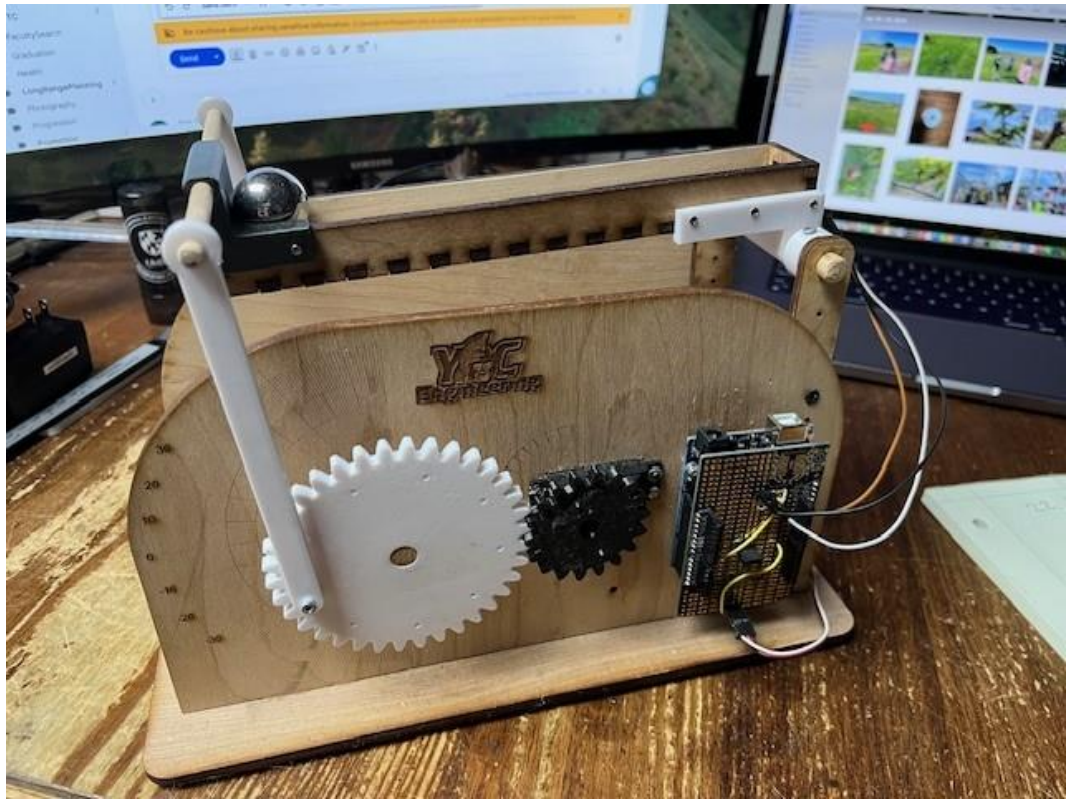
Systems Modeling and Control with Simulink and Simscape

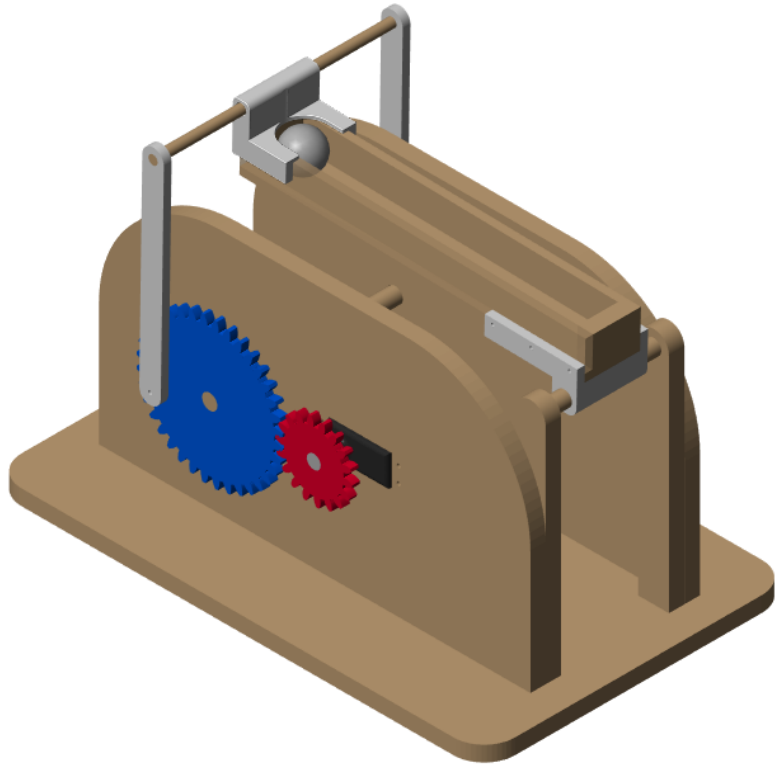
Noah Roberts
Education Application Engineer

Agenda

- Simulink Overview
- Introduction to Model-Based Design
- Various ways of modeling a DC Motor
  - Mathematical Modeling with Simulink
  - Physical Modeling with Simscape
- Control System design
- Automatic Code Generation
- Resources and Support

Great Demo



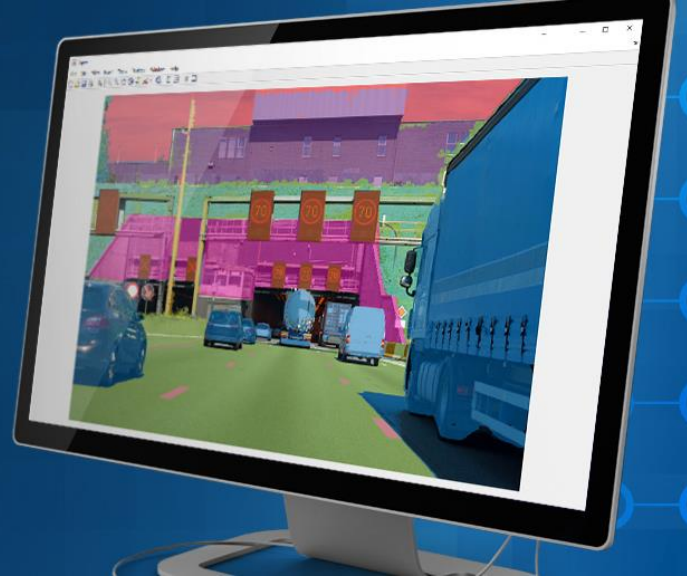


Agenda

- **Simulink Overview**
- Introduction to Model-Based Design
- Various ways of modeling a DC Motor
  - Mathematical Modeling with Simulink
  - Physical Modeling with Simscape
- Control System design
- Automatic Code Generation
- Resources and Support

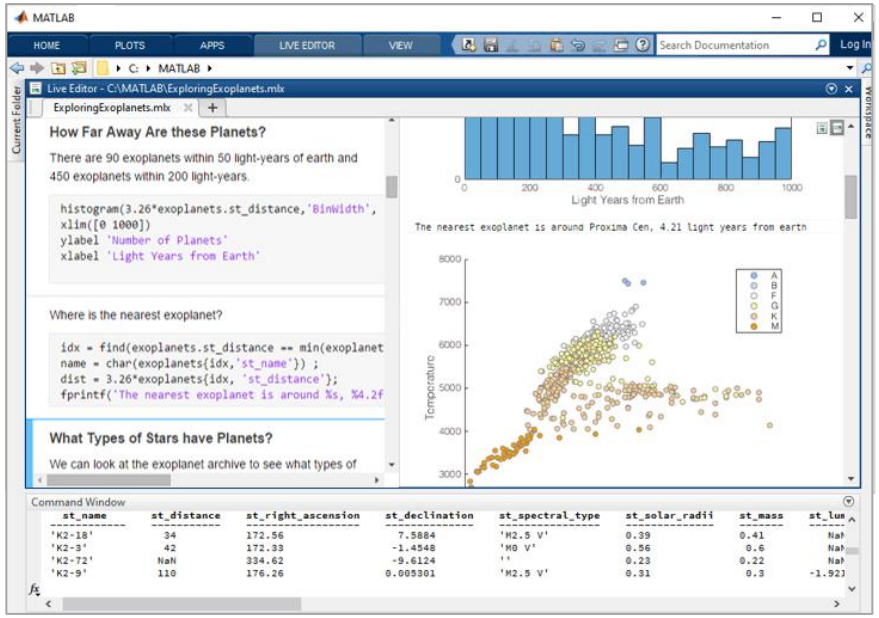
Our Products

MATLAB® & SIMULINK®



- **MATLAB** - Programming environment for algorithm development, data analysis, visualization, and numeric computation.
- **Simulink** - Block diagram environment for simulation and Model-Based Design of multidomain and embedded engineering systems.
- **130+ add-on products** for specialized tasks.

Computer Vision Toolbox



SIMULINK®

Simulation and Model-Based Design

Model and simulate your system

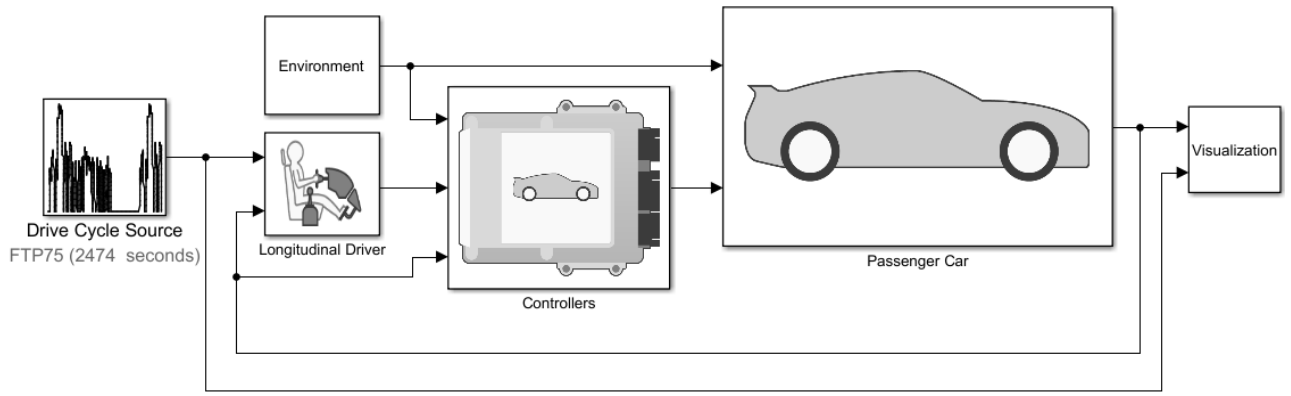
- Use one multi-domain environment
- Model the system under test and the plant
- Simulate closed-loop system behavior

Test early and often

- Test your system under all conditions
- Validate your design with real-time testing
- Trace from requirements to design to code

Automatically generate code

- Generate production-quality C and HDL code
- Deploy directly to embedded processors or FPGA's/ASIC's

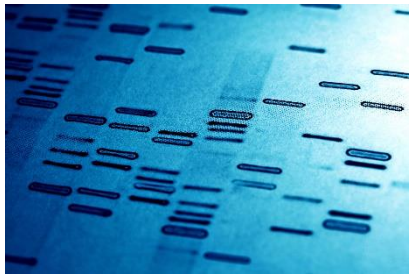


Where are Simulink and Simscape used?

