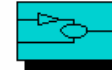
Driver Plot Options



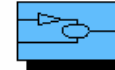
Plot the Results for Vehicle



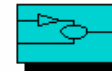
Vehicle Plot Options



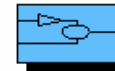
Plot the Results  
for Acceleration Test



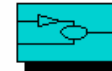
Acceleration Test  
Plot Options



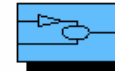
Plot the results  
for HEV Operation



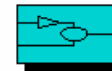
HEV Operation  
Plot Options



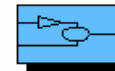
Plot the results  
for Conventional Powertrain



Conventional Powertrain  
Plot Options



Plot the results  
for Electric Powertrain



Electric Powertrain  
Plot Options

**Click on "Start Simulation "**  
**after setting new plot options**  
**to save the new properties**

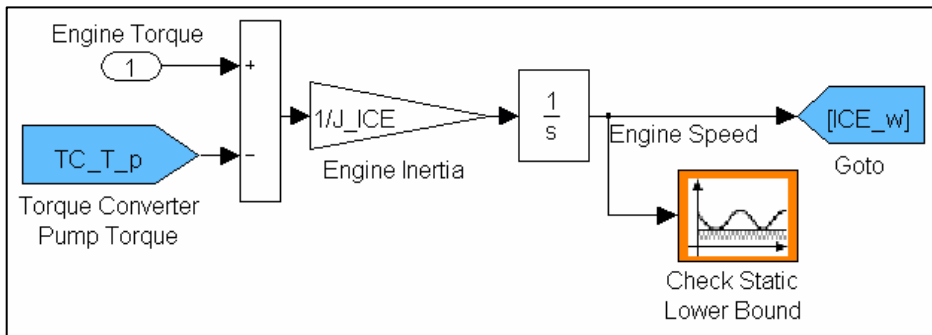
**CLOSE ALL**

Close all figures opened

# Model Checking and Verification

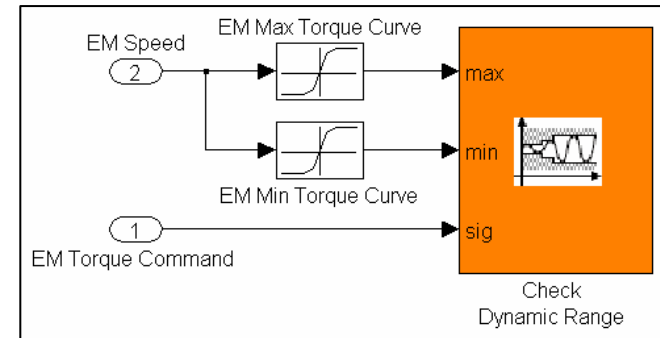
- Model verification toolset simplifies...

## Simulation error debugging:



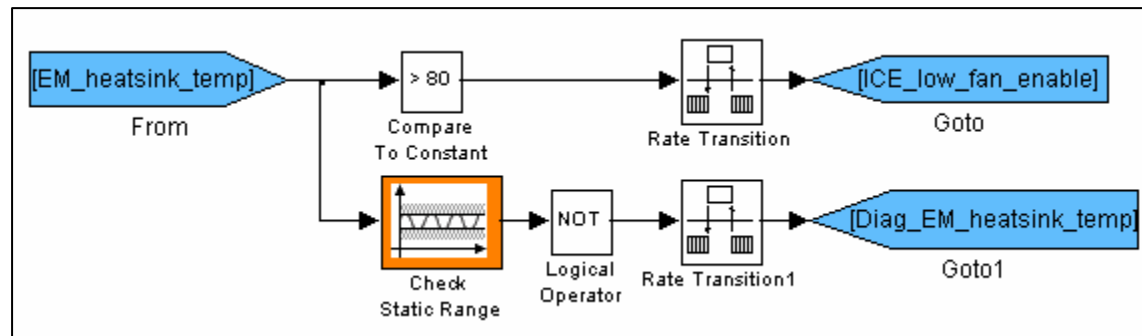
*Stop simulation if engine speed < 0.*

## Control strategy verification:



*Stop simulation if torque command is out of limits.*

## Hardware fault diagnosis:



*Send fault bit if inverter temperature is out of range.*



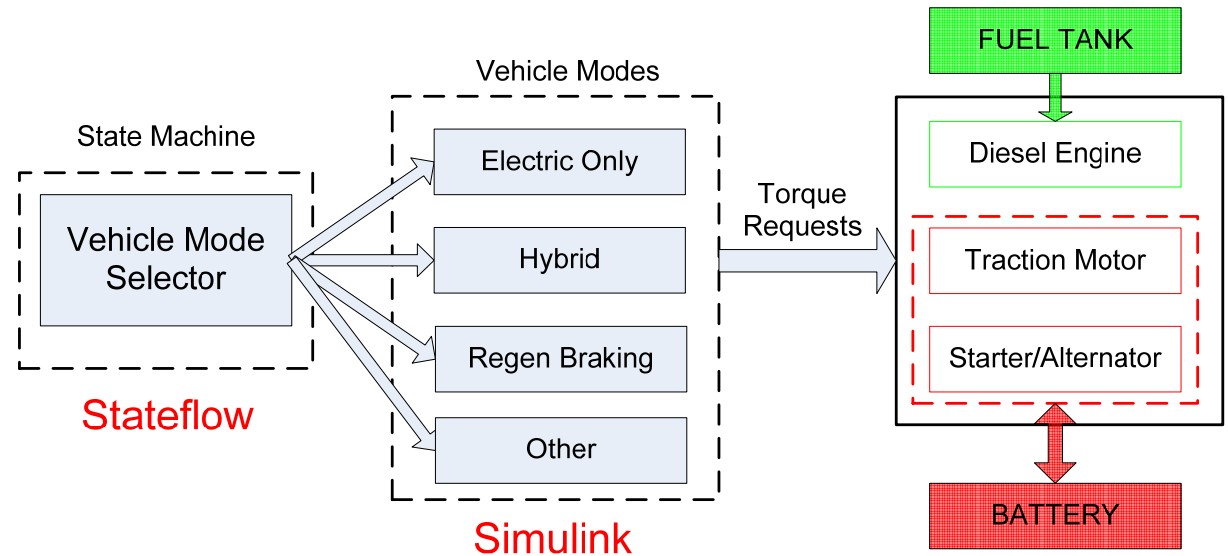
# Control Design, Implementation and Optimization



