

# Model-Based Design: Design with Simulation in Simulink

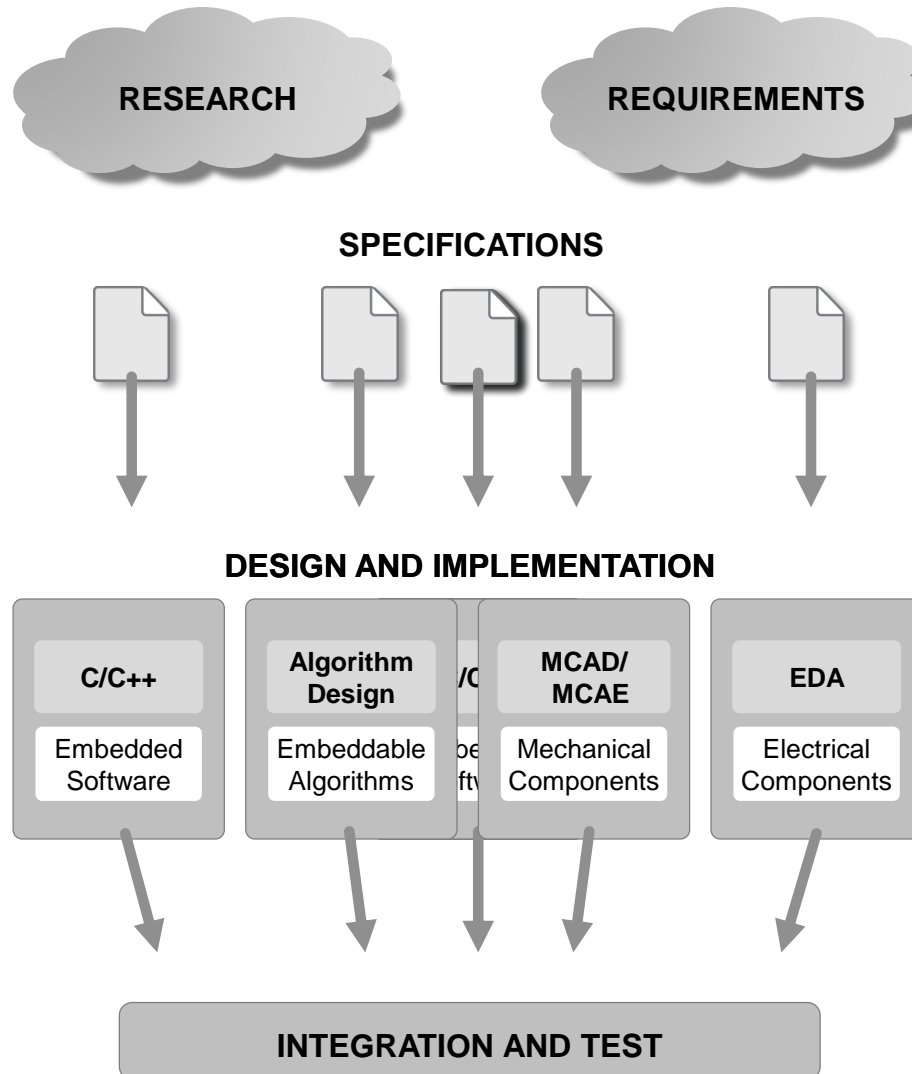
**Ruth-Anne Marchant**  
**Application Engineer**  
**MathWorks**



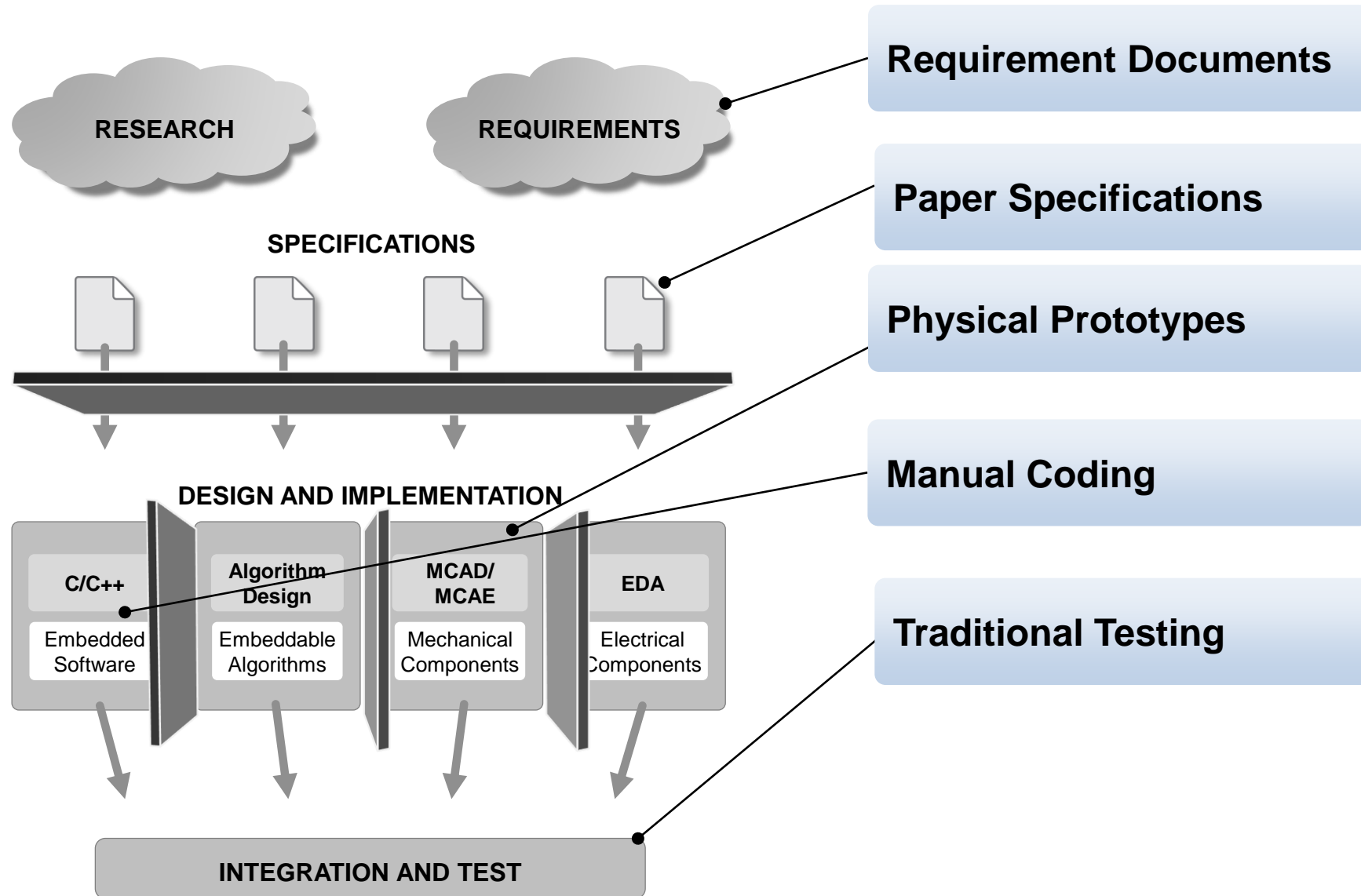
# Outline

- Model-Based Design Overview
- Modelling and Design in Simulink
  - Modelling
    - Physical Systems
    - Control logic
  - Simulation
    - System-level optimisation
    - Verification of design changes
- Summary

# Traditional Development Workflow



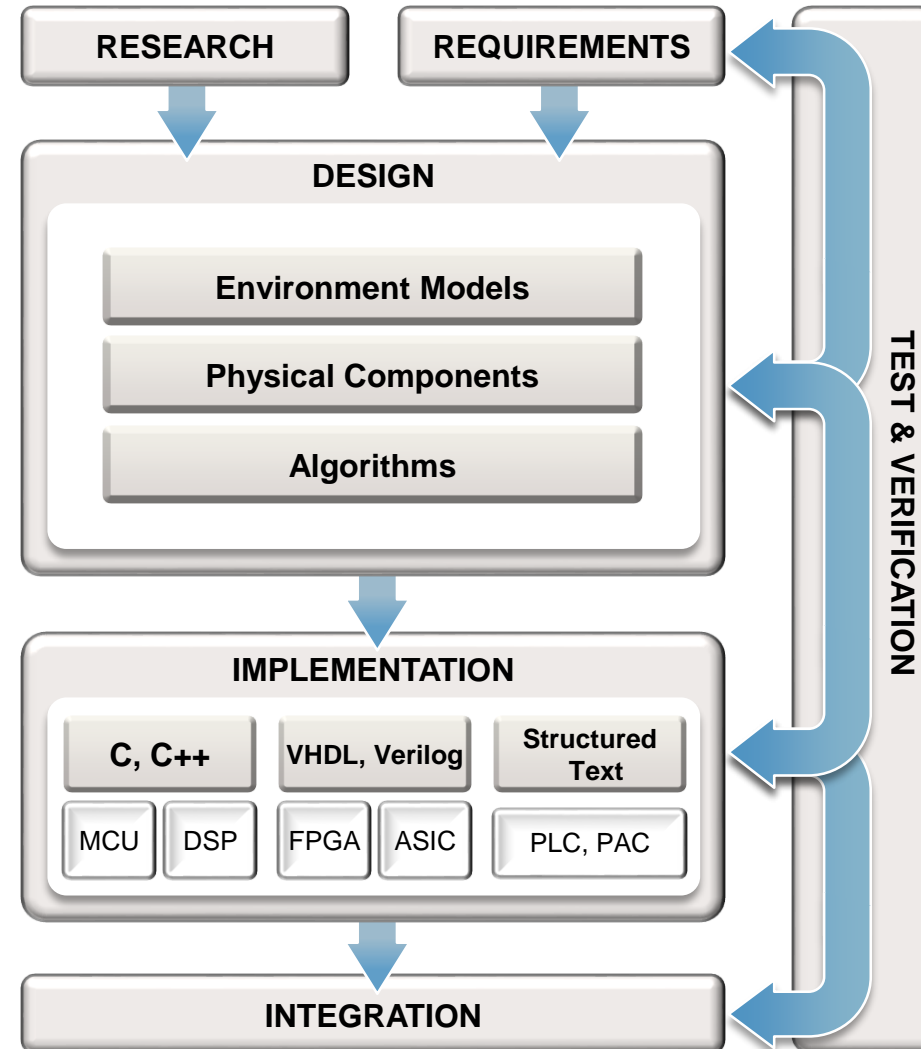
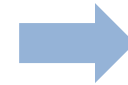
# Problems in Traditional Development Workflow