Aerospace and Defense

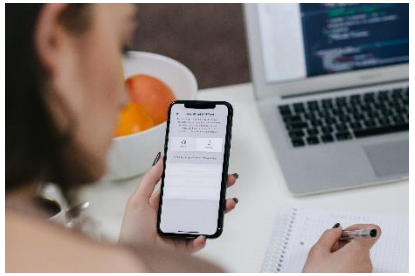
Automotive

Biological Sciences



Biotech and Pharmaceutical

Communications



Electronics

Energy Production

Financial Services

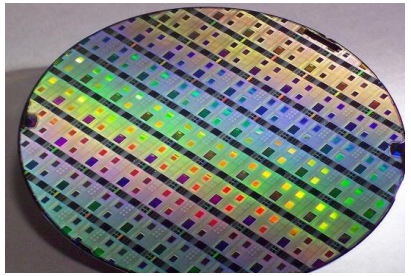
Industrial Machinery

Medical Devices

Metals, Materials, Mining

Neuroscience

Railway Systems



Semiconductors

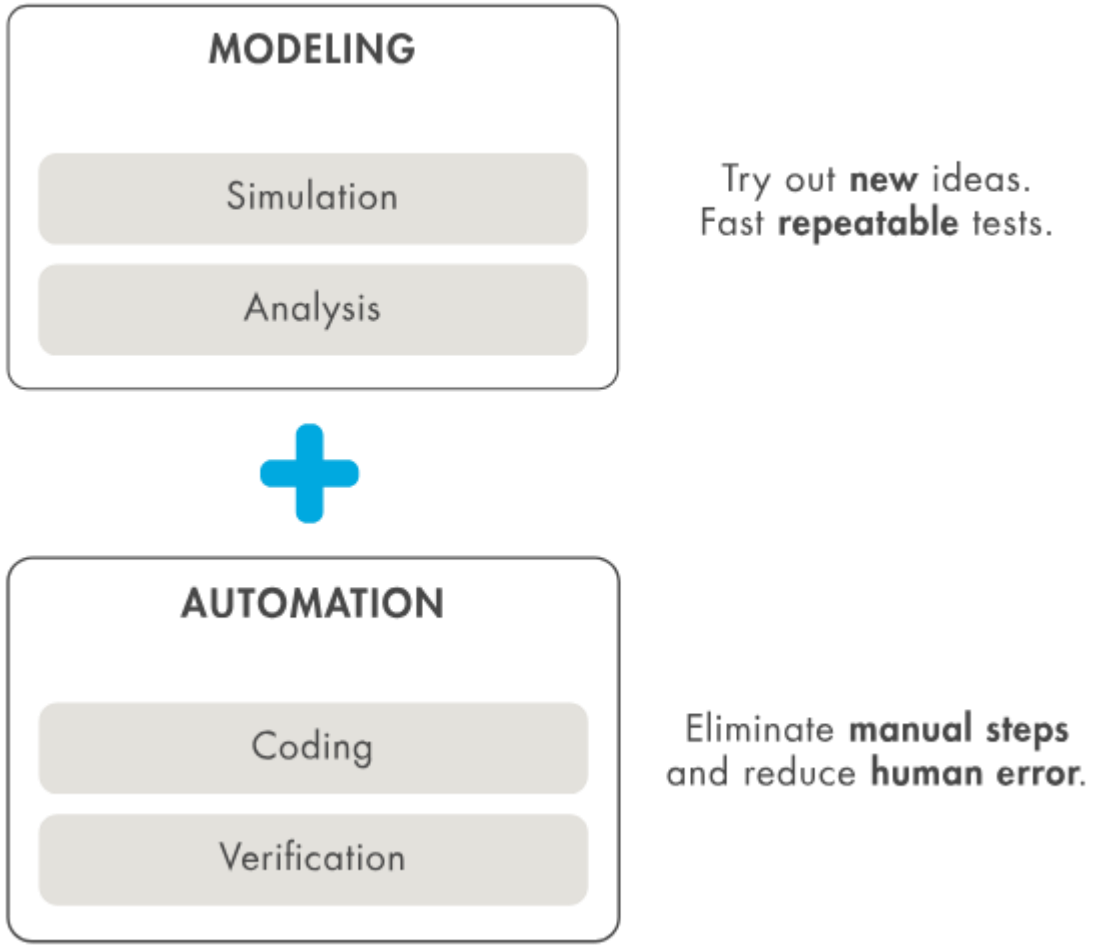
Software and Internet

Agenda

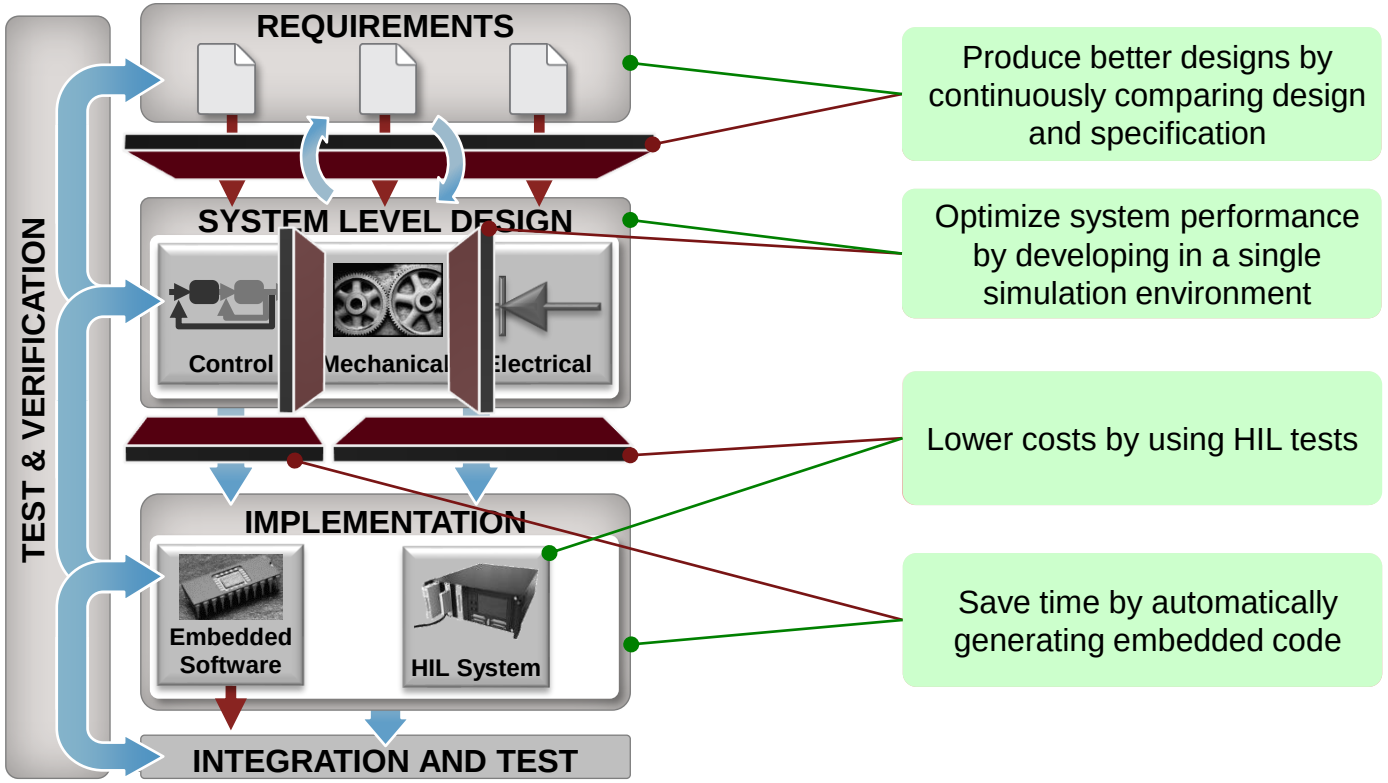
- Simulink Overview
- **Introduction to Model-Based Design**
- Various ways of modeling a DC Motor
  - Mathematical Modeling with Simulink
  - Physical Modeling with Simscape
- Control System design
- Automatic Code Generation
- Resources and Support

What is Model-Based Design?

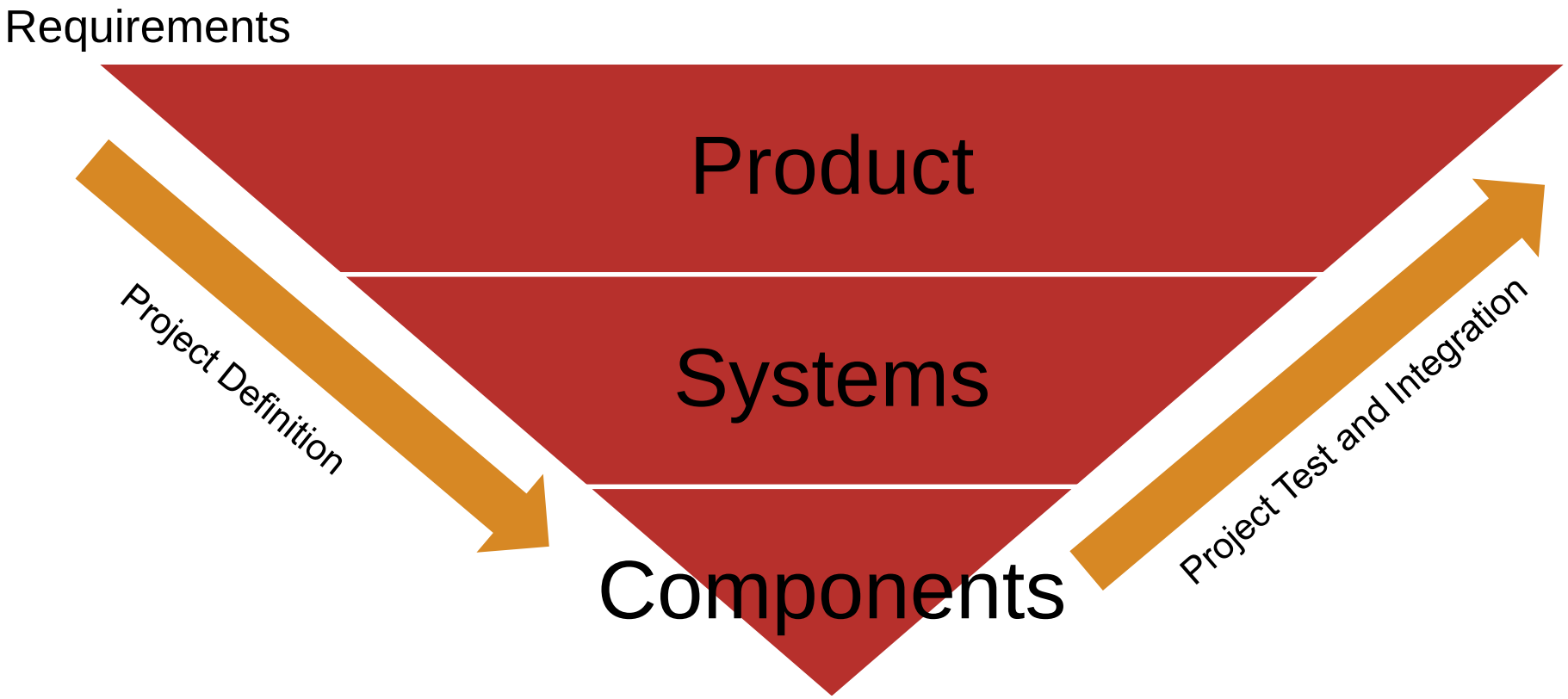
- Creating a virtual representation of a real-world system
  - **Model** new ideas and perform repeatable tests with **simulation**
  - Adjust **design** based on these simulations and **iterate**



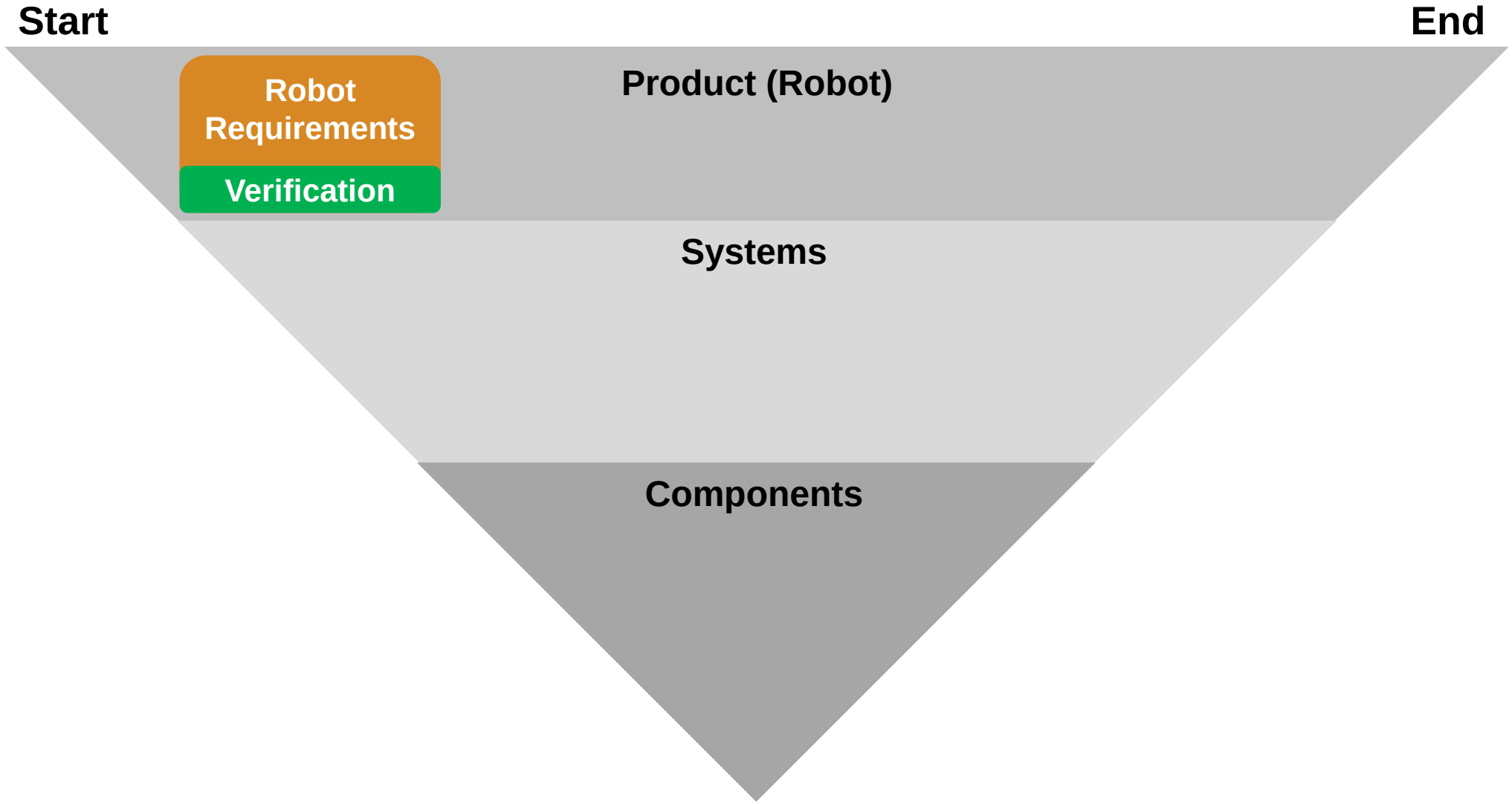
Model-Based Design Process



V Diagram


Requirements
Product
Systems
Components
Project Definition
Project Test and Integration

Justin the Robot MBD Workflow



Justin the Robot MBD Workflow

Start

End

Product (Robot)