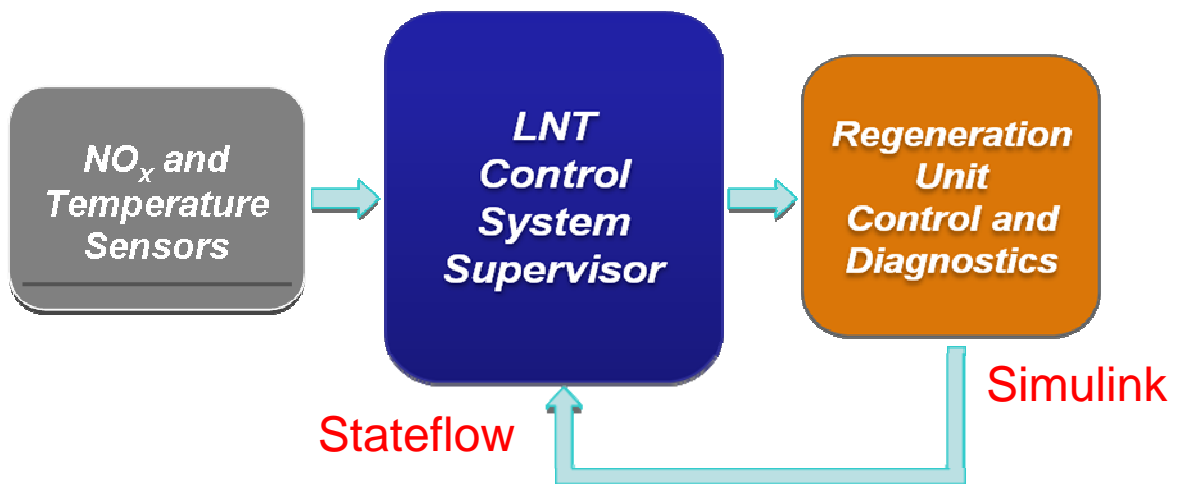
# Supervisory Controls

- Supervisory control strategies implemented using **Stateflow**.

– Vehicle mode management



– High-level exhaust after-treatment controls.



# Adaptive Energy Optimization Strategy

- Equivalent fuel Consumption Minimization Strategy (ECMS):
  - An adaptive algorithm to minimize “weighted equivalent fuel use”:

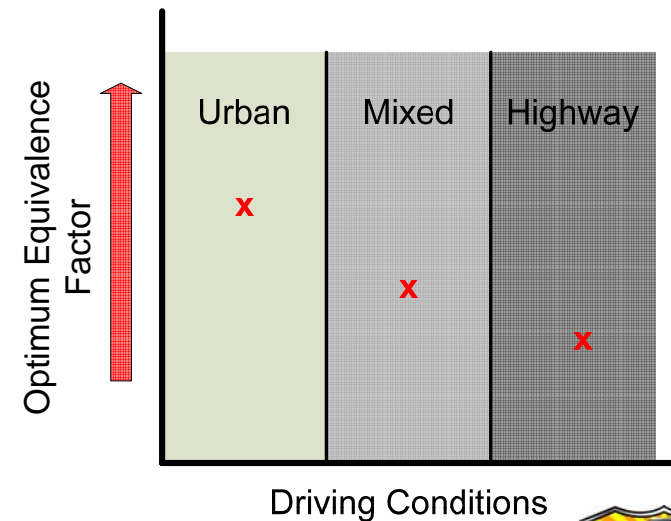
$$\{T_{ICE}, T_{EM}, T_{ISA}\} = \arg \min \{ \dot{m}_{fuel} + s \cdot \sum_i \dot{m}_{EQ,i} \}$$

$$\dot{m}_{EQ,i} = \frac{T_i \cdot \omega_i}{Q_{LHV}} \cdot \begin{cases} 1/\eta_\phi(x,u), & \text{if Discharging} \\ \eta_\phi(x,u), & \text{if Charging} \end{cases} \rightarrow \text{Equivalent fuel for electric machines}$$

- Equivalence Factor,  $s$ :  
Updated online based on driving cycle characteristics and battery state-of-charge (SOC).

$$s = s_{nominal} \cdot \underbrace{\zeta_P \cdot \zeta_I}$$

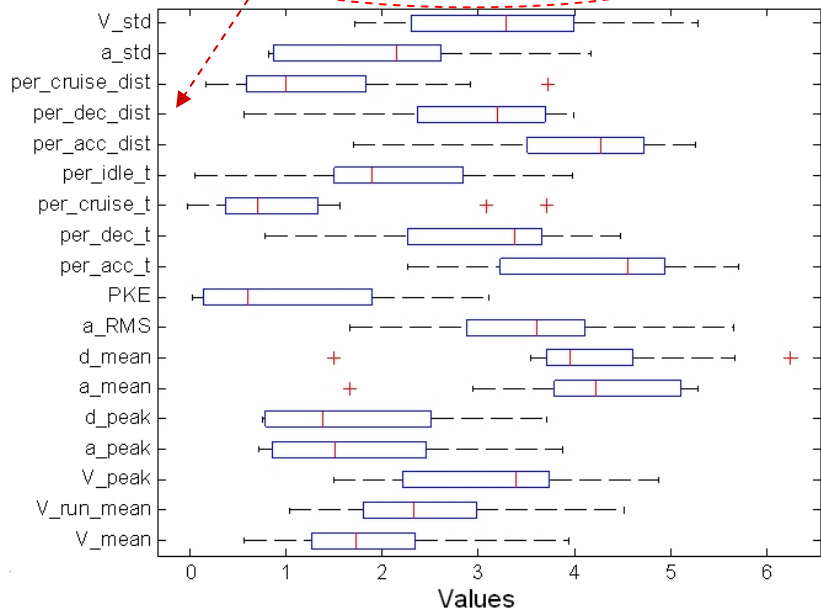
SOC Control: Gains are modified when SOC deviates from nominal range (50-80%).