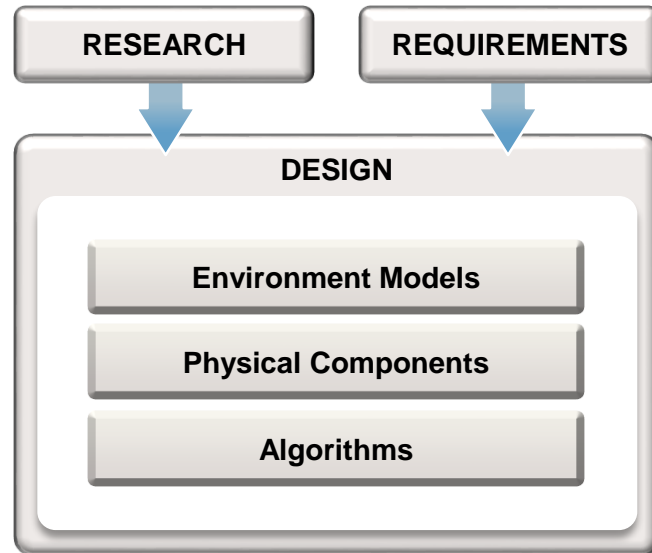
# Model-Based Design Workflow

## Traditional System Development Workflow

1. Research
2. Requirements and Specifications
3. Design
4. Implementation
5. Test and Verification



# Model-Based Design: Specifications



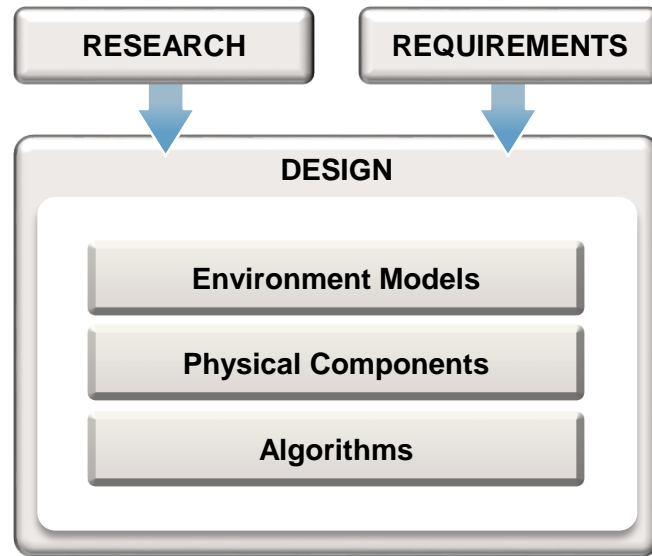
## CAPABILITIES

- Executable specification
- Executable constraints
- Links to requirements

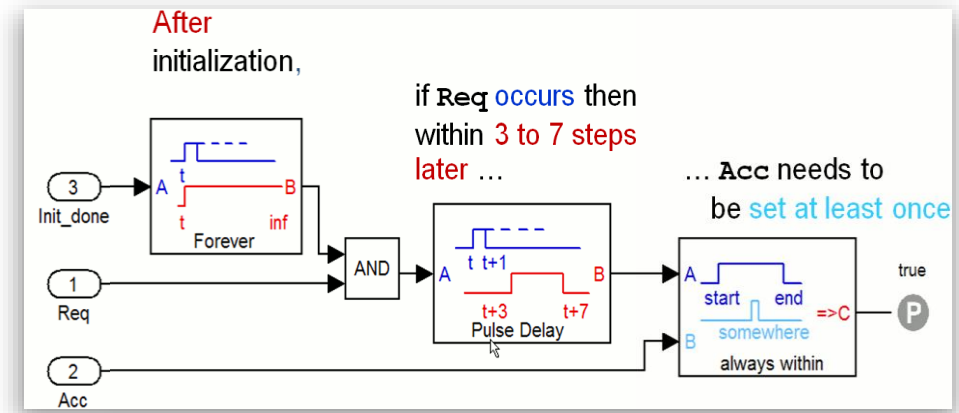
## BENEFITS

- Early validation and test development
- Clear specification
- Simulate whole system, including environment
- Tight link to requirements

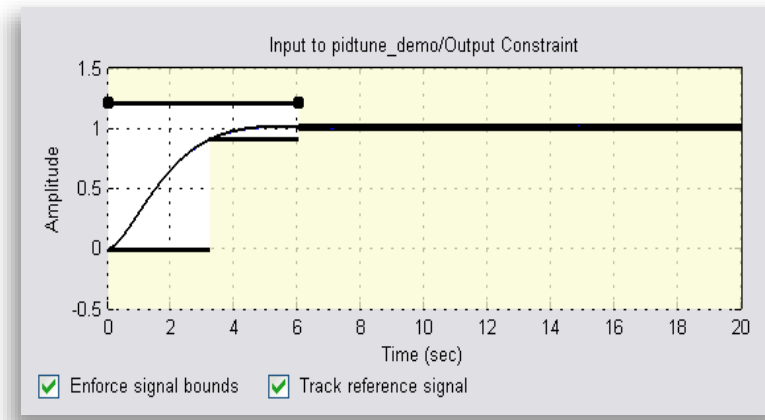
# Model-Based Design: Requirements



## Formalize requirements as properties and objectives



## Model system response bounds

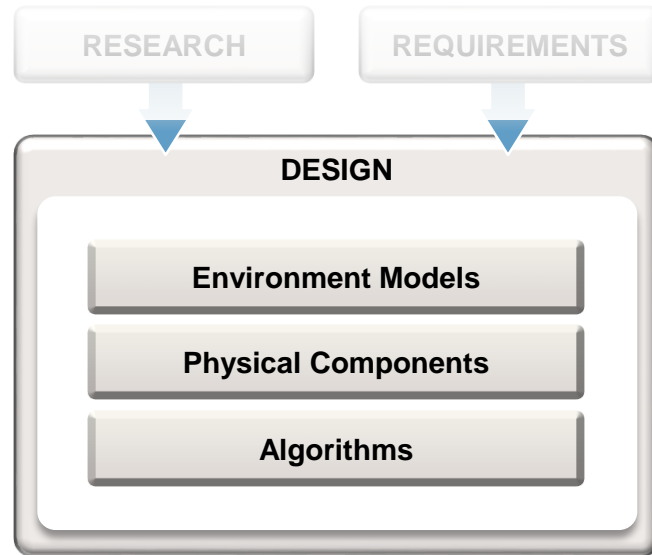


## Trace to requirements in DOORS, Word, Excel, etc.

Requirements	
1.1 Purpose of the document	
Module ID	00000064
Module Location	/QE/unit/v/Attitude Controller Derived Requirements
Object ID	2
Test Cases	<a href="#">TestCase1</a>
Object Heading	Purpose of the document
Object Text	This document provides the derived software requirements for a reusable attitude controller that will be used in the Autopilot Project



# Model-Based Design: Design



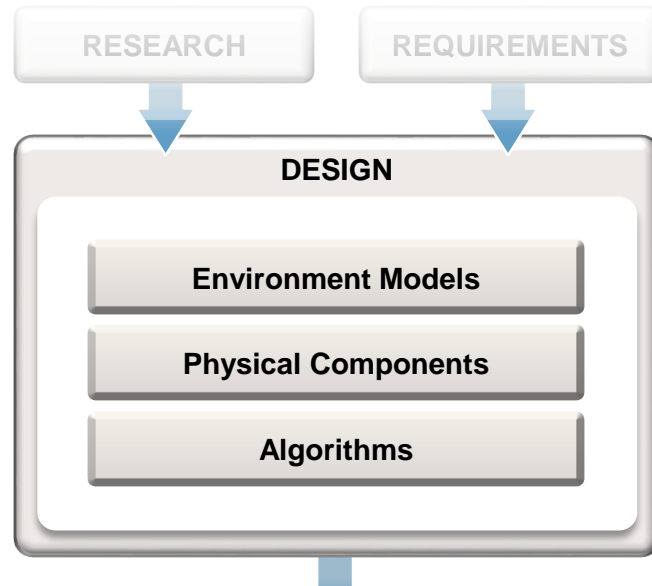
## CAPABILITIES

- Refine model description
- Add fixed point, timing, component interface details

## BENEFITS

- Fast design exploration
- Design optimization
- Find flaws before implementation

# Model-Based Design: Design



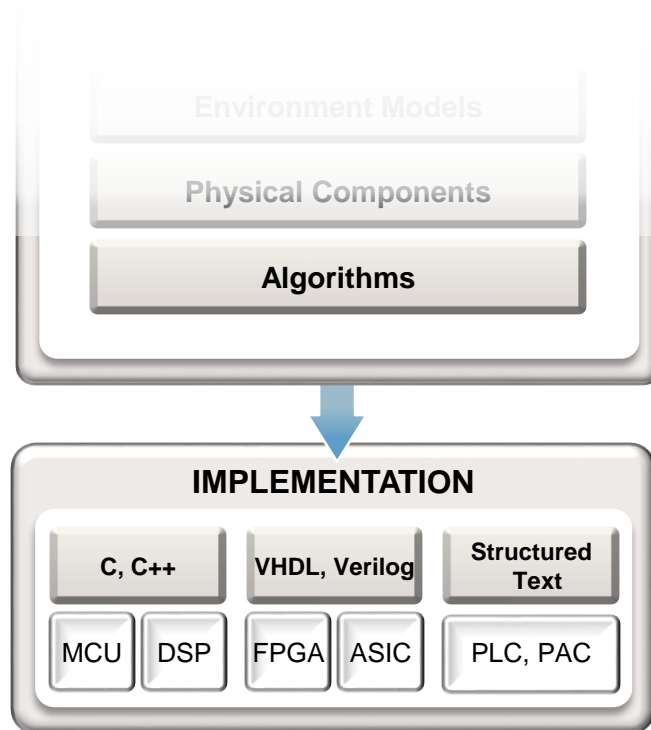
## CAPABILITIES

- Refine model description
- Add fixed point, timing, component interface details