Robot Requirements

Verification

Trace

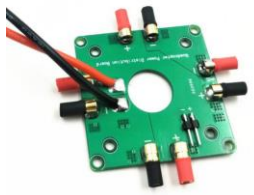
Systems

Manipulators

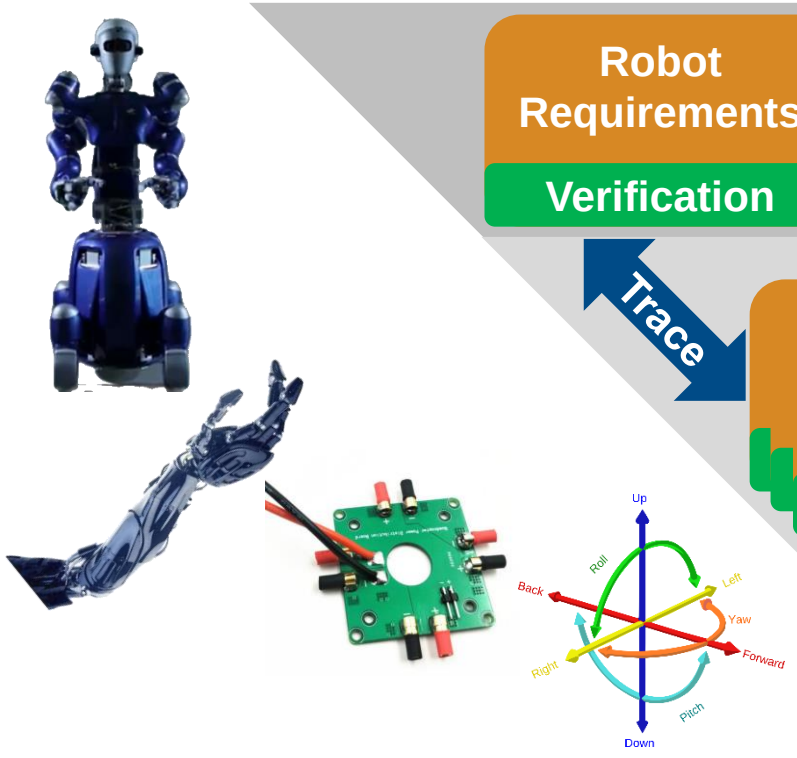
Verification

Components

Justin the Robot MBD Workflow



Justin the Robot MBD Workflow



Start

End

Robot Requirements

Verification

Product (Robot)

Trace

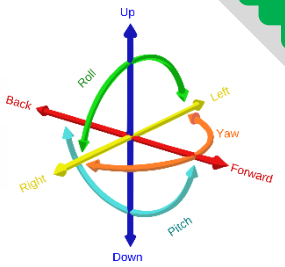
Systems

Motion Controls

Verification

Components

Justin the Robot MBD Workflow



Start

End

Product (Robot)

Robot Requirements

Verification

Trace

Systems

Motion Controls

Verification

Trace

Components

DC Motors

Verification

Justin the Robot MBD Workflow

Start

End

Product (Robot)

Robot Requirements

Verification

Trace

Systems

Motion Controls

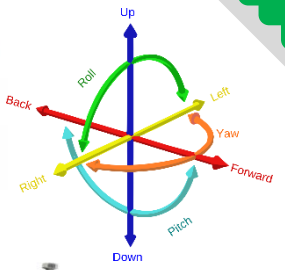
Verification

Trace

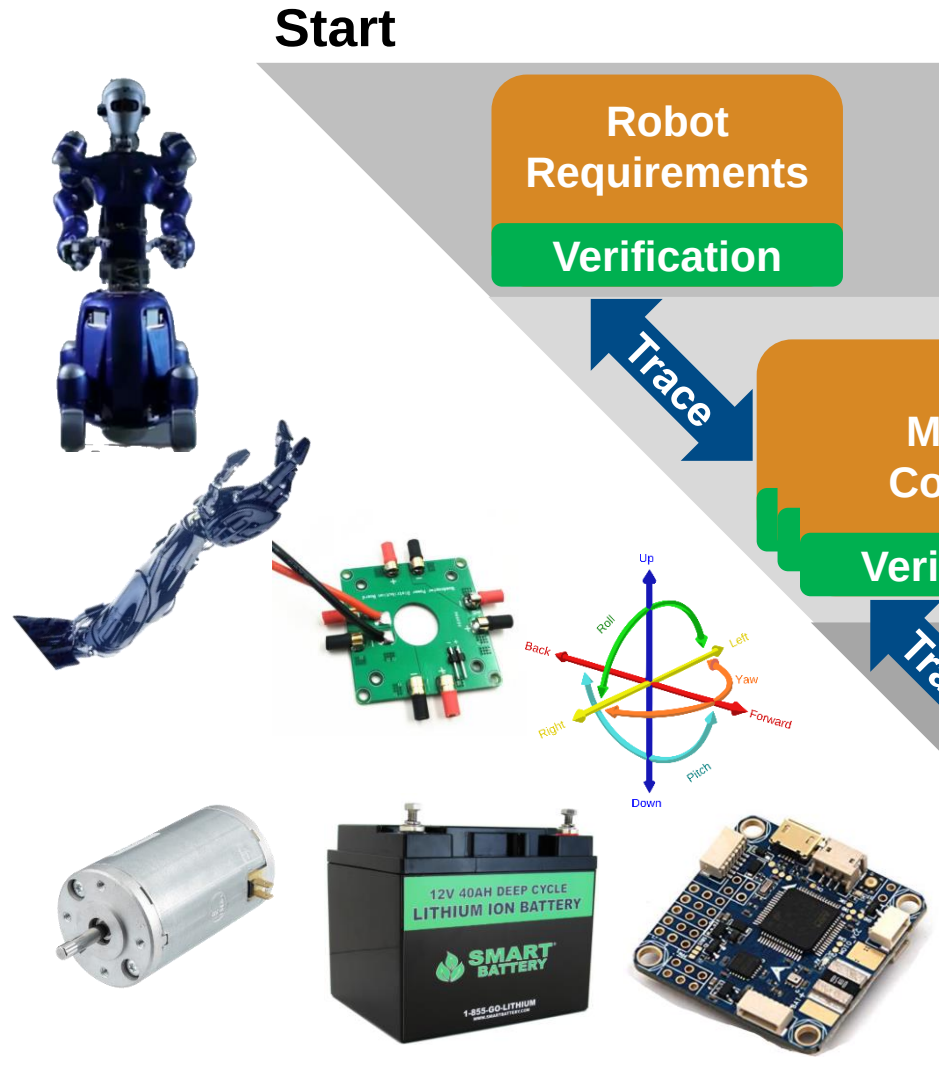
Components

HW (Power Supply)