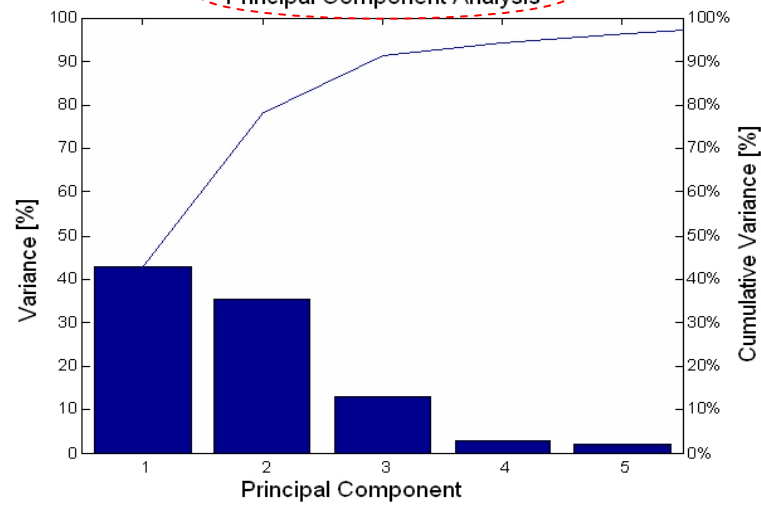
- **MATLAB Statistics Toolbox** is used to recognize the past driving pattern and adaptively tune the control strategy.
- Driving cycle clusters are formed based on 18 statistical metrics.

Commands: `kmeans`, `silhouette`, `boxplot`, `princomp`, `pareto`

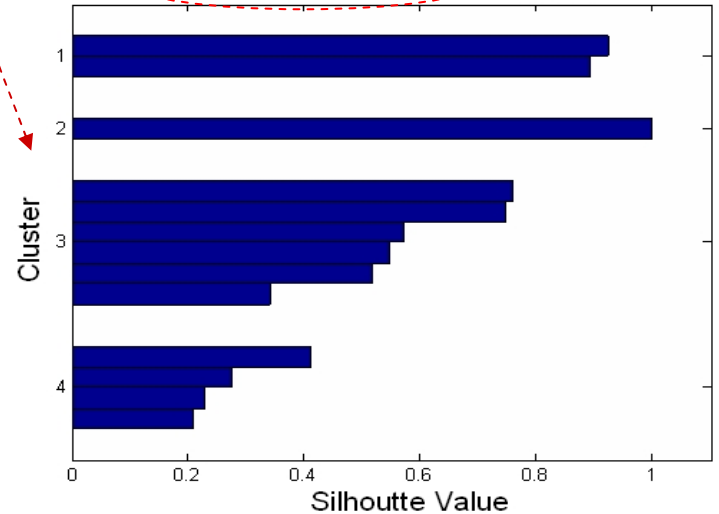
Boxplot of Standardized Data



Principal Component Analysis



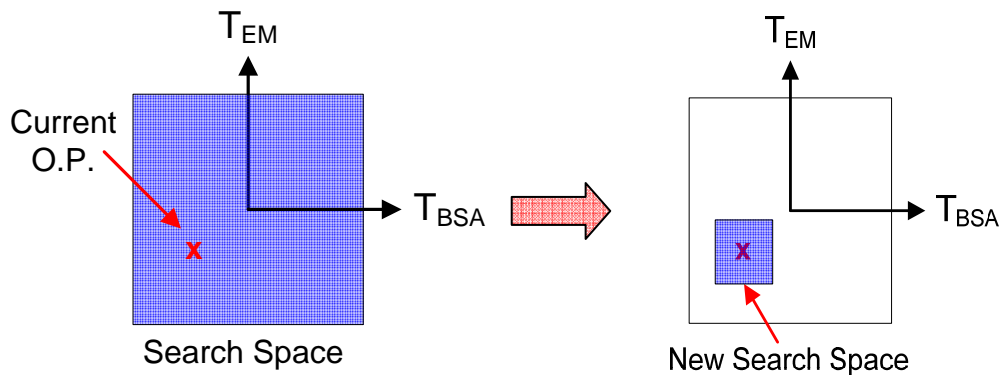
K-Means Clustering - 4 Classes



# Implementation - Code Optimization

- Fuel minimization strategy modified for efficient real-time implementation.