## BENEFITS

- Fast design exploration
- Design optimization
- Find flaws before implementation

# Model-Based Design: Implementation



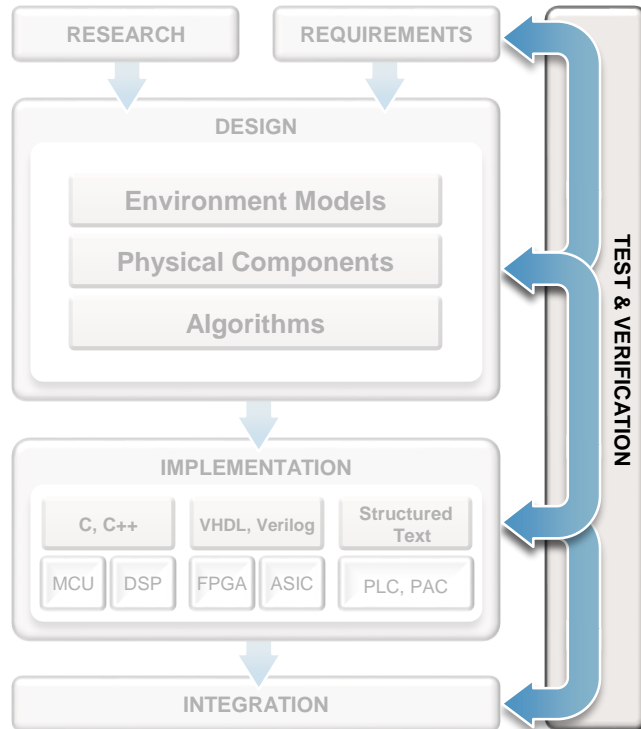
## CAPABILITIES

- Rapid Prototyping
- Automatic Code Generation:
  - C/C++
  - HDL
  - PLC

## BENEFITS

- Eliminate hand-coding
- Eliminate hand-code errors
- Hardware target portability
- Better testability and reuse
- Bridge between domain, software, and hardware knowledge experts

# Model-Based Design: Test and Verification



## CAPABILITIES

- Model Verification
- Software Verification
- Hardware-in-Loop
- Test and Measurement

## BENEFITS

- Detect errors earlier
- Reduce use of physical prototypes
- Implementations that work first time
- Reuse tests throughout development stages

# Building a Wave Farm with Model-Based Design

As engineering tools, MATLAB and Simulink provide **significant value**. They are just as valuable as innovation tools because they enable us to **quickly test ideas** that we would otherwise never try.

— Jonathan Fiévez, Carnegie Wave Energy



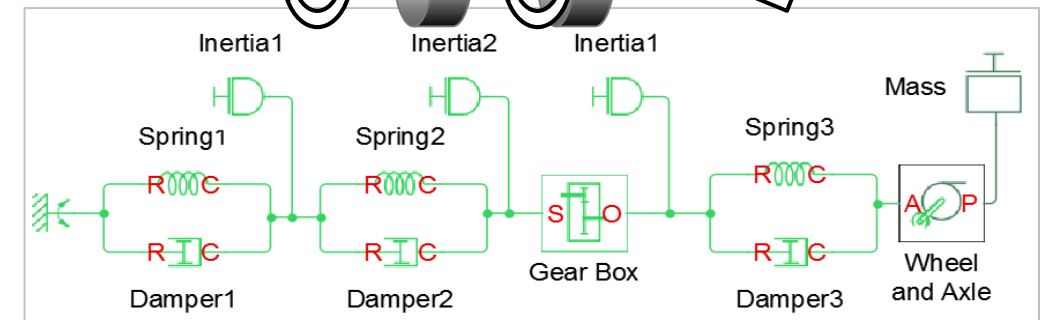
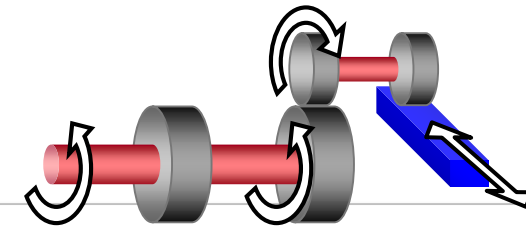
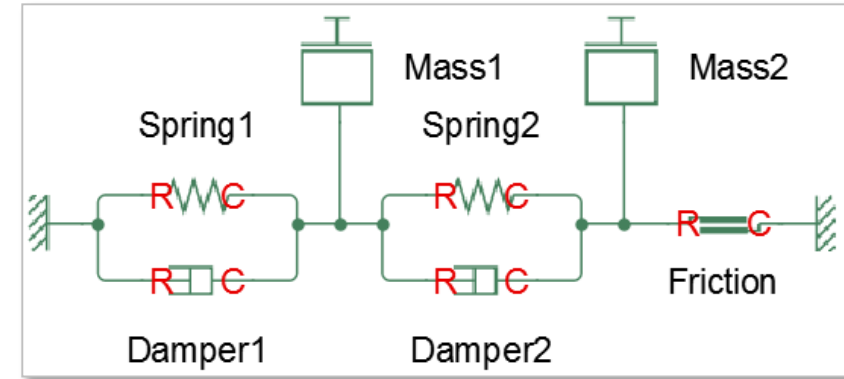
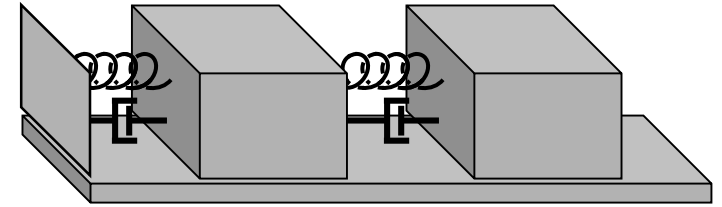
# Outline

- Model-Based Design Overview
- **Modelling and Design in Simulink**
  - Modelling
    - Physical Systems
    - Control logic
  - Simulation
    - System-level optimisation
    - Verification of design changes
- Summary

# Outline

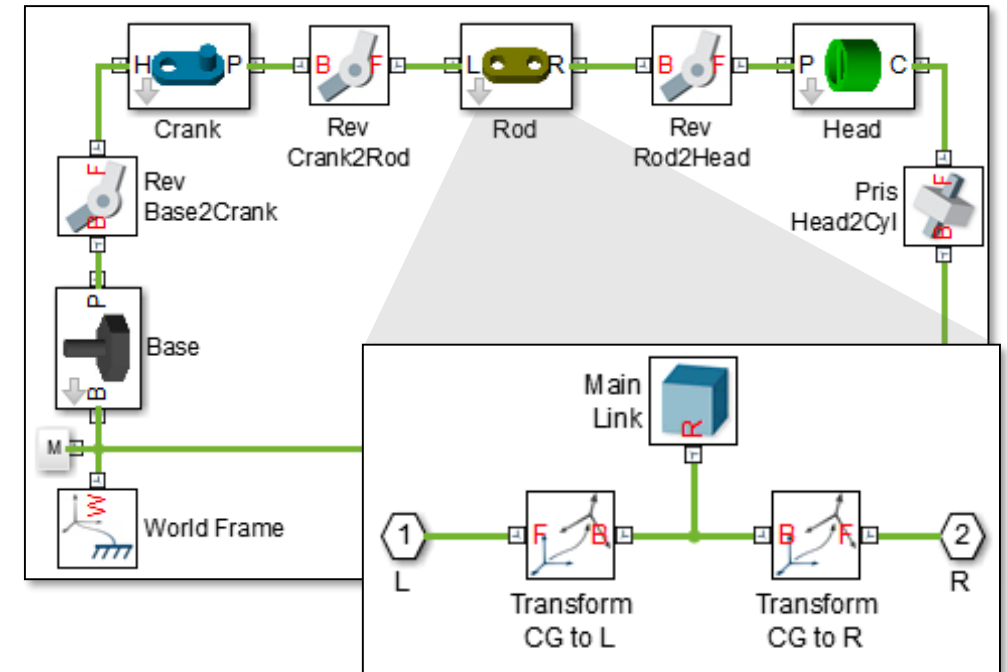
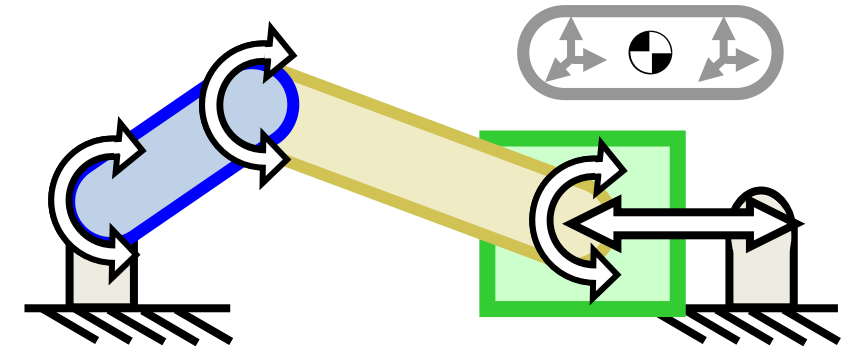
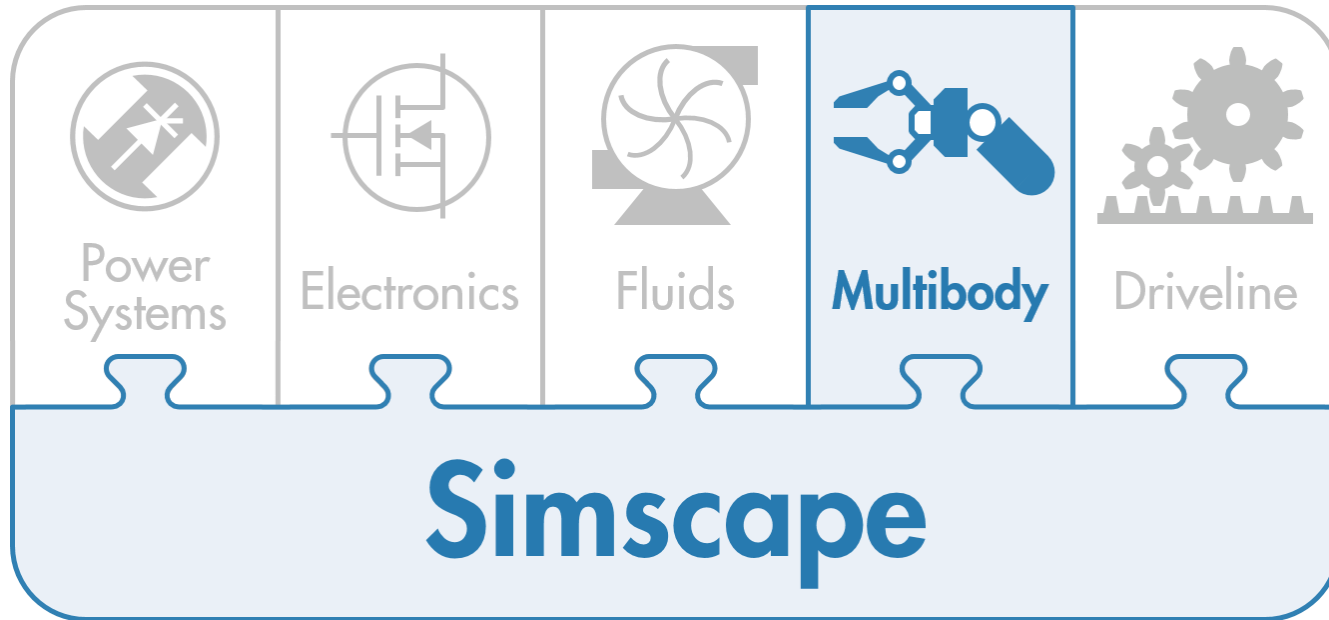
- Model-Based Design Overview
- Modelling and Design in Simulink
  - Modelling
    - Physical Systems
    - Control logic
  - Simulation
    - System-level optimisation
    - Verification of design changes
- Summary