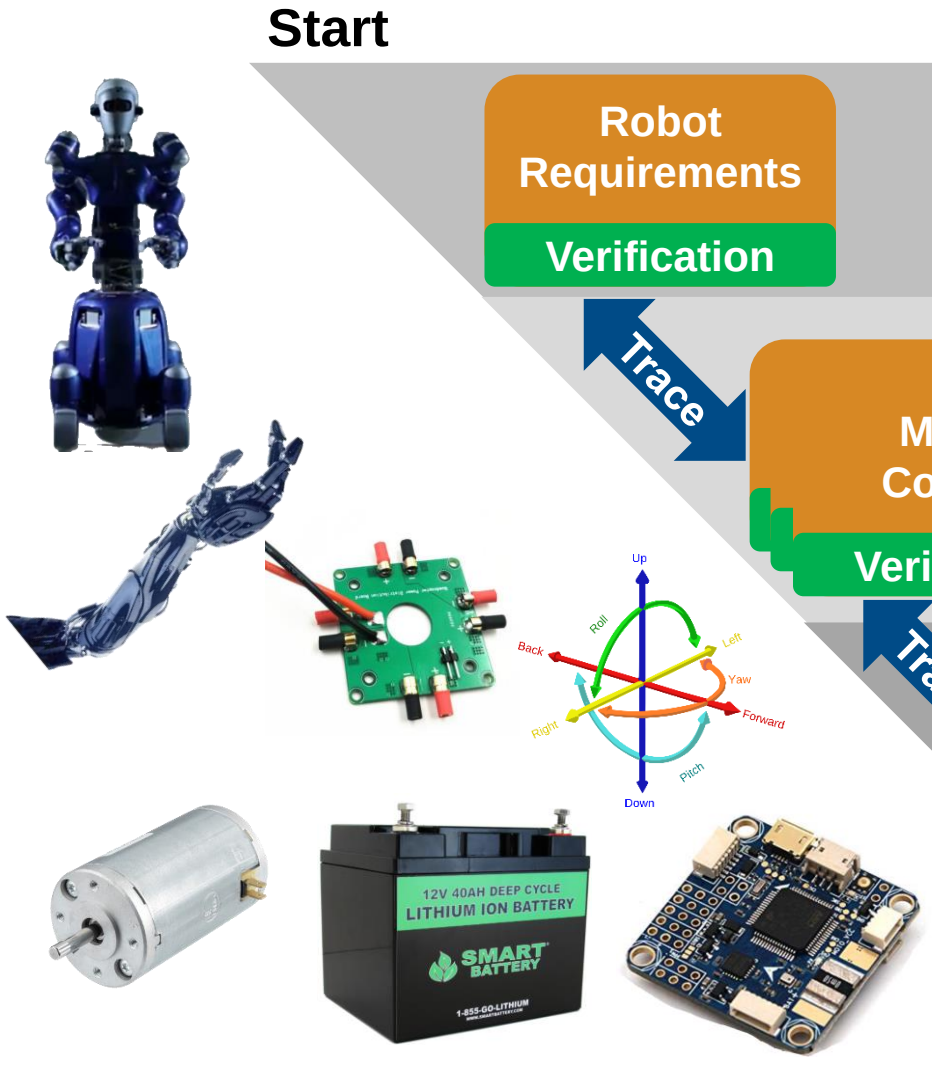
Verification



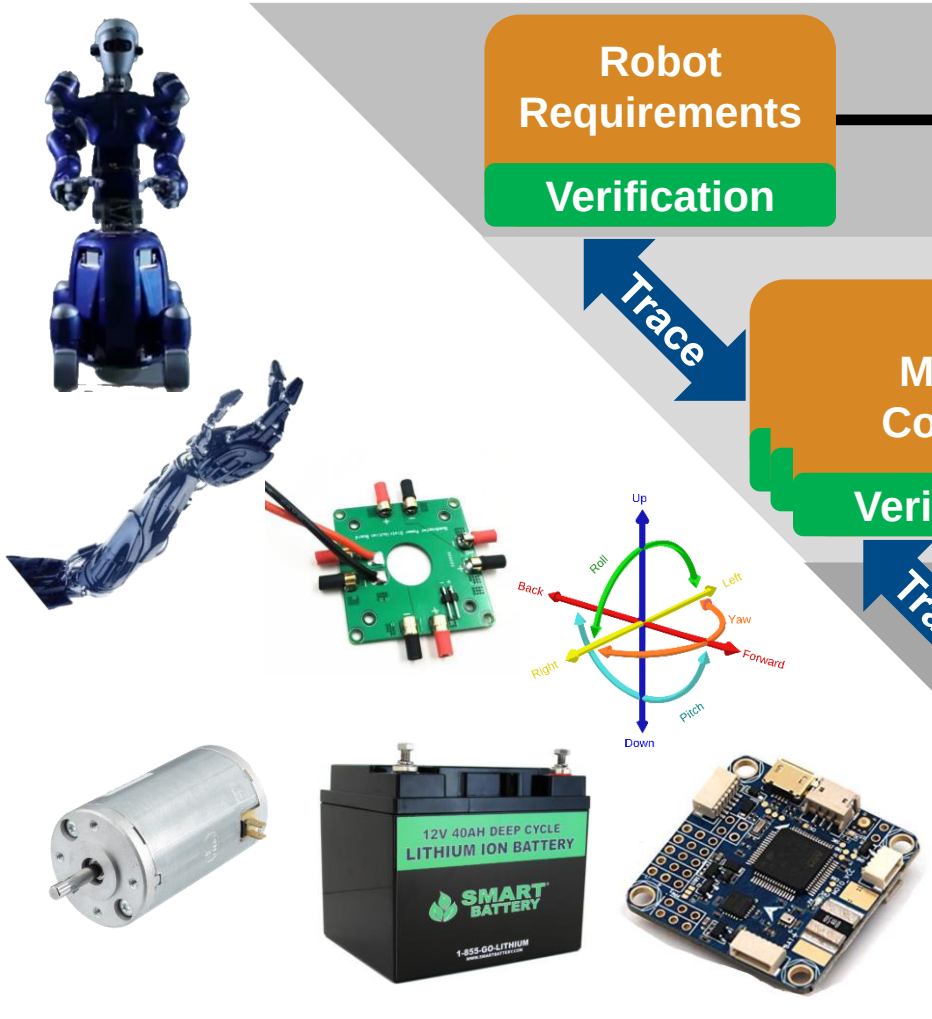
Justin the Robot MBD Workflow



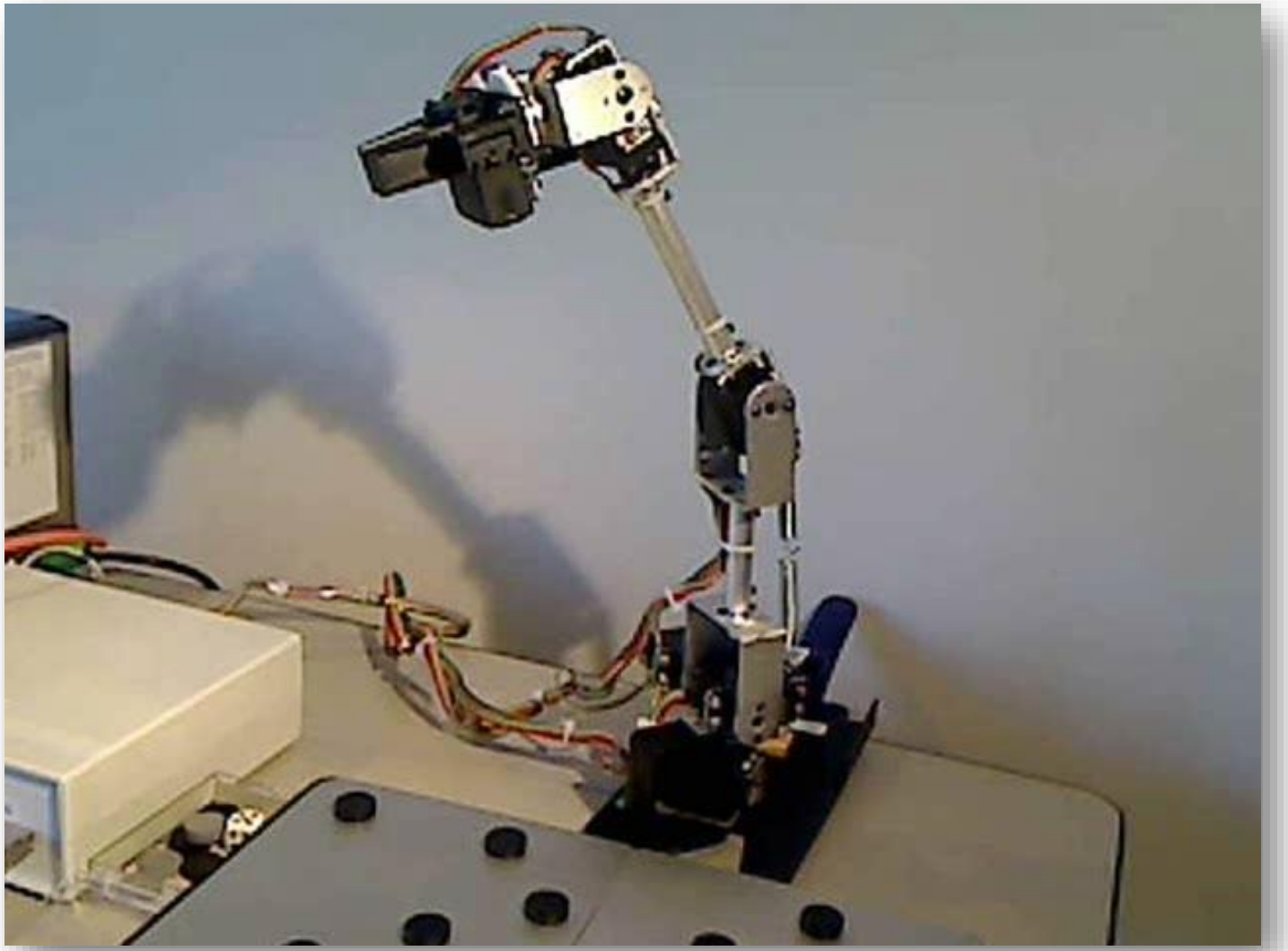
Justin the Robot MBD Workflow



Justin the Robot MBD Workflow

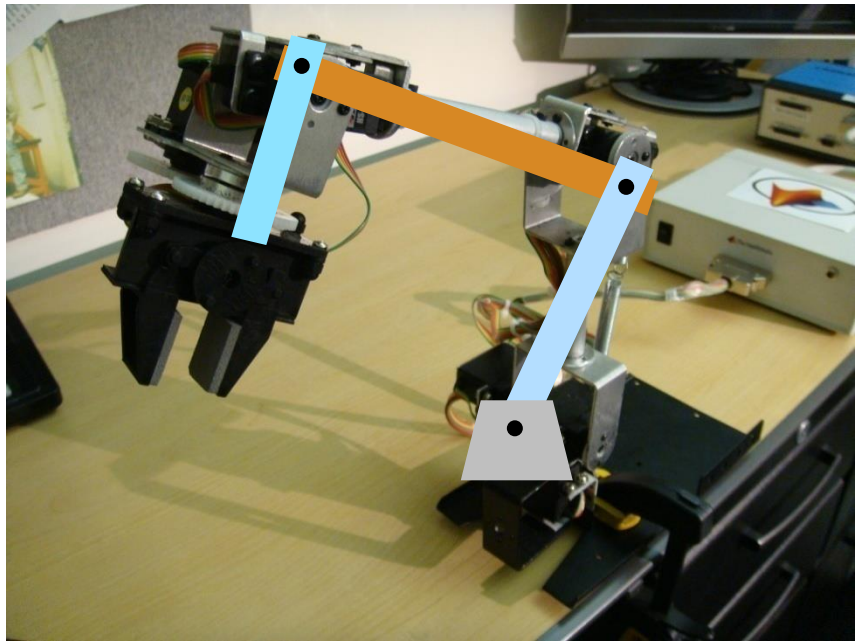


How can you get started?

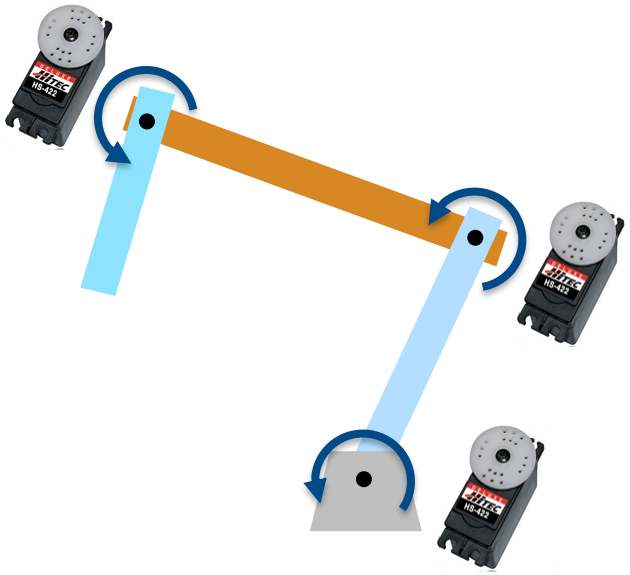


So, where do we really start?

- If possible, break down a big problem into “smaller”(i.e., more manageable) problems
- Use a “divide and conquer” approach



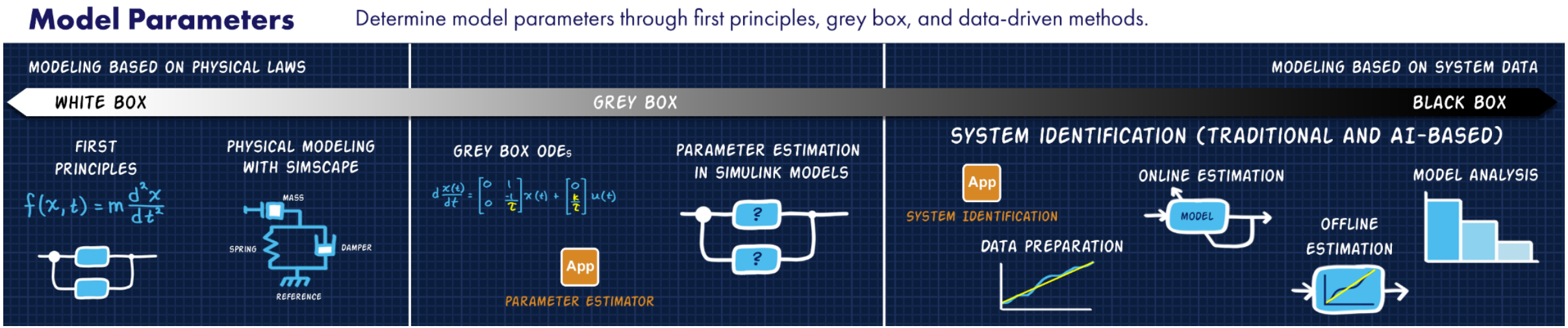
And, for our robot to be of any use, it has to be able to move…



We should include the effects of actuator and sensor dynamics in the simulation model

Different Approaches for Modeling Dynamic Systems

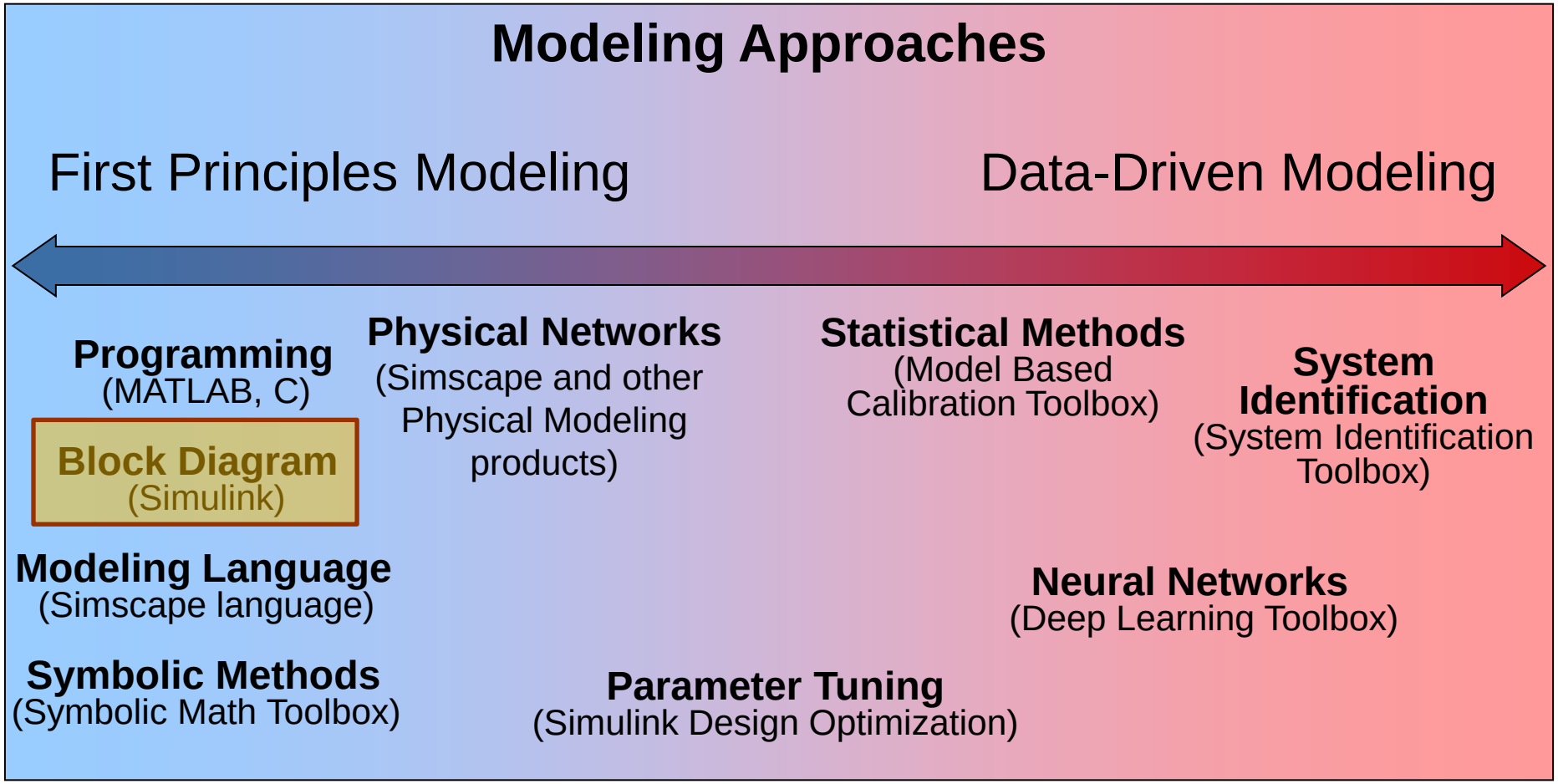
- *How much/what kind of information do we know?*



Agenda

- Simulink Overview
- Introduction to Model-Based Design
- Various ways of modeling a DC Motor
  - **Mathematical Modeling with Simulink**
  - Physical Modeling with Simscape
- Control System design
- Automatic Code Generation
- Resources and Support