Year 3      Year 4  
*Look-up table* → *MATLAB code suitable for implementation*



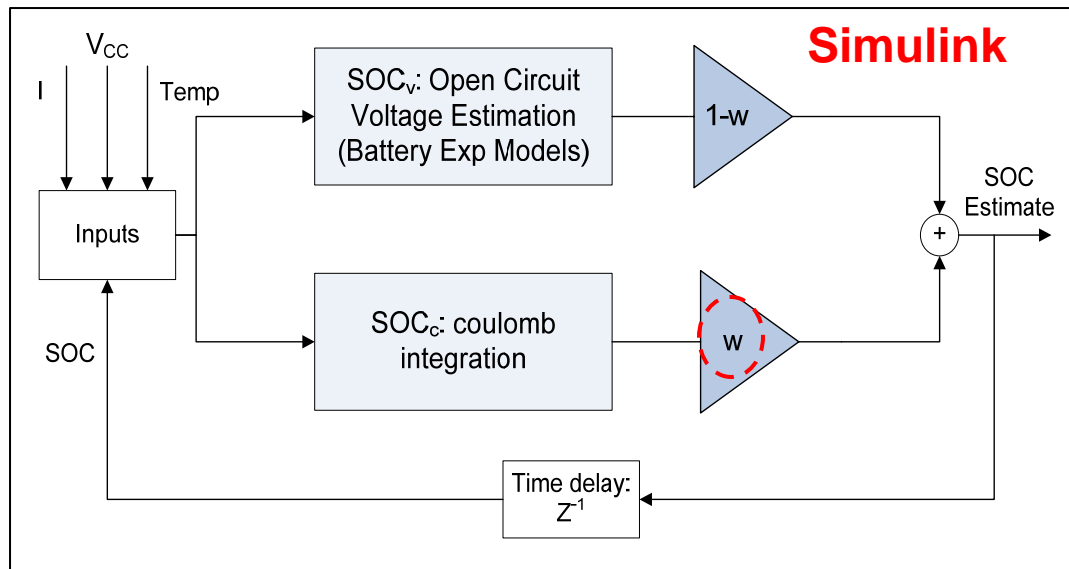
### Benefits:

- Fast code execution.
- Reduces “torque chattering”.
- Advantageous for initialization during controller transfer.

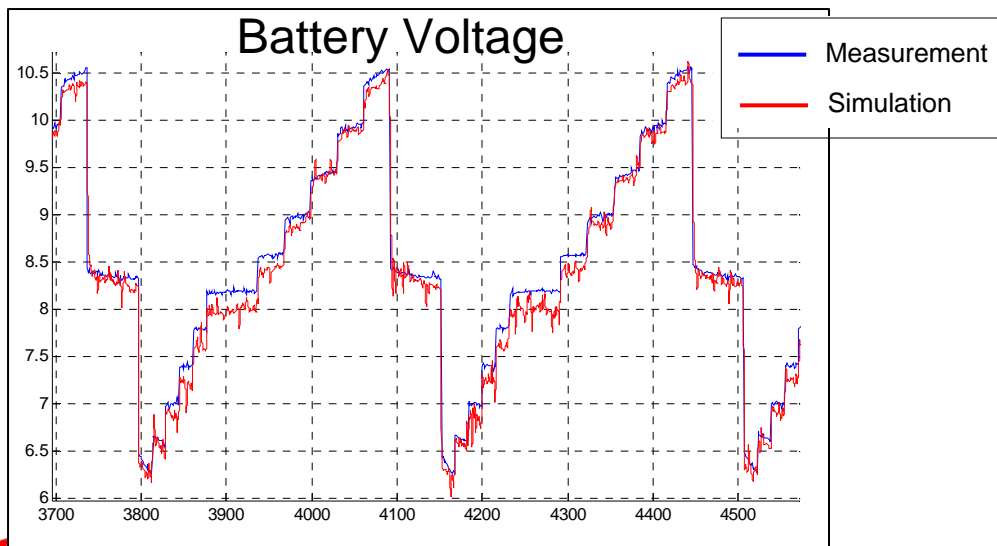
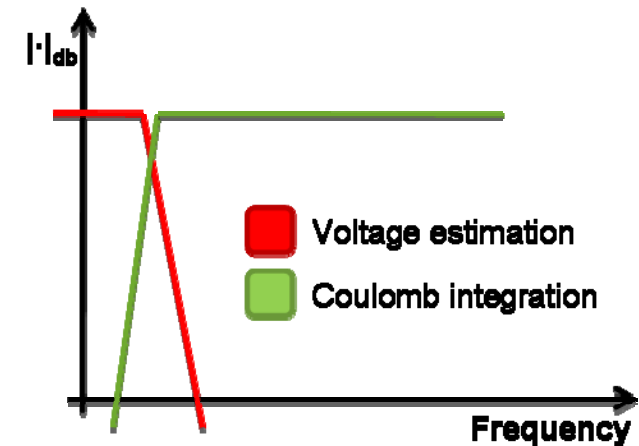
- ECMS code is implemented using an **embedded MATLAB function**.
- Look-up table data types are optimized for low memory use.
- Computational capacity of the vehicle control unit (code “turnaround time”) is monitored to ensure proper operation.



# Battery State-of-Charge Estimation

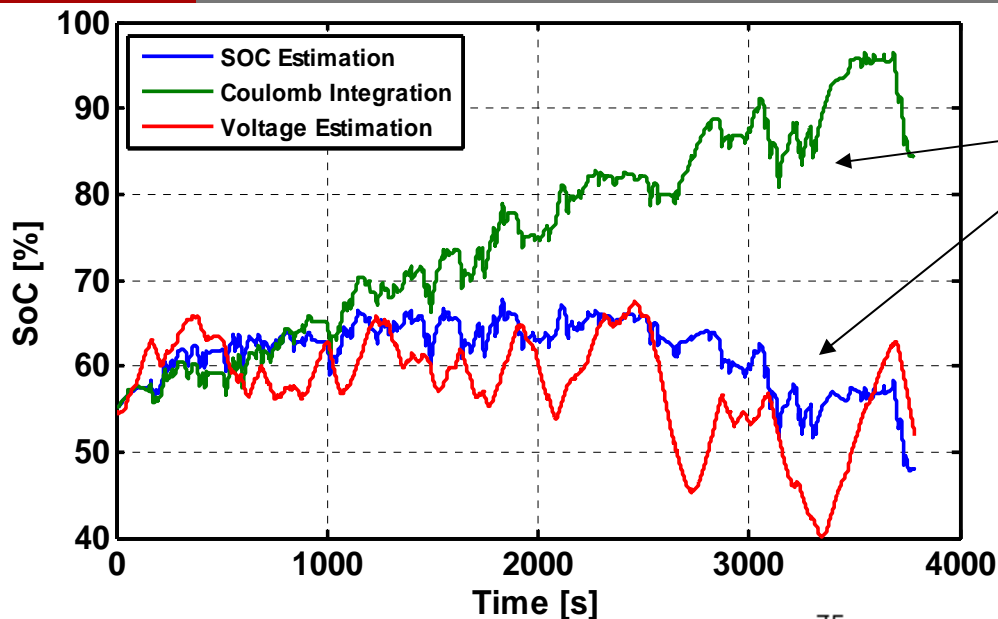


Frequency separation principle:



- Experimental battery model (used for open-circuit voltage estimation) is improved in year 4.
- Estimator's weighting algorithm is improved.

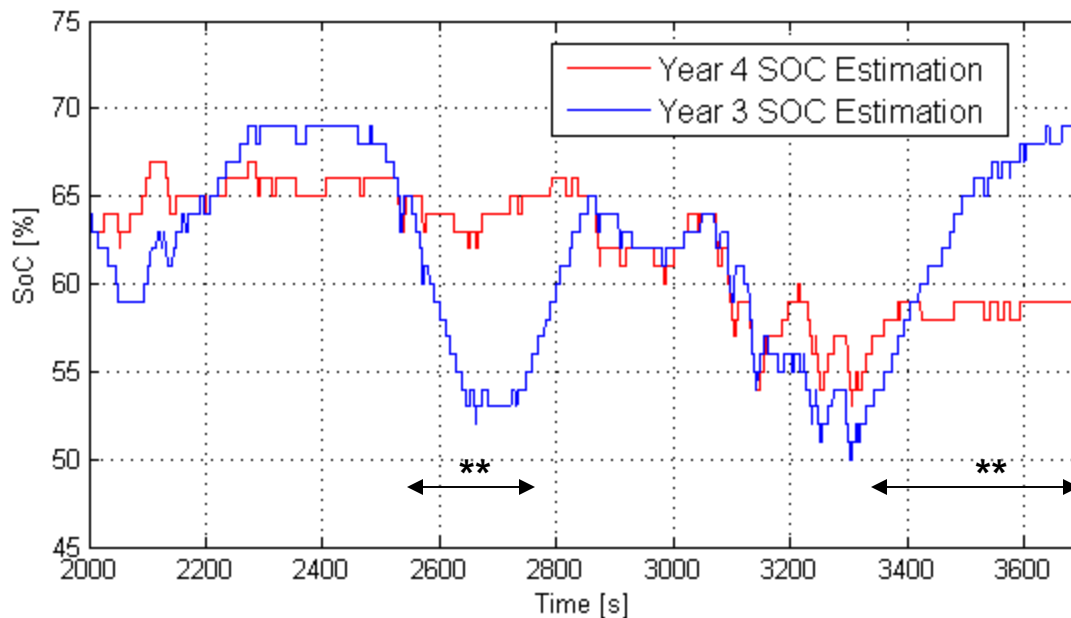
# Battery State-of-Charge Estimation



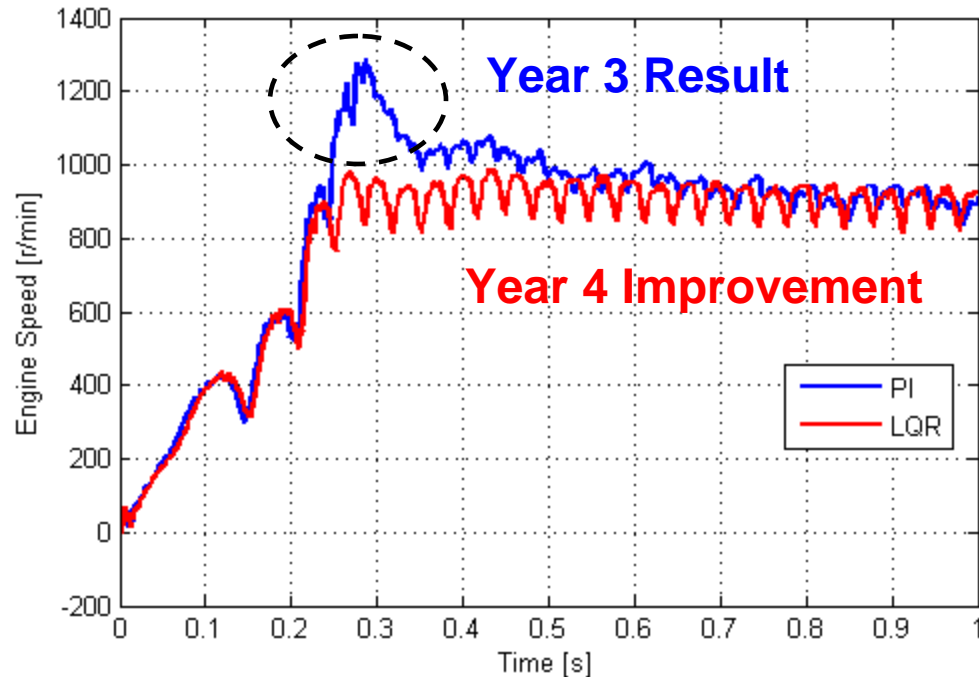
- Cumulative bias in the current measurement is compensated by the SOC estimator.

\*\* Year 3 algorithm over-responsive to voltage variations at low charge/discharge rates.

- Improved initialization.
- Accurate response to short-term charge variations.
- Proven stability.