# Mechanical System Modelling

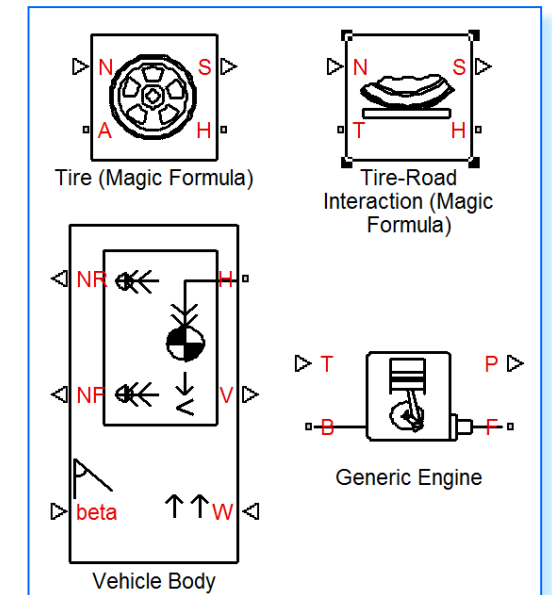
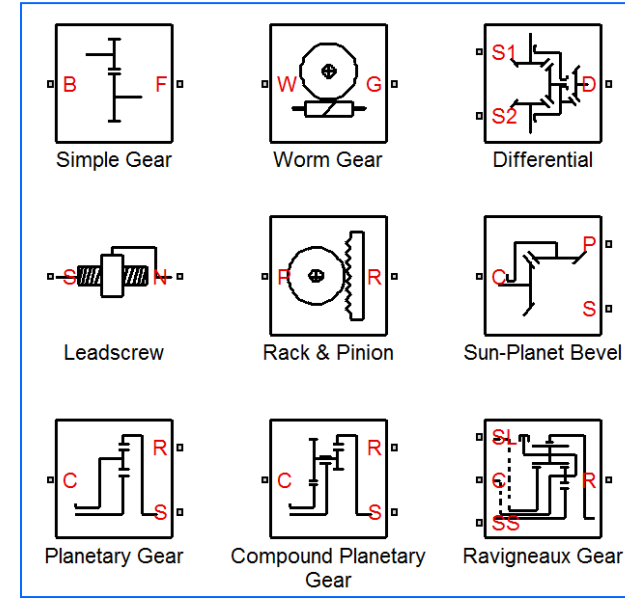
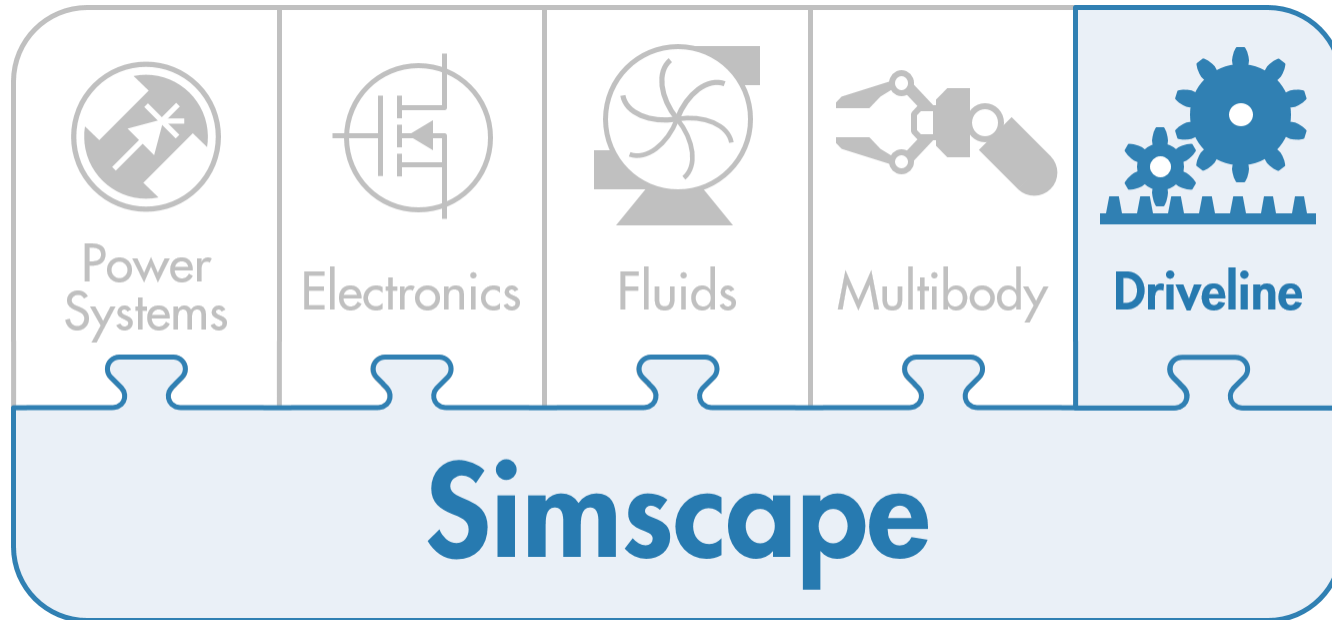