Different Approaches for Modeling Dynamic Systems



Modeling a DC Motor in Simulink

$$V = K\omega + iR + L\frac{di}{dt}$$

$$\frac{di}{dt} = \frac{1}{L}(V - K\omega - iR)$$

$$i = \int \frac{1}{L}(V - K\omega - iR)dt$$

Electrical

$$J\frac{d\omega}{dt} = Ki - b\omega - T_{Load}$$

$$\frac{d\omega}{dt} = \frac{1}{J}(Ki - b\omega - T_{Load})$$

$$\omega = \int (Ki - b\omega - T_{Load})dt$$

Mechanical

Modeling a DC Motor in Simulink

Based on its equations:

$$V = K\omega + iR + L\frac{di}{dt}$$

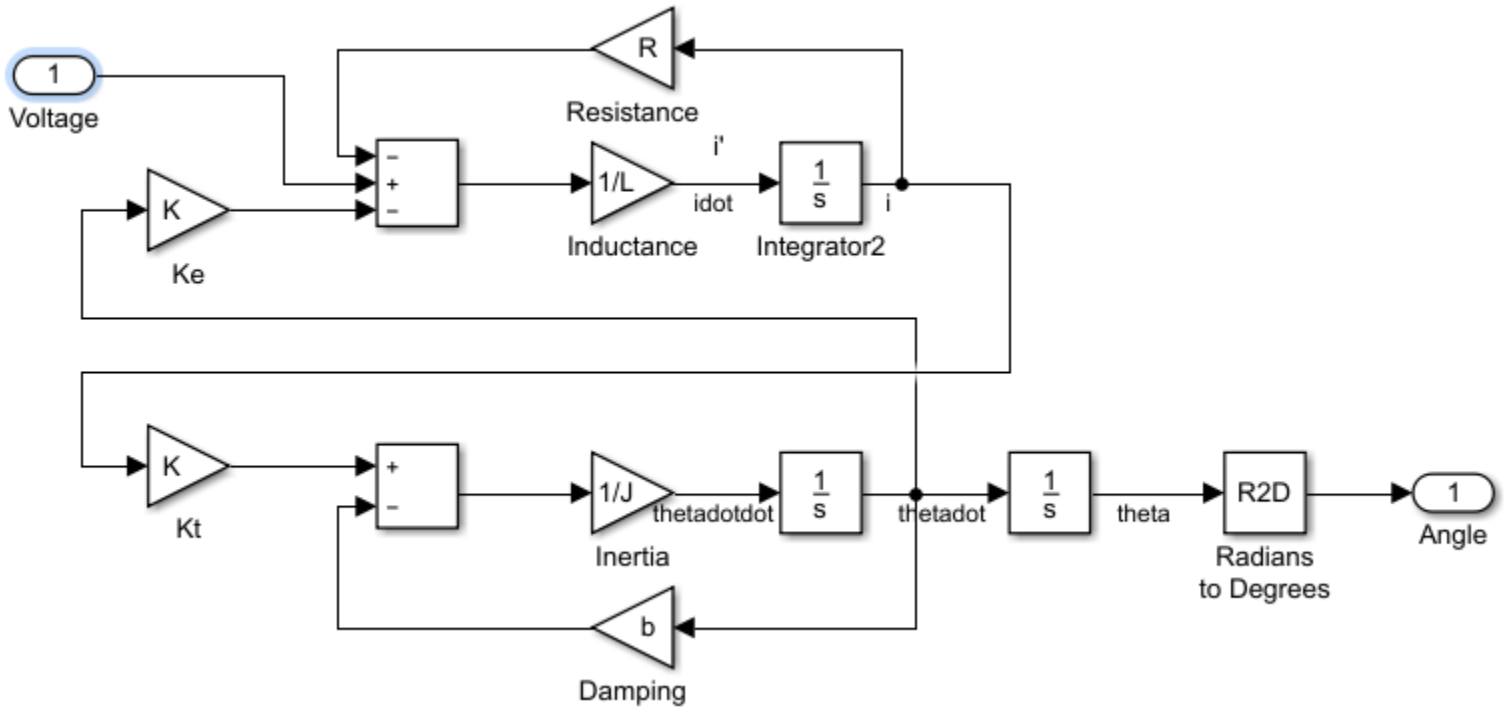
$$\frac{di}{dt} = \frac{1}{L}(V - K\omega - iR)$$

$$i = \int \frac{1}{L}(V - K\omega - iR)dt$$

$$J\frac{d\omega}{dt} = Ki - \mathrm{b}\omega - T_{Load}$$

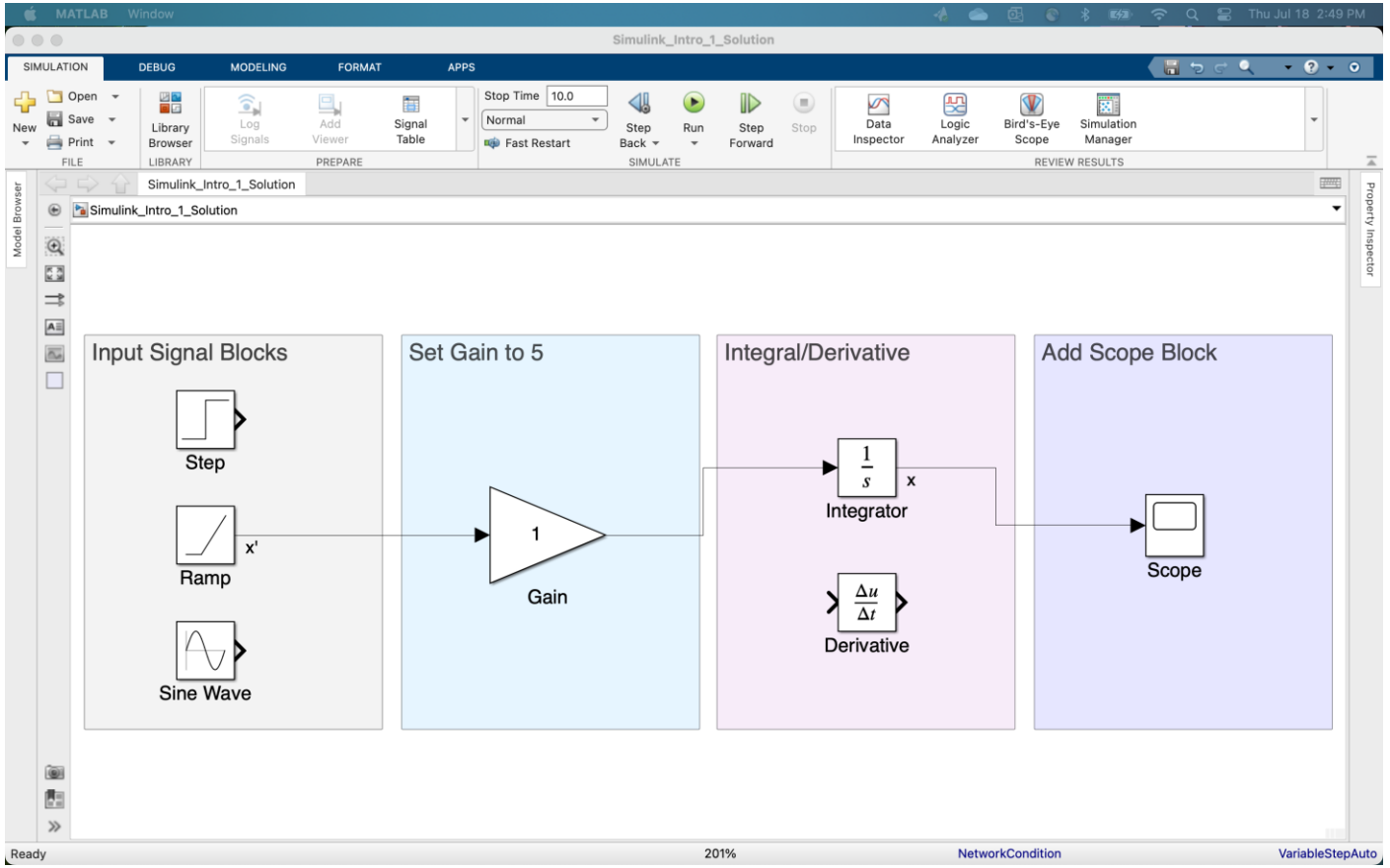
$$\frac{d\omega}{dt} = \frac{1}{J}(Ki - \mathrm{b}\omega - T_{Load})$$

$$\omega = \int \frac{1}{J}(Ki - \mathrm{b}\omega - T_{Load})dt$$



Exercise 1: Getting our feet wet with Simulink (10 mins)

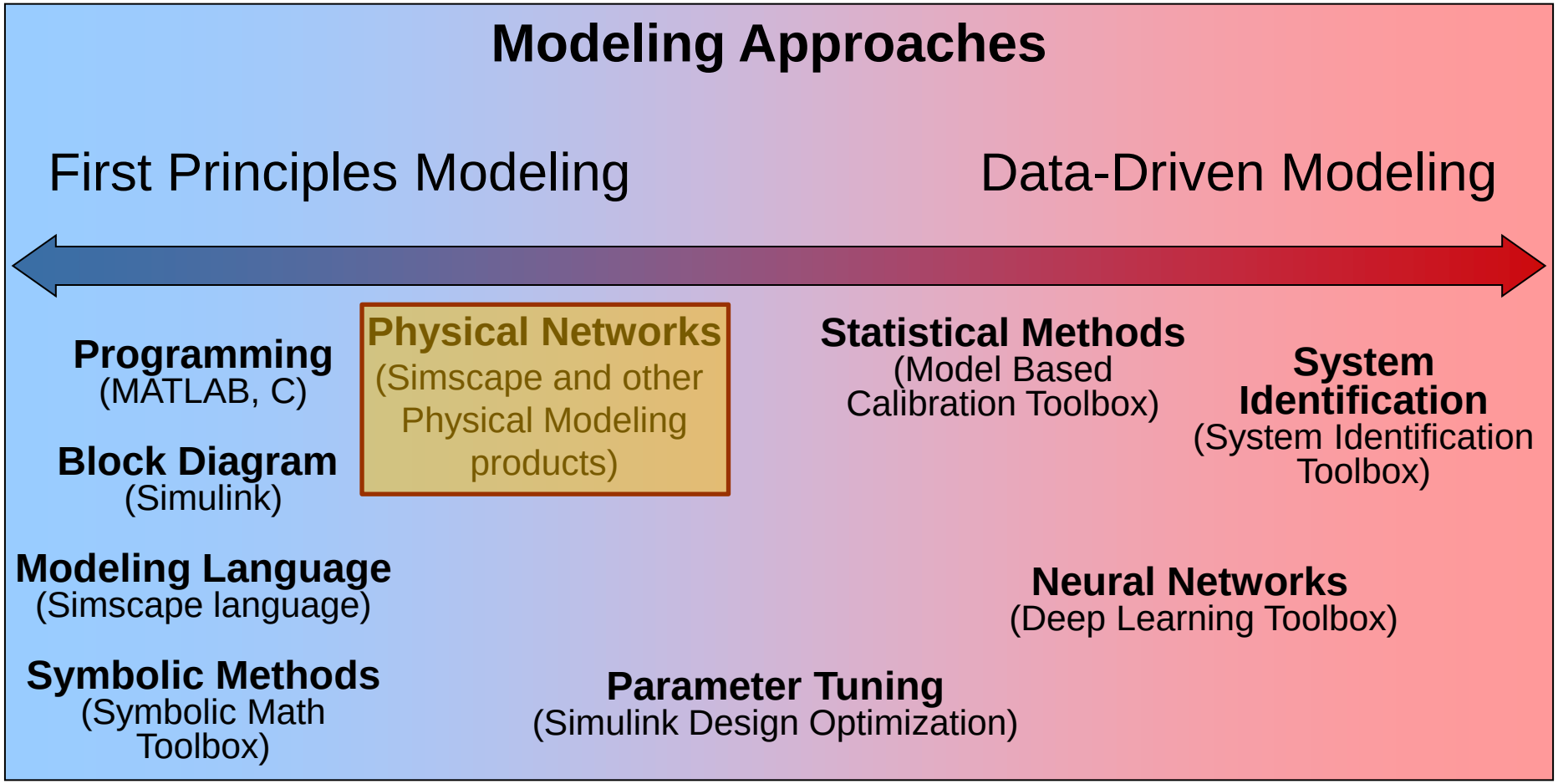
- Open: ***Simulink_Intro.slx***
- Connect any one of the ‘Input Signal Blocks’ to the Gain.
- Decide if you want to integrate or differentiate and connect to the corresponding block
- Add a scope block and connect. You can also connect the input block to see both signals



Agenda

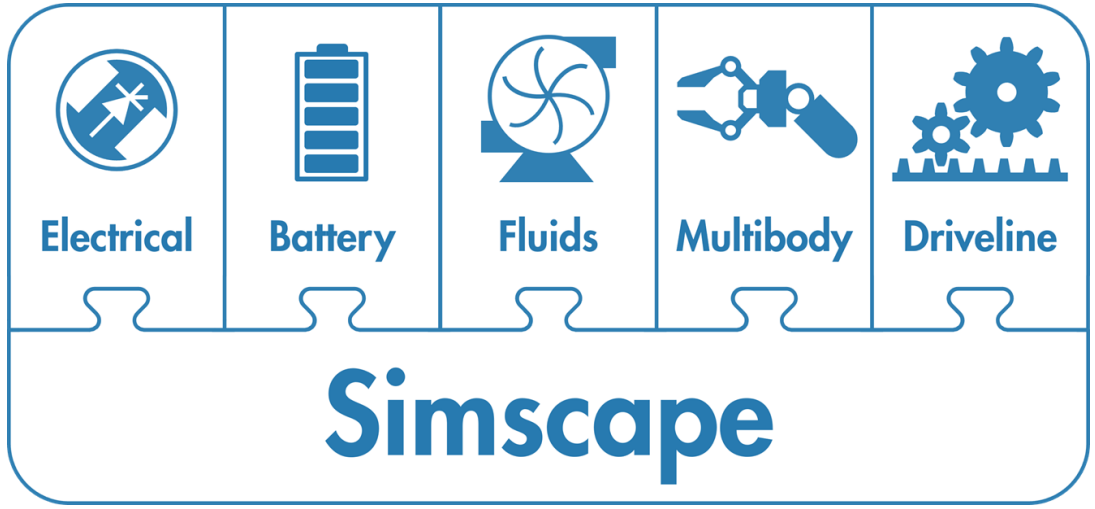
- Simulink Overview
- Introduction to Model-Based Design
- Various ways of modeling a DC Motor
  - Mathematical Modeling with Simulink
  - **Physical Modeling with Simscape**
- Control System design
- Automatic Code Generation
- Resources and Support