# Engine Start: Engine Speed Control



- Engine start strategy revised in year 4.
- PI-type engine start controller replaced with a model-based (LQR) controller:  
→ Less calibration effort.

Control Design Toolbox : `lqr`

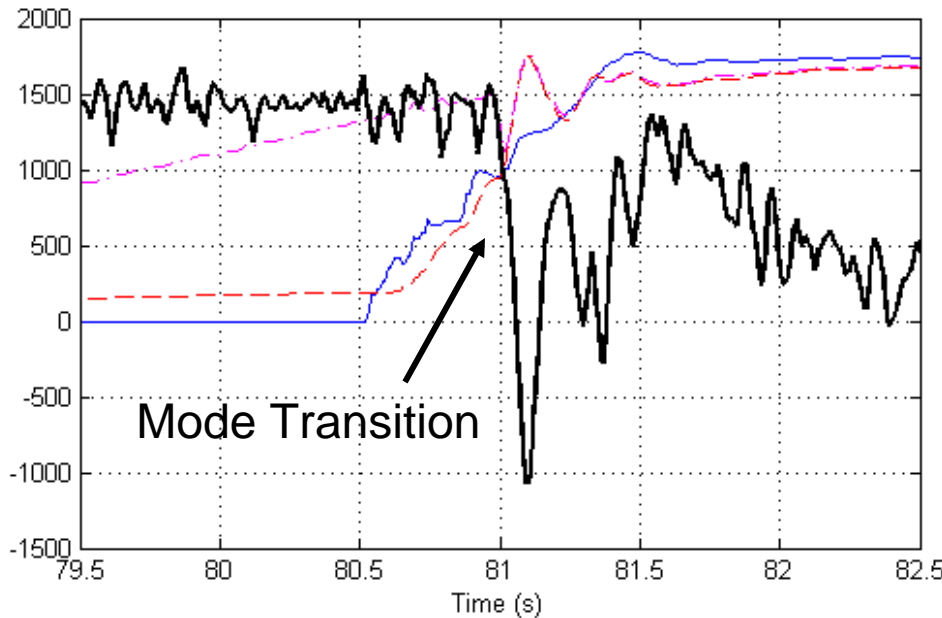
- LQR controller eliminates engine speed overshoot.
- Less BSA torque is used to start the engine:  
→ Better fuel economy.

# Engine Start: Torque Blending

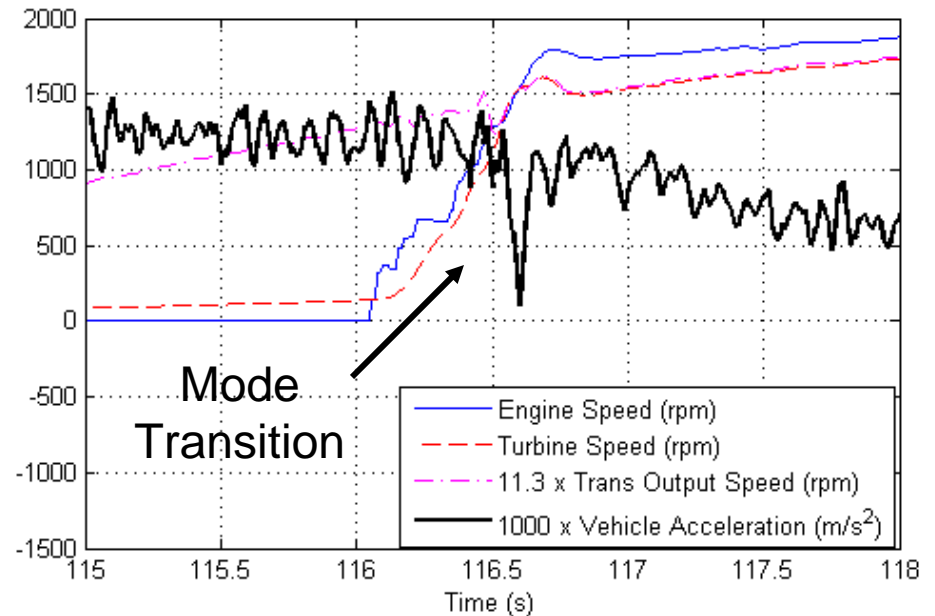
- A torque blending strategy is implemented to avoid driveline disturbances after engine start.
- A model-based hybrid control technique is used.

- This method exploits the benefits of the fast dynamic response of the electric machines.