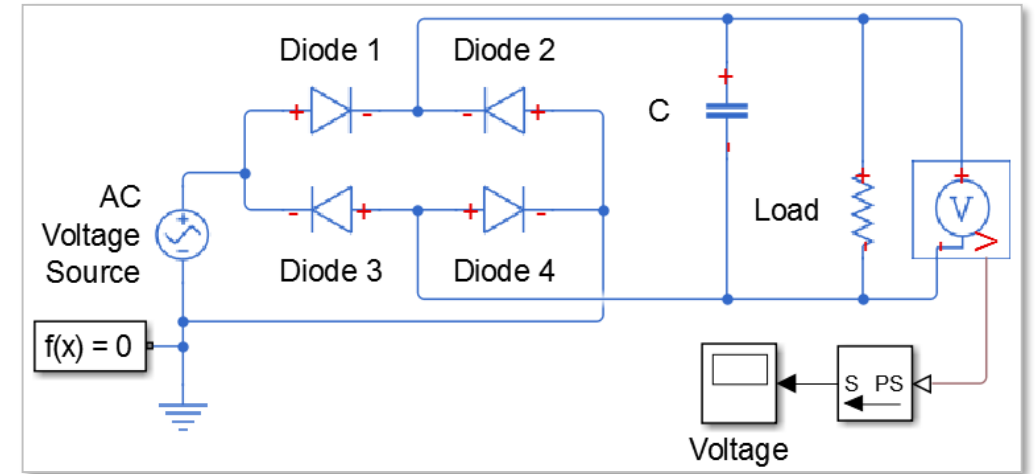
# Mechanical System Modelling



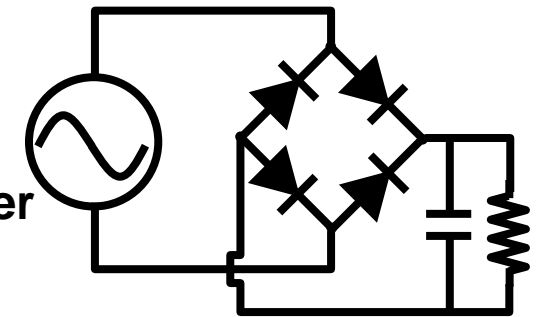
# Mechanical System Modelling



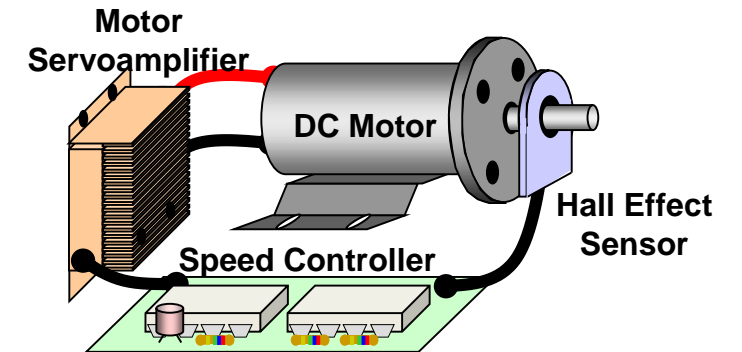
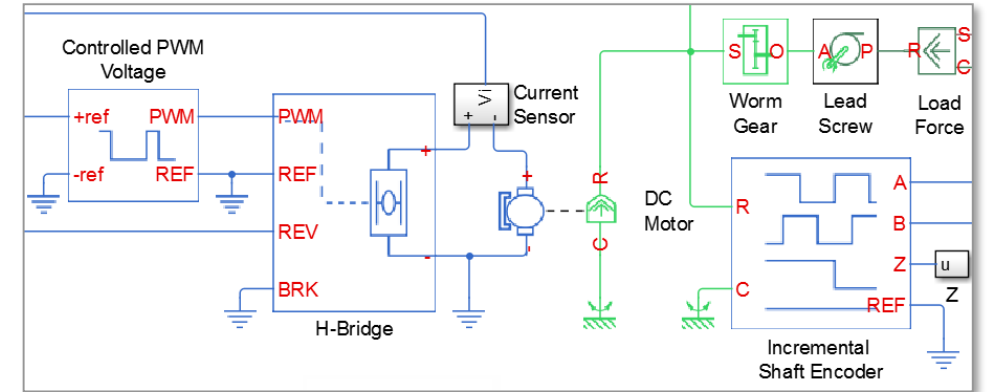
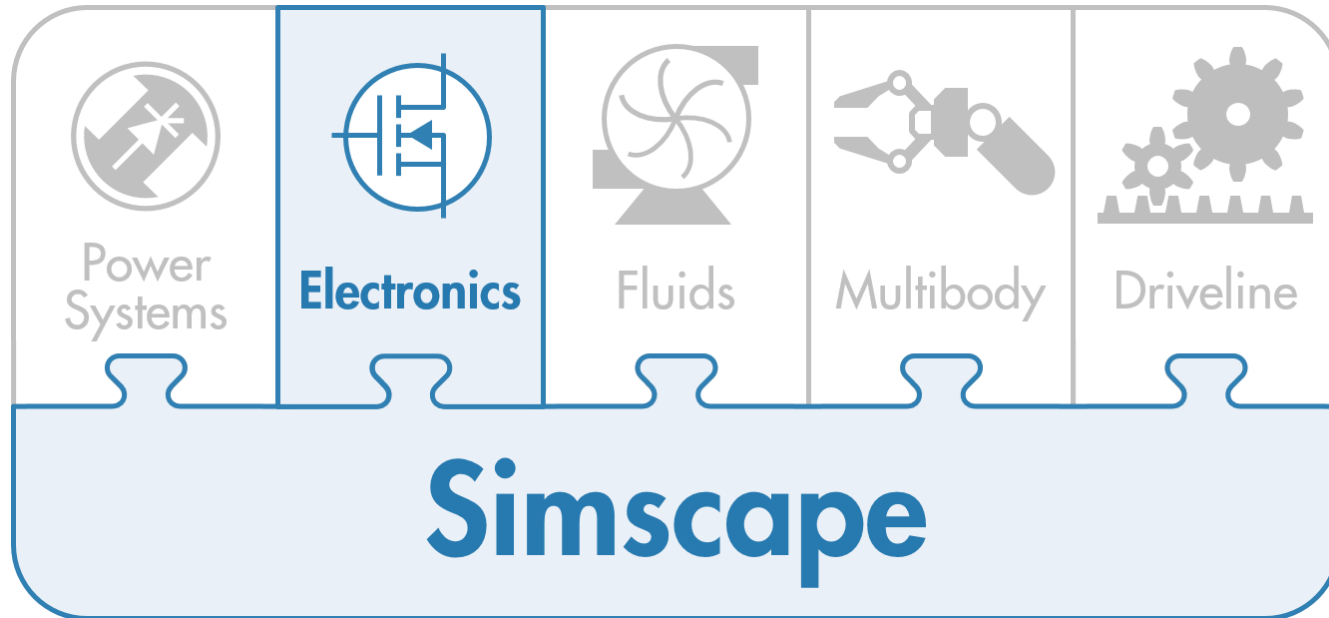
# Electrical System Modelling



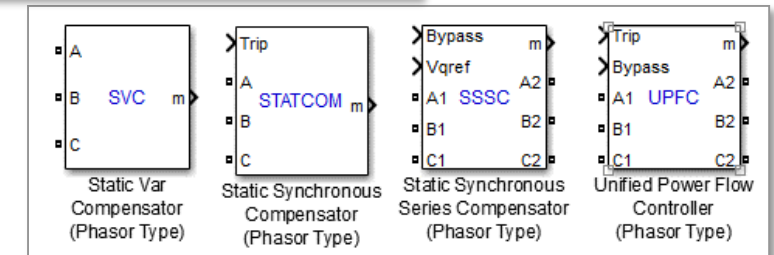
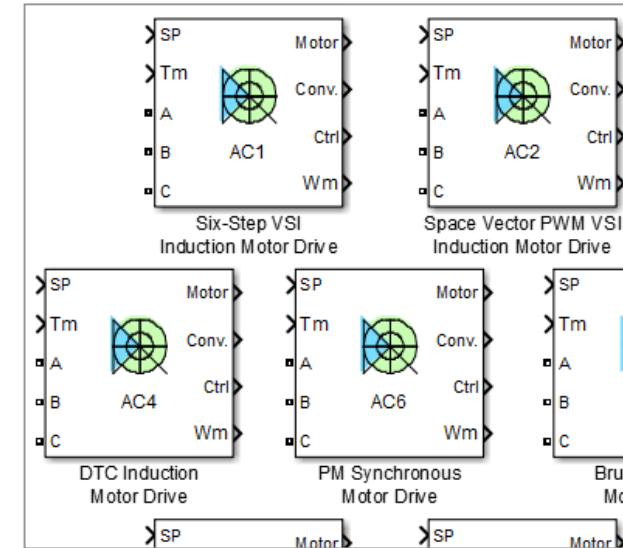
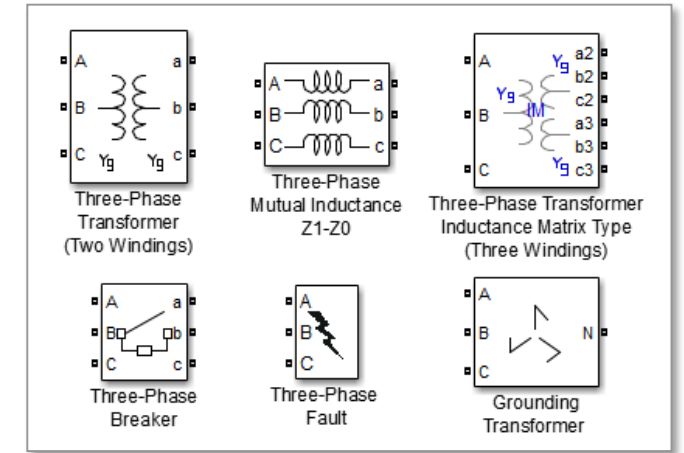
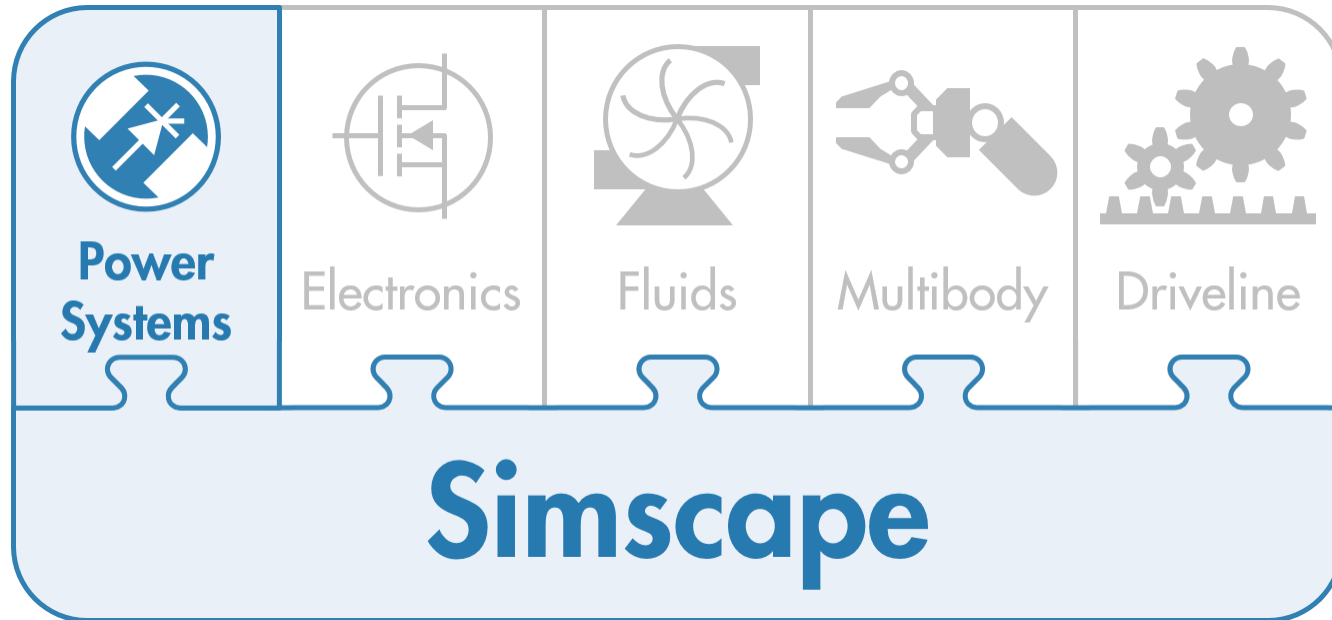
**Bridge Rectifier  
(AC to DC)**