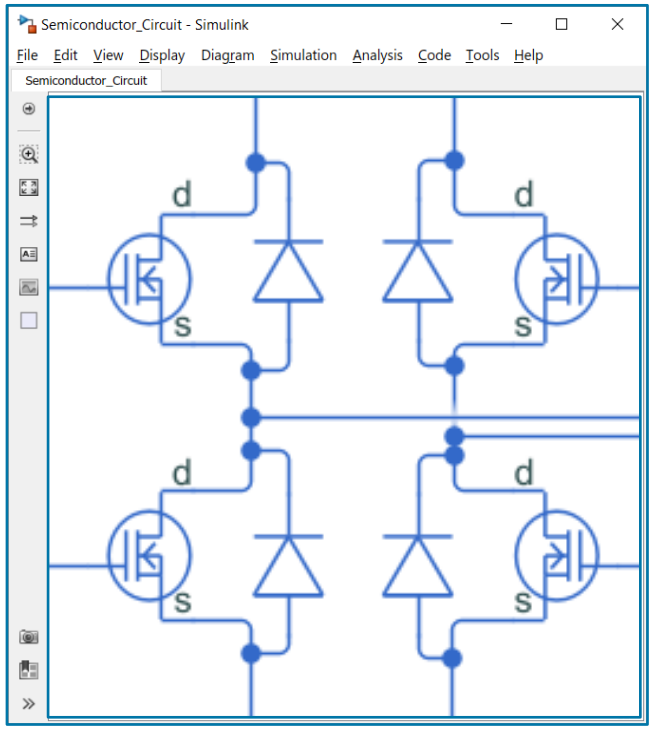
Different Approaches for Modeling Dynamic Systems

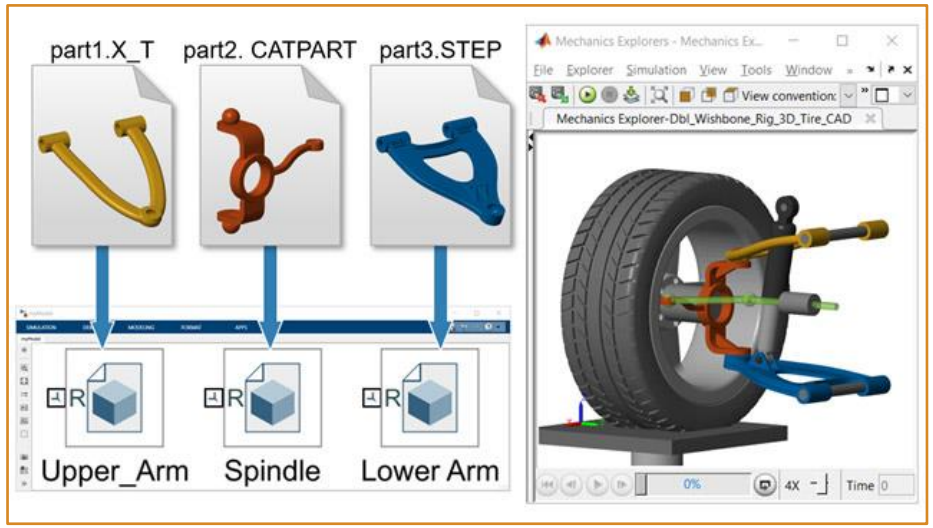


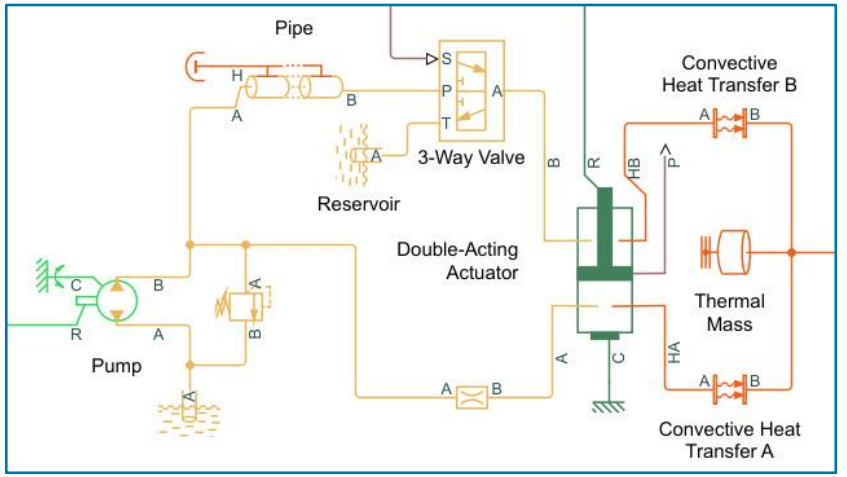
Simscape
Overview

- Simscape enables you to rapidly create models of physical systems within the Simulink environment

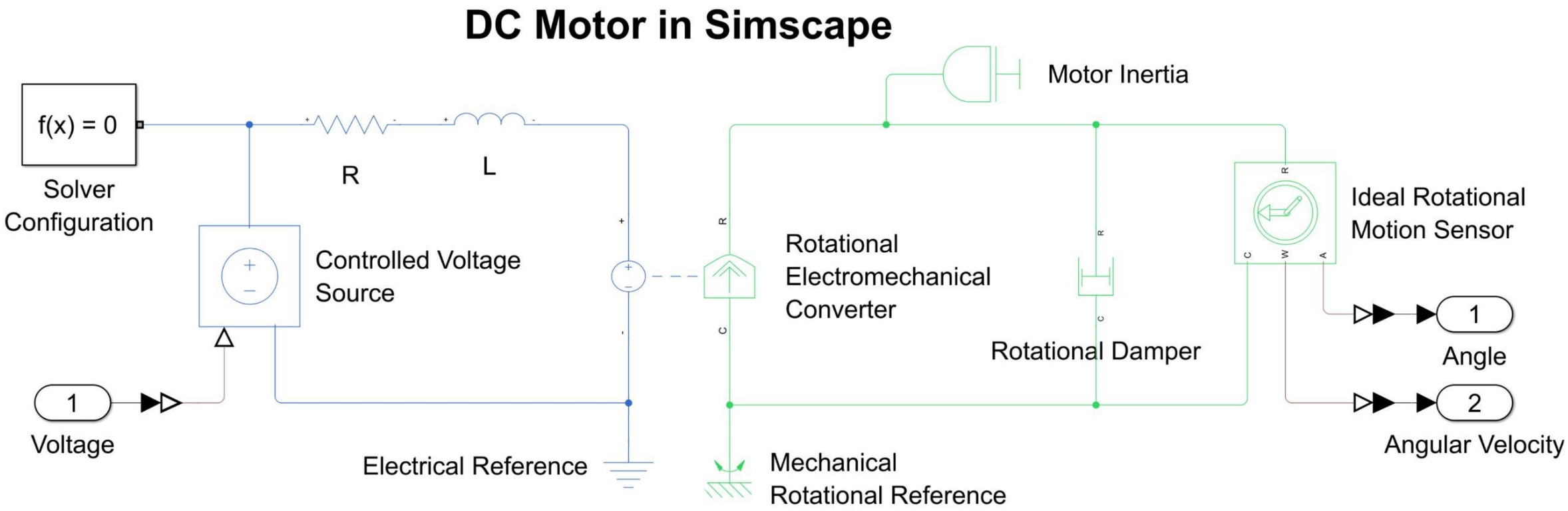






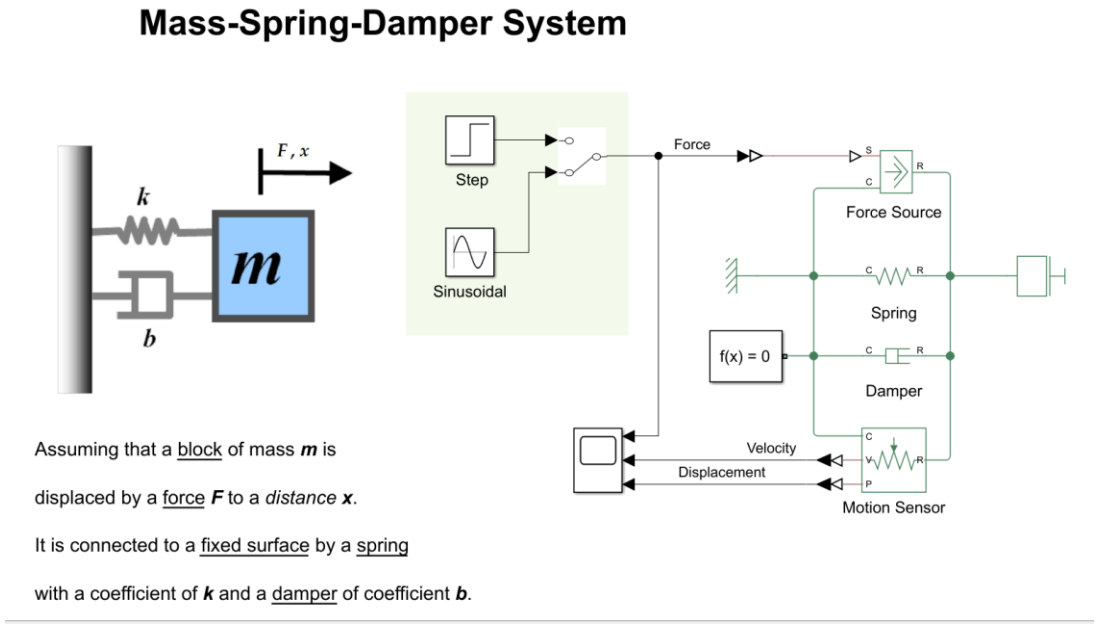


Model Physical Network of DC Motor using Simscape



Exercise 2: Mass-Spring Damper

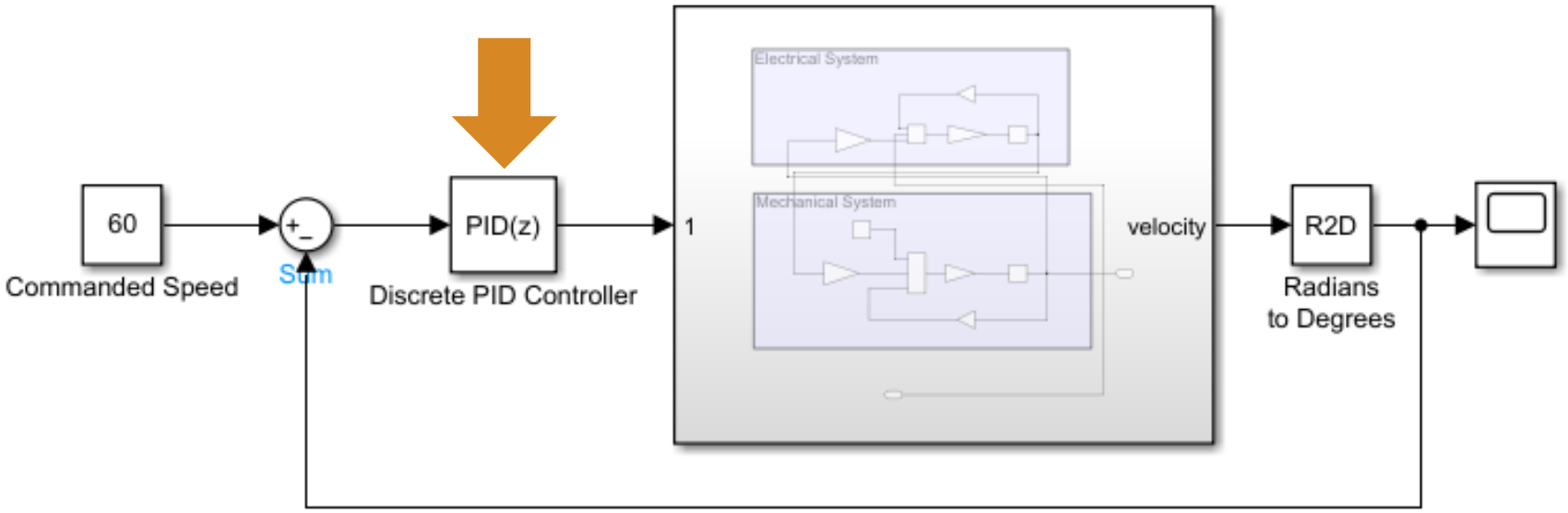
- Open: ***Mass_Spring_Damper.slx***
- Add Translational Spring and Mass blocks
- Edit parameters and view results



Agenda

- Simulink Overview
- Introduction to Model-Based Design
- Various ways of modeling a DC Motor
  - Mathematical Modeling with Simulink
  - Physical Modeling with Simscape
- **Control System design**
- Automatic Code Generation
- Resources and Support

Controlling Your System

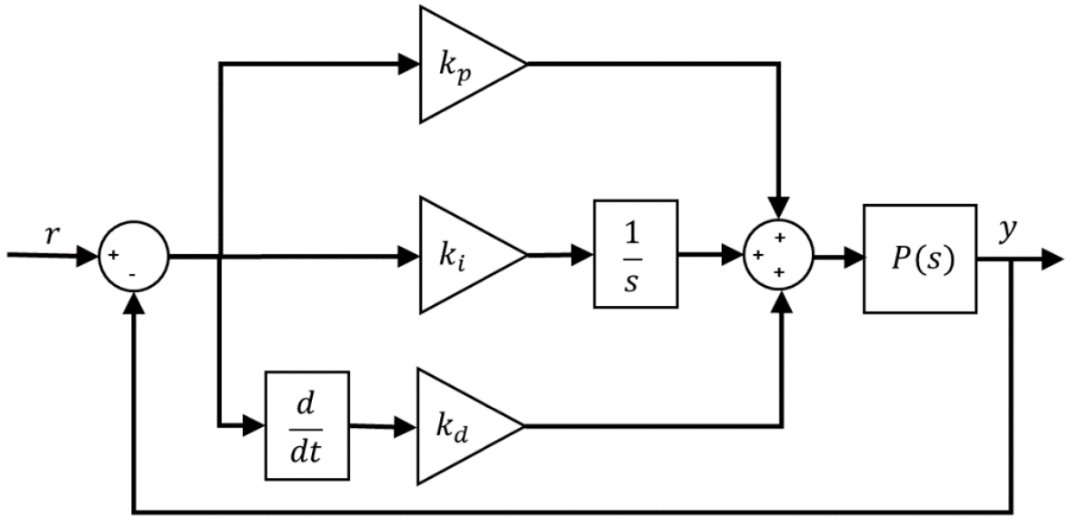


- Modeling the physical system is part of the job.
- What about the controls?