## Vehicle Acceleration and Driveline Speeds



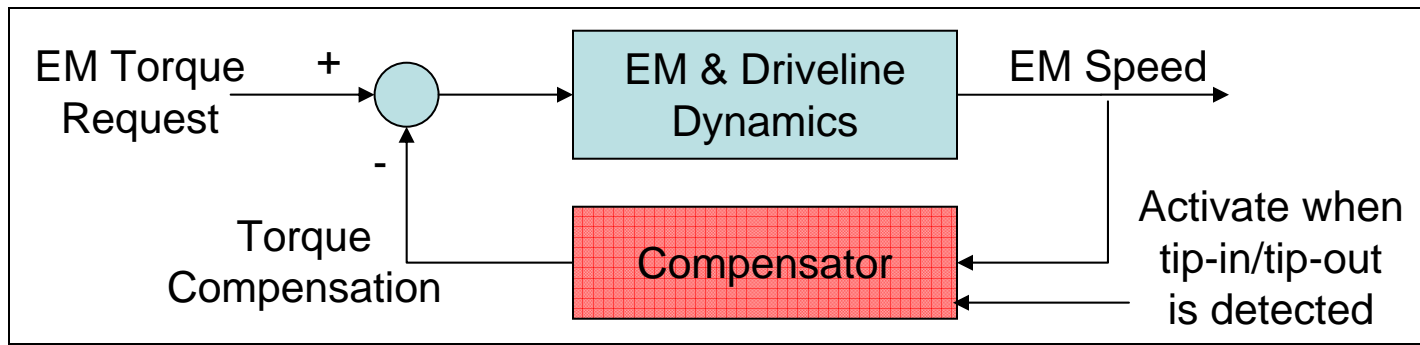
**Year 3 Result**



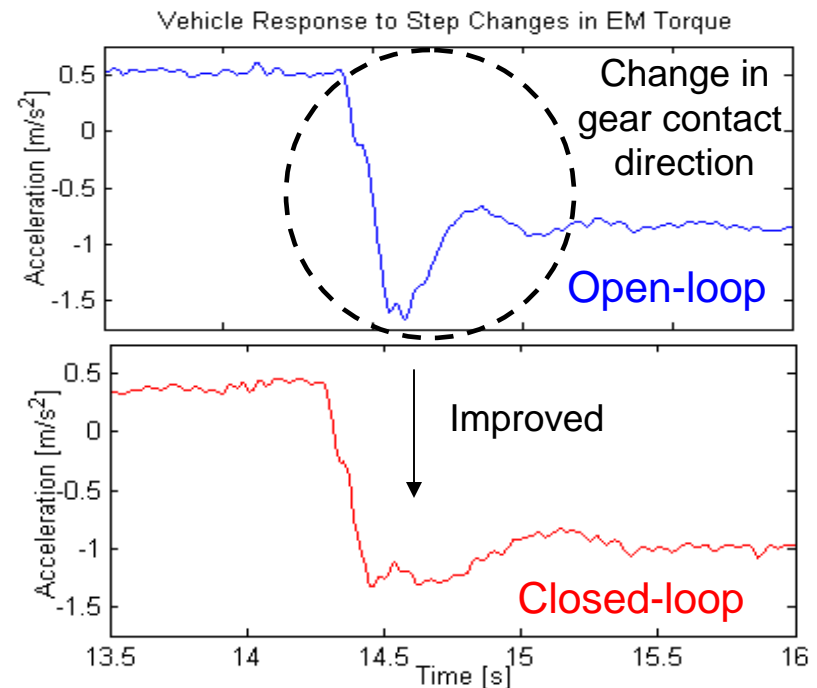
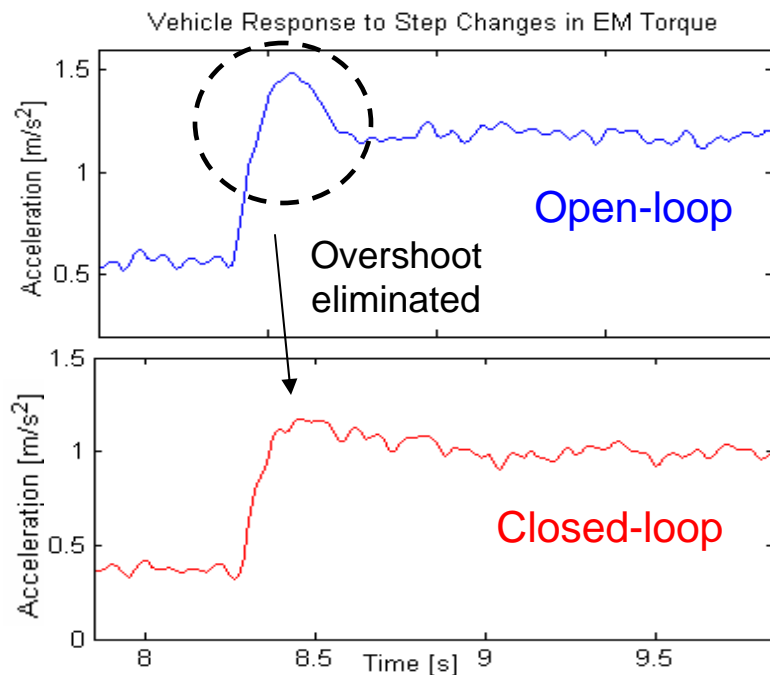
**Year 4 Improvement**

# Electric Mode: Driveline Control

- Gear backlash and the absence of a damping element causes torque disturbances in the rear driveline during **pedal tip-in tip-out**.