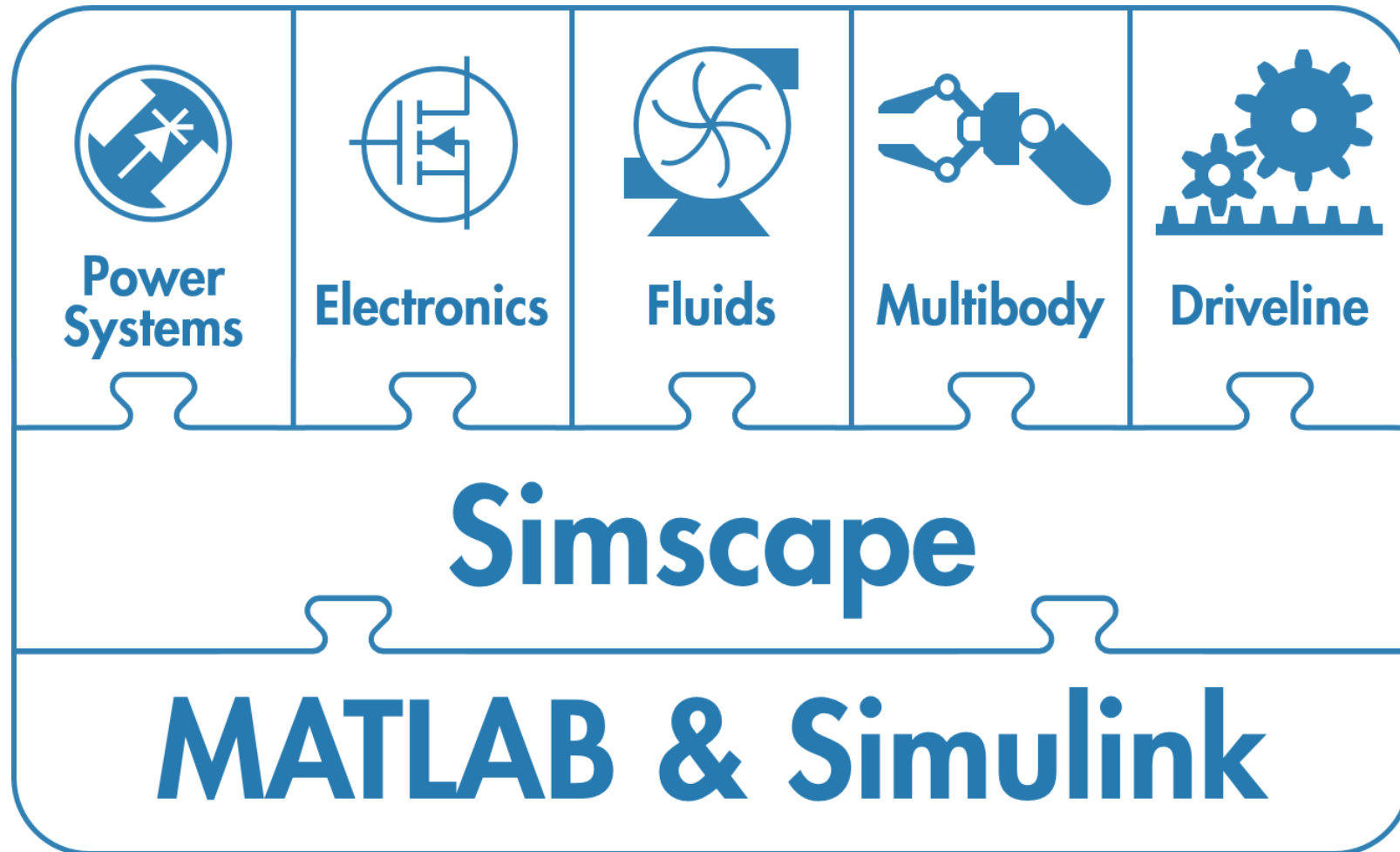
# Electrical System Modelling



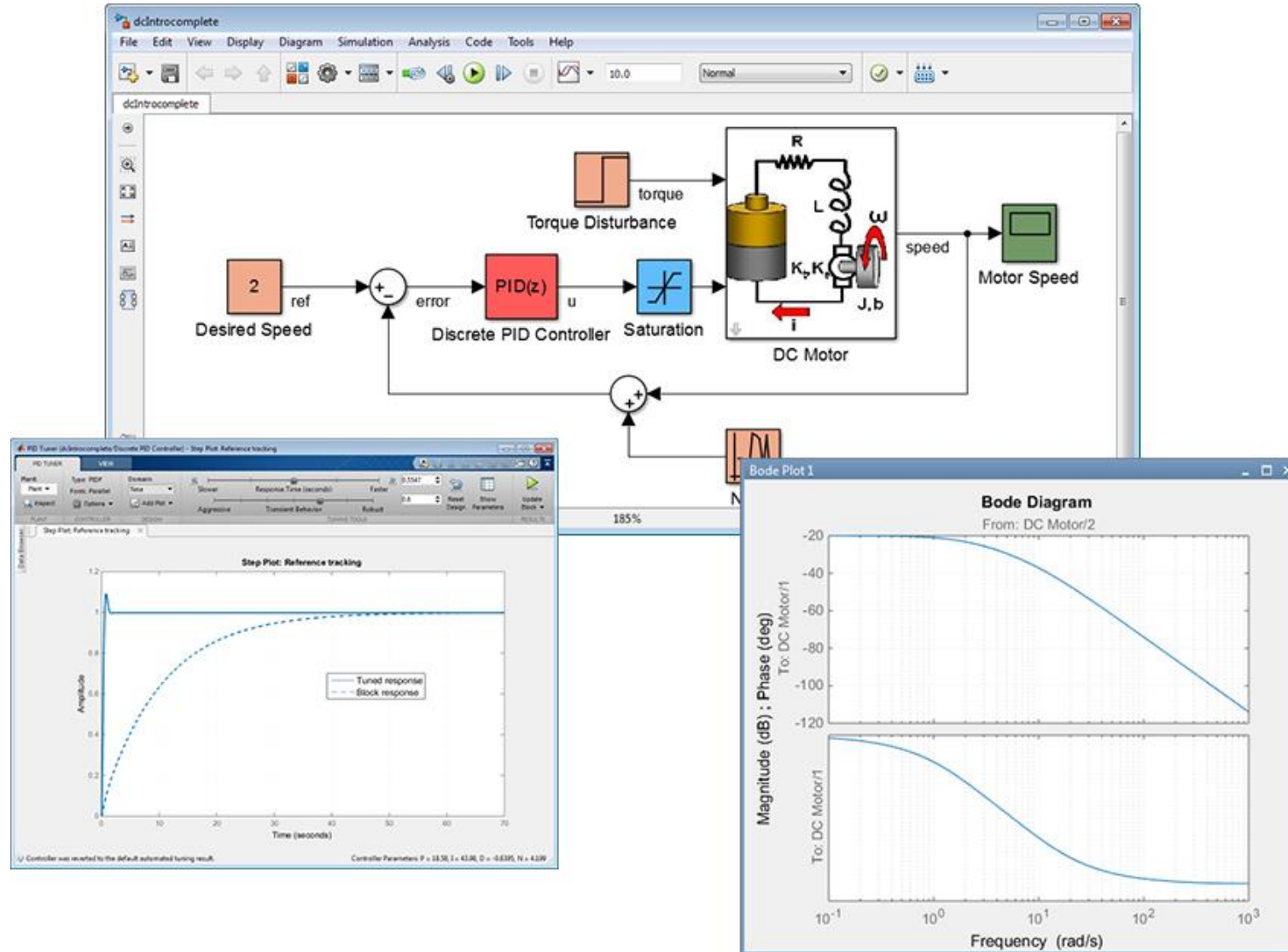
# Electrical System Modelling



# Multi-Domain Modelling of Physical Systems

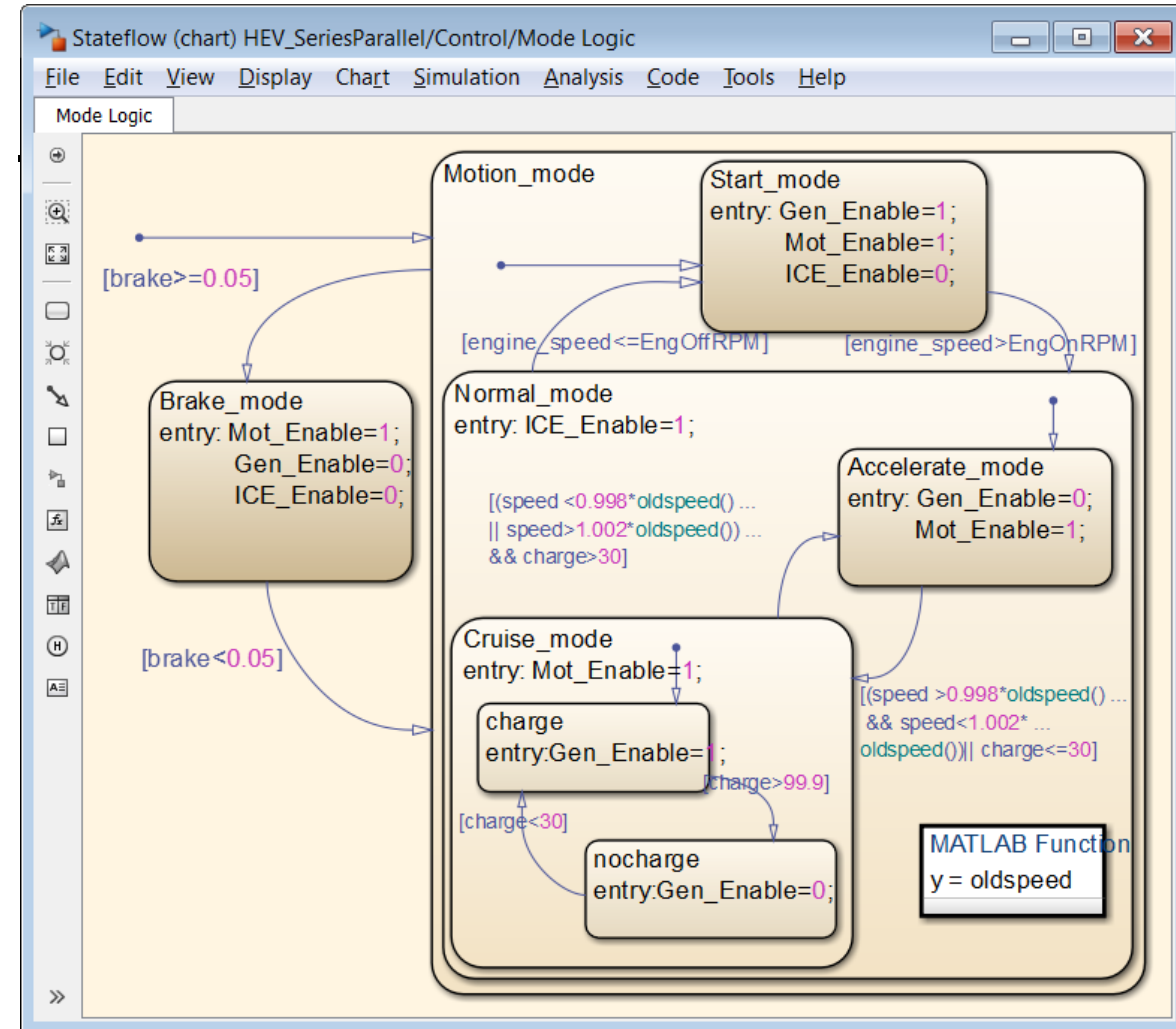


# Control System Design



# Defining System Mode Logic

- Define mode logic using state machine in Stateflow
- Generate production code directly from model



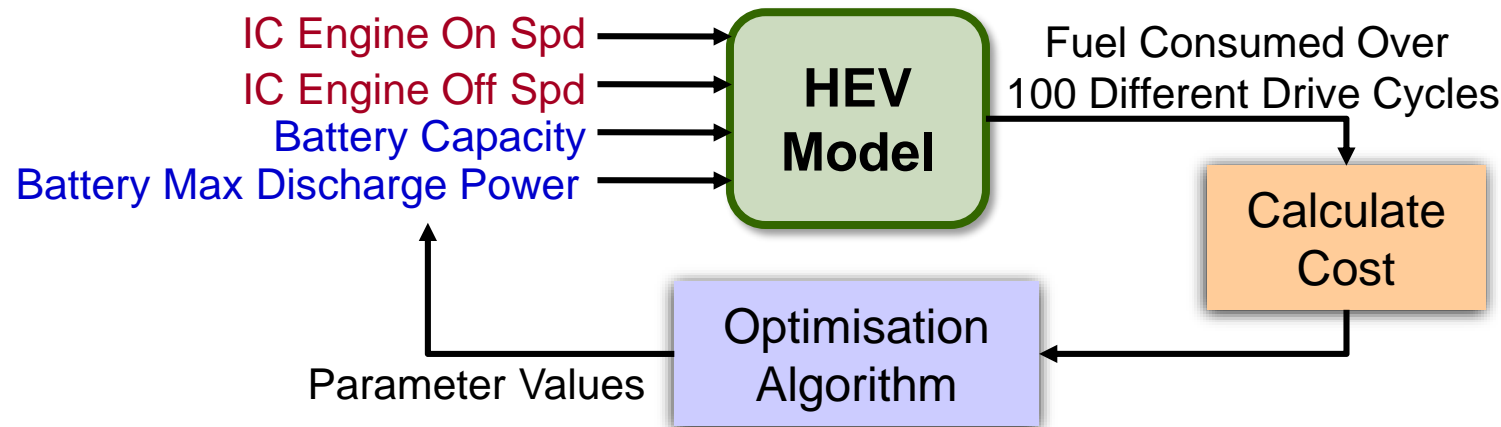
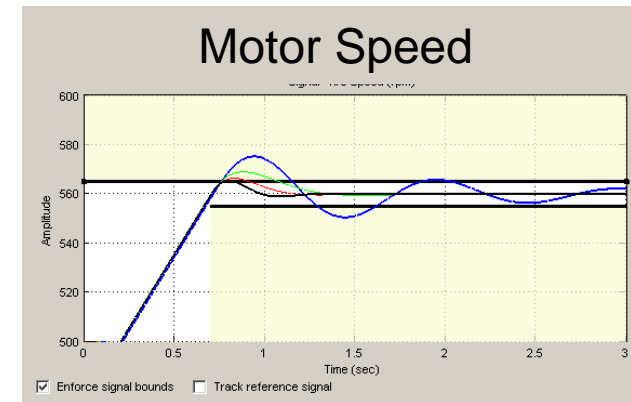