PID Controller – Proportional, Integral, Differential Controller

$$u(t) = k_p\, e(t) \;+\; k_i \int_0^t e(\tau)d\tau \;+\; k_d \frac{d\, e(t)}{dt}$$

$$U(s) = \left(k_p + \frac{k_i}{s} + k_d s\right) E(s)$$

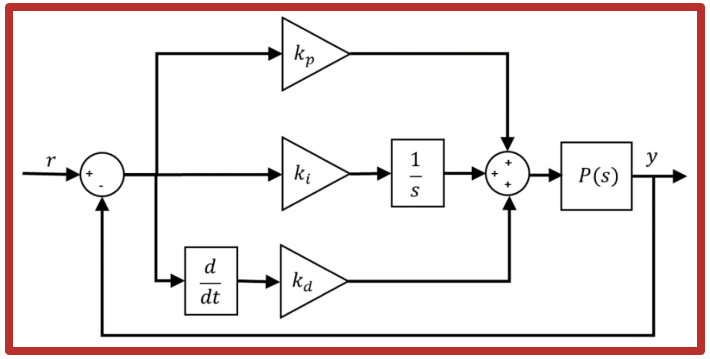


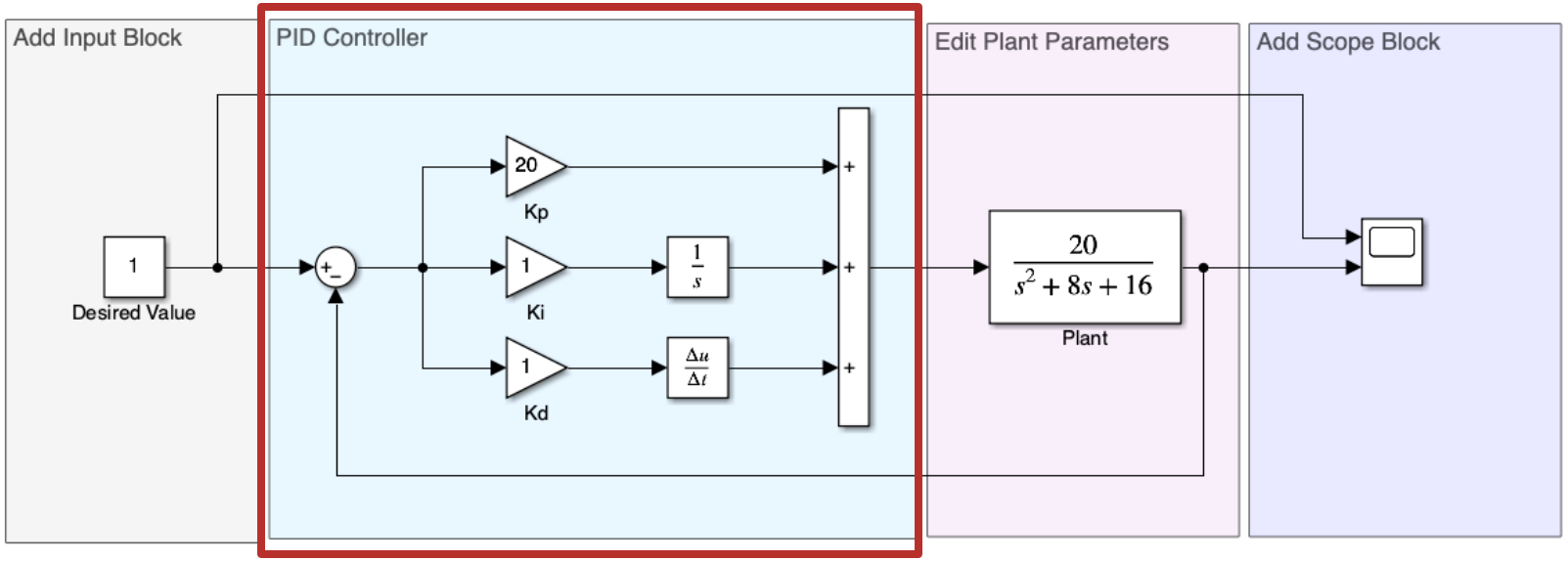
A PID controller continuously adjusts a system's output to minimize the error between a desired setpoint and a measured variable using proportional, integral, and derivative actions.

[What is a PID Controller?]

Exercise 3: Build a PID controller in Simulink

- Open: ***PID_Control.slx***
- Add a desired value block as input
- Decide the P, I, D values
- Edit the plant parameters by adding a “Transfer Function” block
- Add a scope block





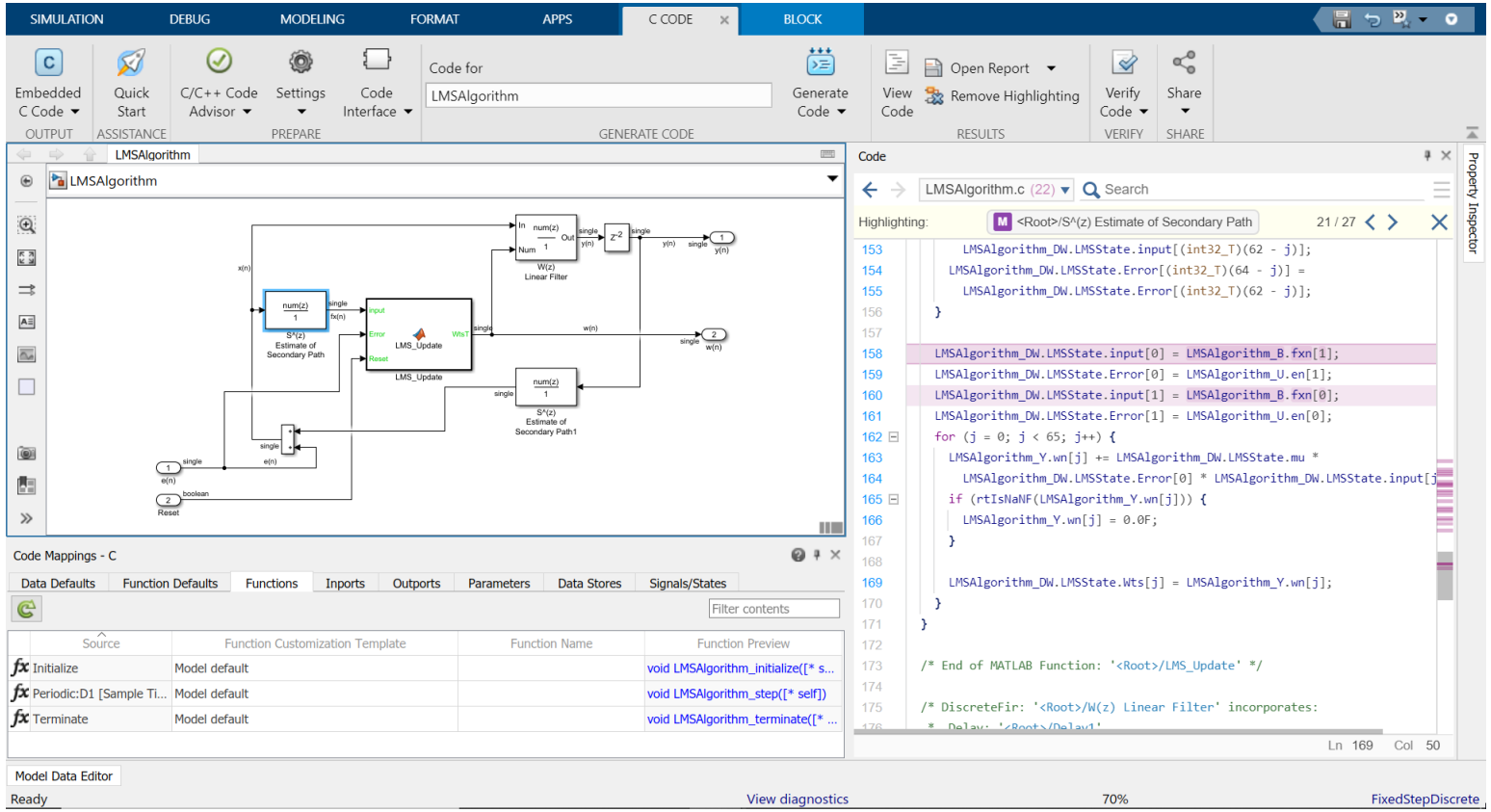
Agenda

- Simulink Overview
- Introduction to Model-Based Design
- Various ways of modeling a DC Motor
  - Mathematical Modeling with Simulink
  - Physical Modeling with Simscape
- Control System design
- **Automatic Code Generation**
- Resources and Support

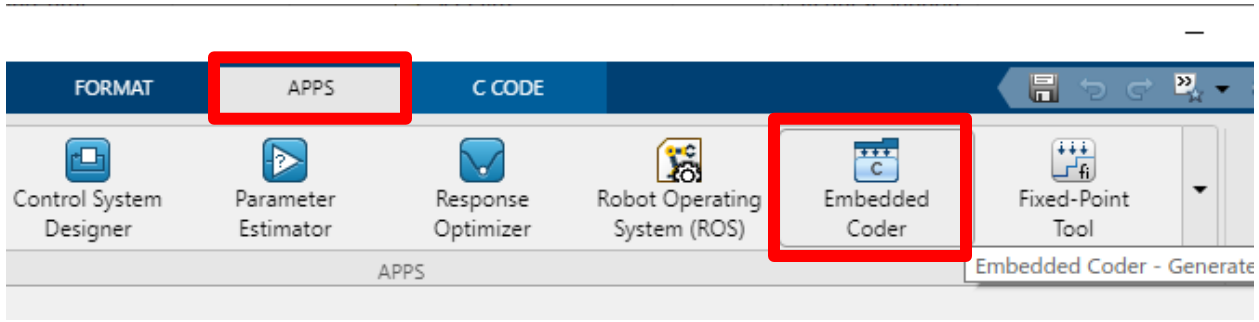
How do we go from model to code?

Embedded Coder – Generate C/C++ code optimized for embedded systems

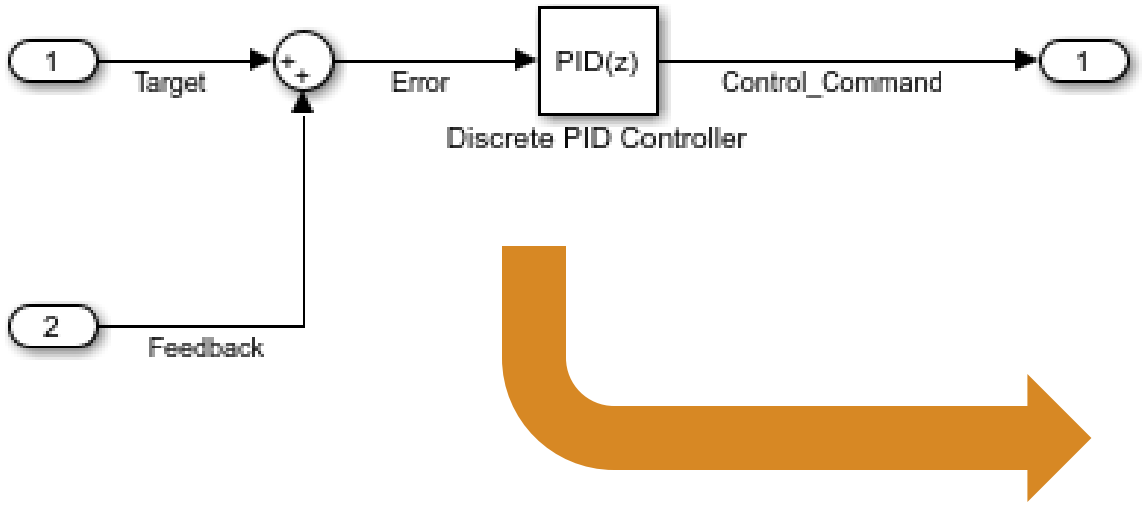
- **Improve** efficiency in design iterations
- **Trace** among requirements, model, code, and tests
- **Streamline** code integration with customizable code interfaces and appearances
- **Leverage** hardware optimized code including DSP library for ARM processors