Active driveline damping technique



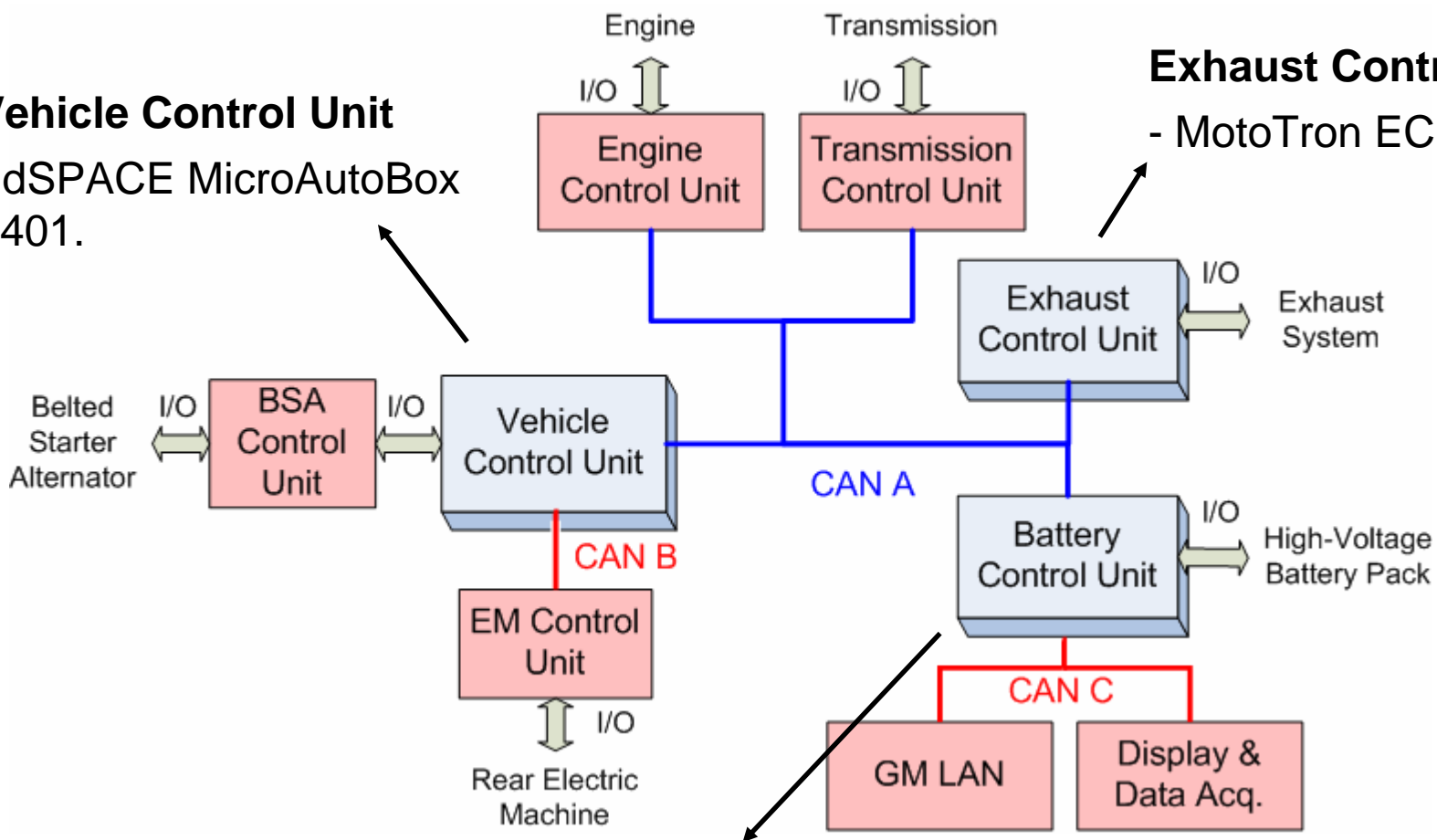
# Control Architecture

- Hardware
- Software



# Control Hardware Architecture

**Vehicle Control Unit**  
- dSPACE MicroAutoBox 1401.

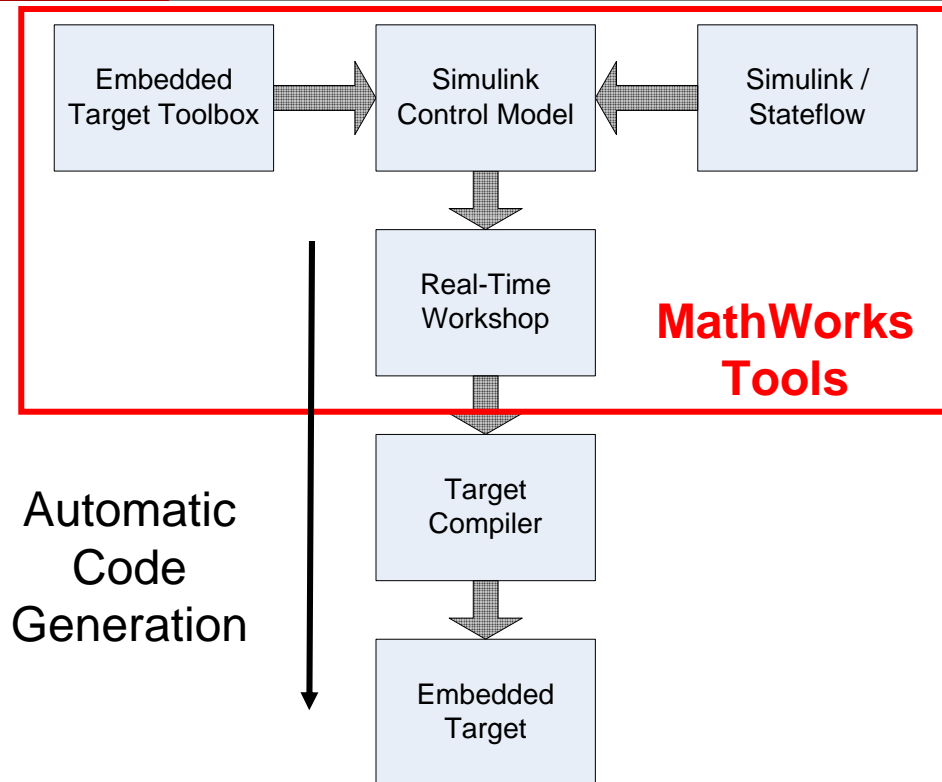


**Exhaust Control Unit**  
- MotoTron ECU-80.

**Battery Control Unit**  
- Phytec development board (Freescale MPC555).



# Control Software Architecture



- MathWorks tools simplify the control development and code generation processes.
- Automatic code generation process works seamlessly regardless of the choice of the embedded target.

## Vehicle Controller Software Specifications

- Operations are divided into 4 sampling times (1-10-100-1000 ms).
- Sampling times are selected according to the task priorities and the computational burdens of code segments.
- Operations are processed in multiple timer task mode.

# Lessons Learned in Year 4

- MathWorks tools greatly simplified:
  - Design of simulation models and data analysis,
  - Design, implementation and verification of control strategies.
- Pros and cons of using custom designed simulators versus using modeling packages (such as SimDriveline) are well-understood.
- Code optimization is crucial to achieve high control performance. More emphasis should be placed on code optimization if the control application is production intended.
- Current experiences will be carefully documented and delivered to the next competition team to minimize their initial learning period.



**Thank you for your attention.**