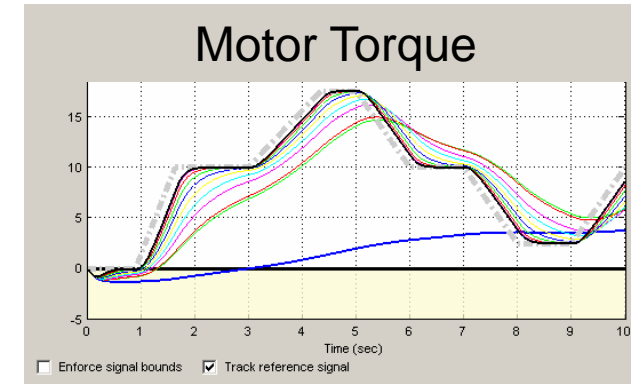


# Outline

- Model-Based Design Overview
- Modelling and Design in Simulink
  - Modelling
    - Physical Systems
    - Control logic
  - Simulation
    - System-level optimisation
    - Verification of design changes
- Summary

# Optimise Entire System

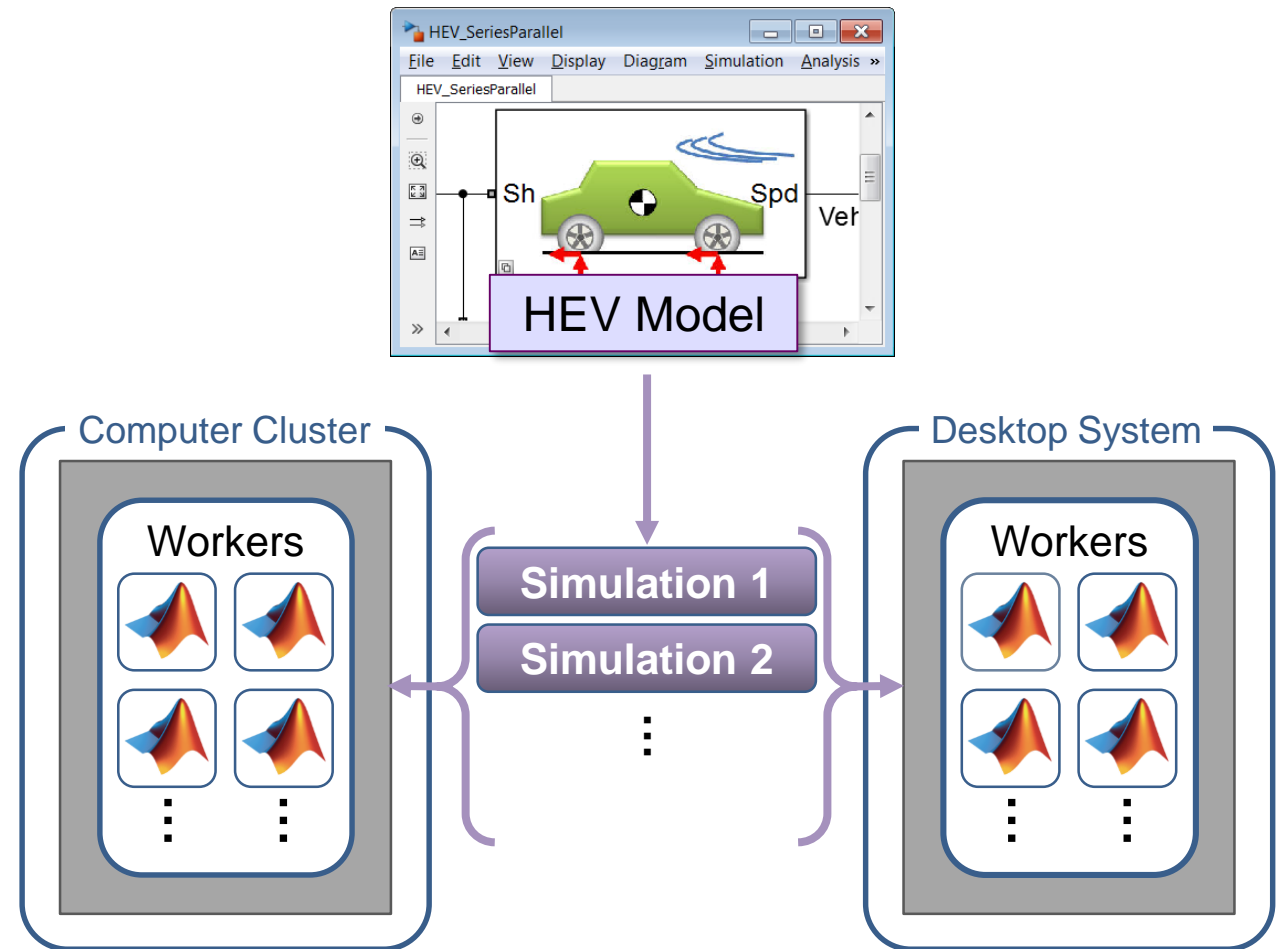
- Use optimisation algorithms to automatically tune parameter values
  - Match response
  - Meet requirements
- Optimise system performance (controller and physical system)



# Distributing Simulations with Parallel Computing

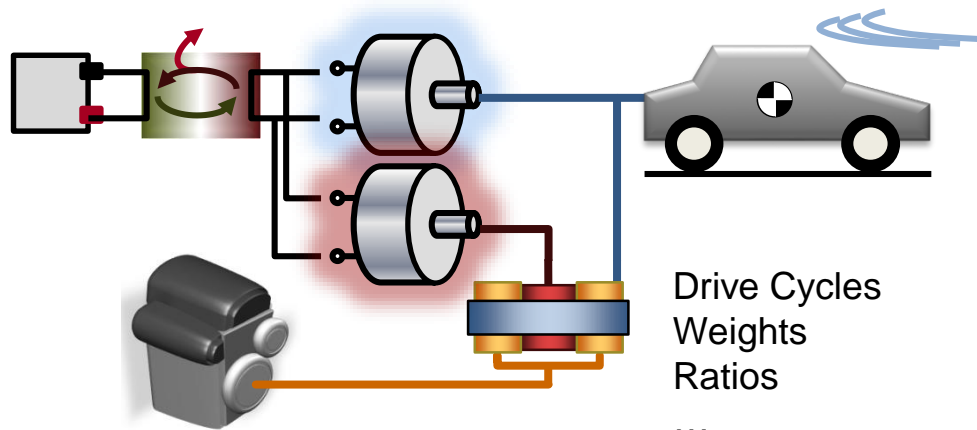
- Simulating in parallel
  - Distribute simulations to multiple cores/processors
  - Dramatic speedup for sets of simulations (parameter sweeps, flight cycles optimisations, and more)

**for** → **parfor**



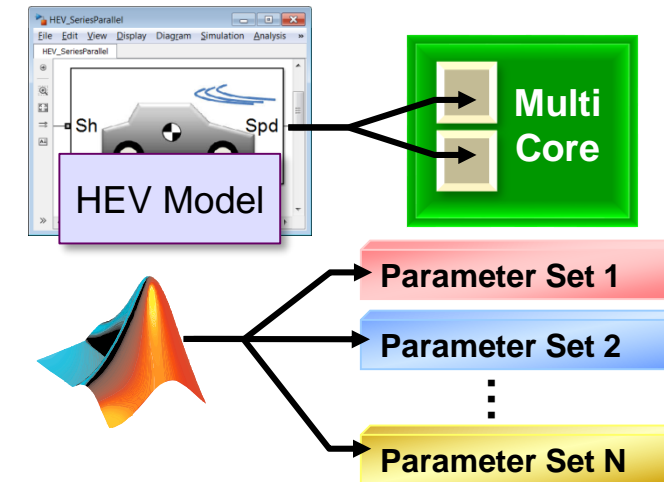
# Shorten Simulation Times With Parallel Computing

## Model:



**Problem:** Minimize simulation time to run a parameter sweep on the HEV model

**Solution:** Use [Parallel Computing Toolbox](#) to speed up the sweep

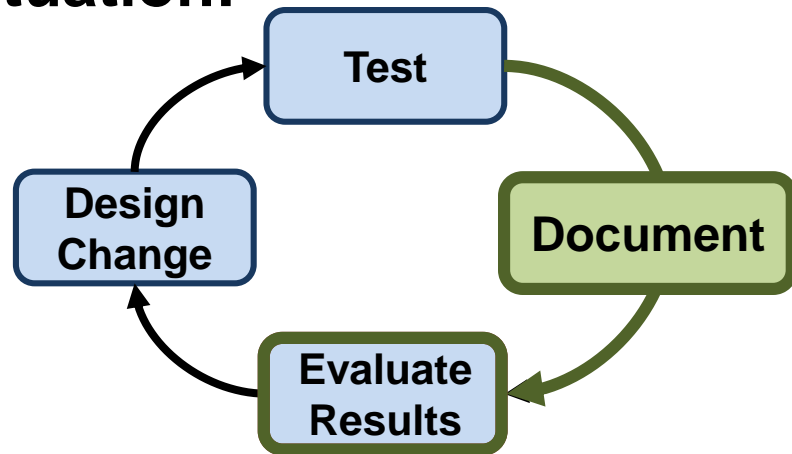


# Outline

- Model-Based Design Overview
- Modelling and Design in Simulink
  - Modelling
    - Physical Systems
    - Control logic
  - Simulation
    - System-level optimisation
    - Verification of design changes
- Summary

# Automatically Run Tests And Document Results

**Situation:**

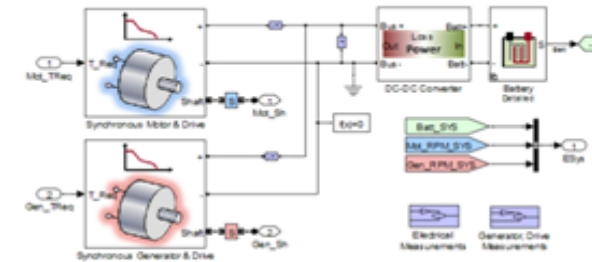


**Problem:** Evaluate test results quickly to make design changes and document the results

**Solution:** Use [Simulink Report Generator](#) to automatically document tests and results

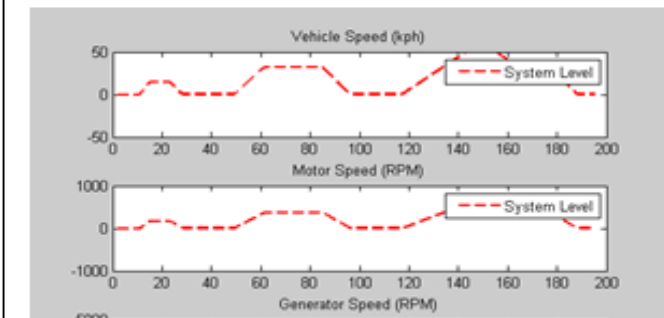
## HEV Test Report

### Chapter 1. System Level



### 1. Drive Cycle 1

**Figure 1.1. Speeds From Drive Cycle 1**



# Outline

- Model-Based Design Overview
- Modelling and Design in Simulink
  - Modelling
    - Physical Systems
    - Control logic
  - Simulation
    - System-level optimisation
    - Verification of design changes
- Summary

## Key Points

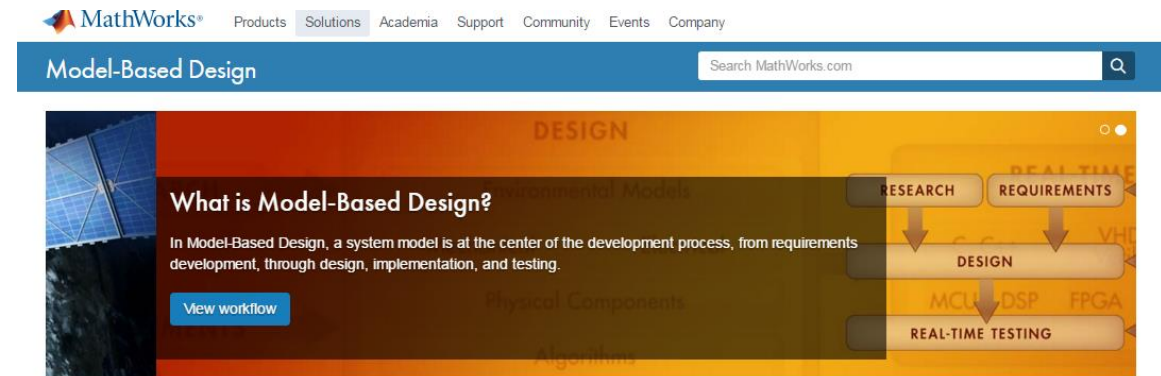
- Simulink is a multi-domain modelling and simulation environment facilitating Model-Based Design
- Optimise the system-level performance
- Accelerate your development
  - Speed up simulations using Parallel Computing Toolbox
  - Speed-up processes using Simulink Report Generator



# Call to Action

## Learn more about Model-Based Design with Simulink

- Explore our [website](http://au.mathworks.com)
  - [au.mathworks.com](http://au.mathworks.com)
  
- Contact me:
  - Ruth-Anne Marchant
    - [ruth-anne.marchant@mathworks.com.au](mailto:ruth-anne.marchant@mathworks.com.au)



### Why Use Model-Based Design?

Model-Based Design is transforming the way engineers and scientists work by moving design tasks from the lab and field to the desktop.

When software and hardware implementation requirements are included, such as fixed-point and timing behavior, you can **automatically generate code** for embedded deployment and create test benches for **system verification**, saving time and avoiding the introduction of manually coded errors.

Use Model-Based Design with **MATLAB®** and **Simulink®** to improve product quality and reduce development time by 50% or more.



Model-Based Design of Safety-Critical Avionics Systems (Highlights)



Weinmann Develops Life-Saving Transport Ventilator



Alstom Grid Develops HVDC Transmission Control System

With Model-Based Design, you can:

- Use a common design environment
- Link designs directly to requirements
- Integrate testing with design
- Refine algorithms through multidomain simulation
- Automatically generate embedded software code and documentation
- Develop and reuse test suites

[Explore Model-Based Design with Simulink](#)

[Contact sales](#)

[Request a trial](#)

# Q & A