Embedded Coder



Access code generation tools from the Embedded Coder App



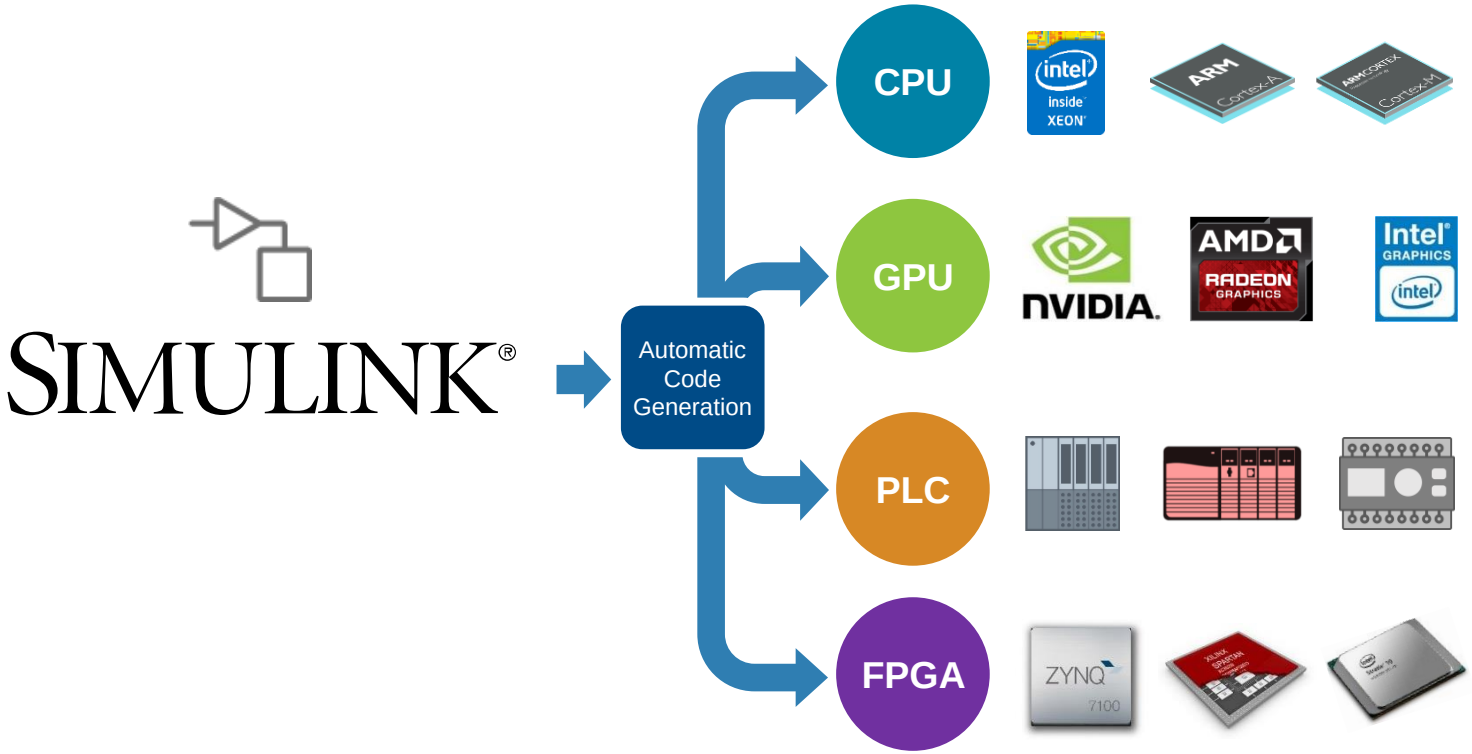

```
32
33  /* Model step function */
34  void codegen_example_step(void)
35  {
36    real_T rtb_Error;
37    real_T rtb_Tsamp;
38
39    /* Sum: '<Root>/Sum' incorporates:
40     *  Inport: '<Root>/Feedback'
41     *  Inport: '<Root>/Target'
42     */
43    rtb_Error = rtU.Target + rtU.Feedback;
44
45    /* SampleTimeMath: '<S30>/Tsamp' incorporates:
46     *  Gain: '<S27>/Derivative Gain'
47     *
48     * About '<S30>/Tsamp':
49     *  y = u * K where K = 1 / ( w * Ts )
50     */
51    rtb_Tsamp = rtP.DiscretePIDController_D * rtb_Error * rtP.Tsamp_WtEt;
52
53    /* Outport: '<Root>/Control_Command' incorporates:
```

How do we go from model to code?

Automate **code generation** for prototyping and production

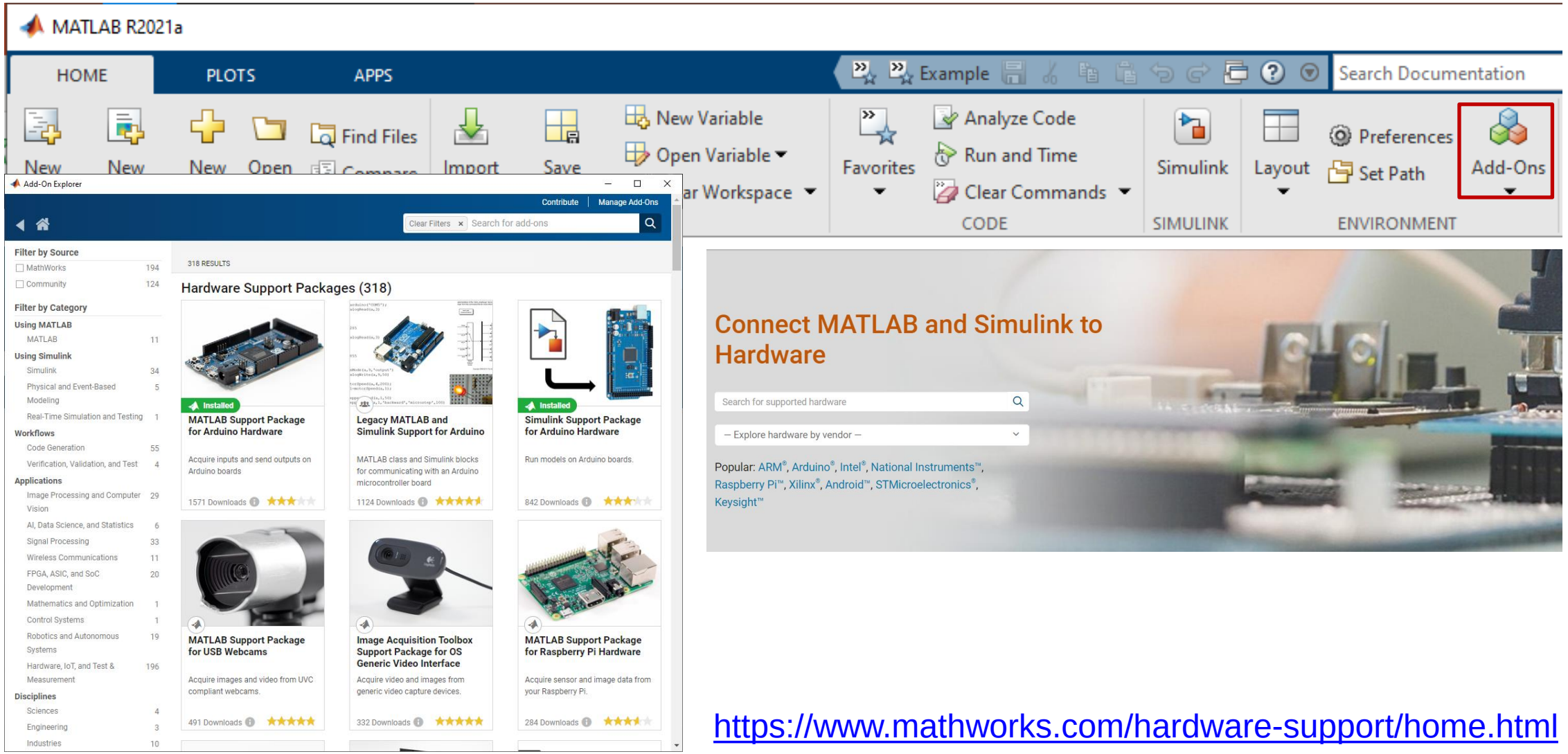
Automation

Coding | Verification



Reliable and **high performance**, with **flexible** choice of targets

Hardware Support Packages



https://www.mathworks.com/hardware-support/home.html

Key Takeaways

- Design and simulate multi-domain systems with **Simulink and Simscape**
- Tune system parameters and outputs with **Control Design**
- Automatically generate code for hardware with **Coders*** and **Support Packages**

** MATLAB Coder, Simulink Coder, Embedded Coder, HDL Coder, GPU Coder*

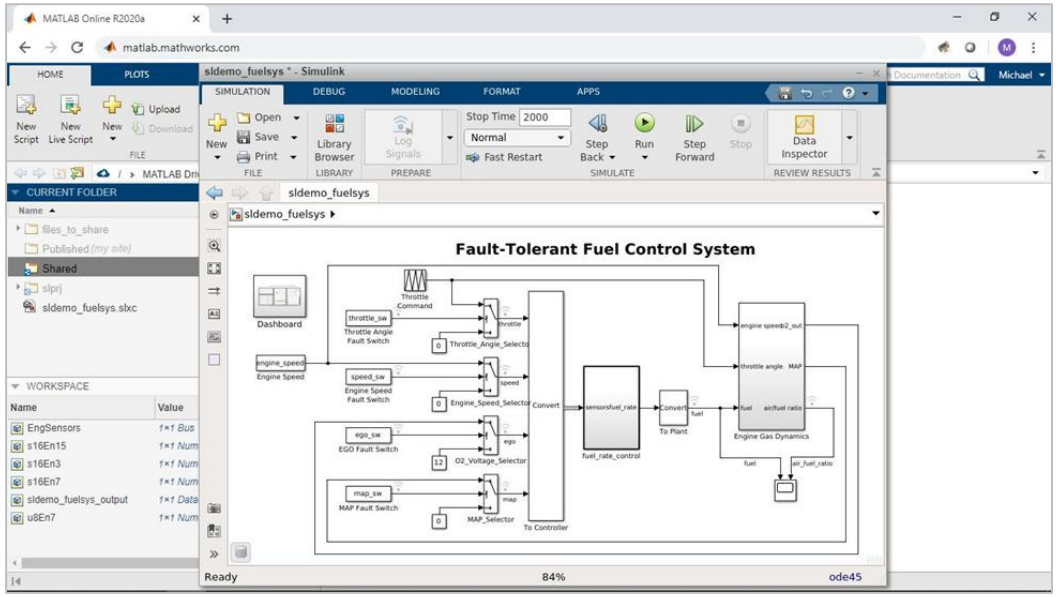
Agenda

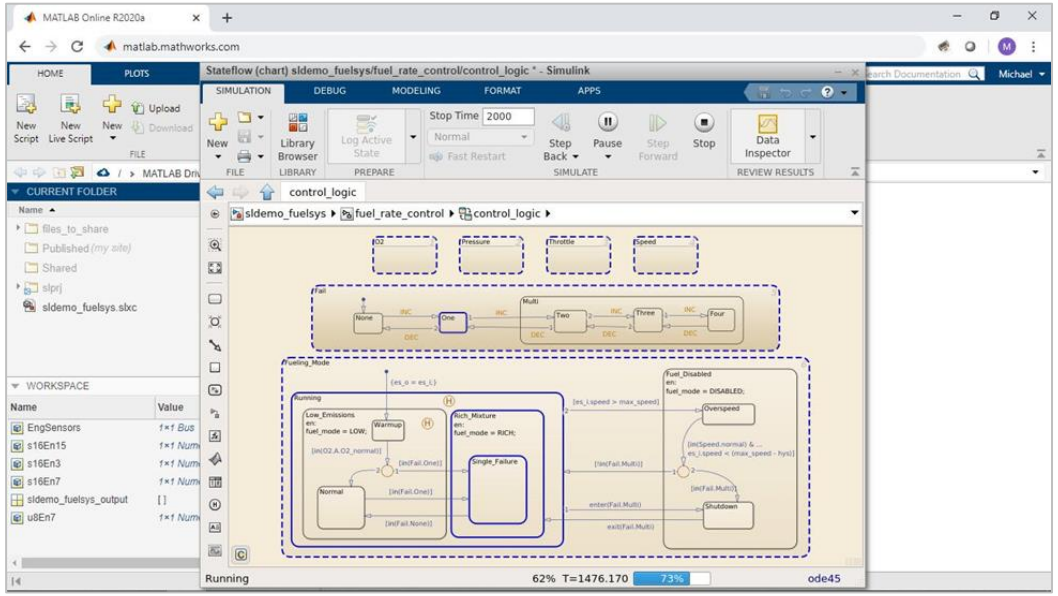
- Simulink Overview
- Introduction to Model-Based Design
- Various ways of modeling a DC Motor
  - Mathematical Modeling with Simulink
  - Physical Modeling with Simscape
- Control System design
- Automatic Code Generation
- **Resources and Support**

Simulink Online

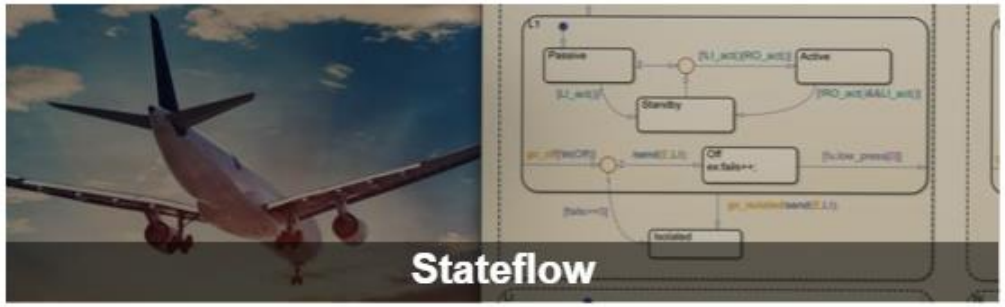
Use Simulink through your web browser

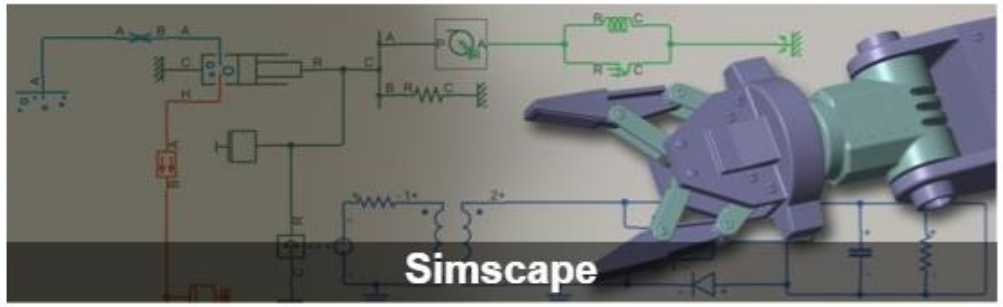
- No downloads or installations required
- Available to anyone with a Simulink license who is eligible to use MATLAB Online
  - Most EDU licenses and Individual
- Ideal for teaching, learning, and convenient, lightweight access

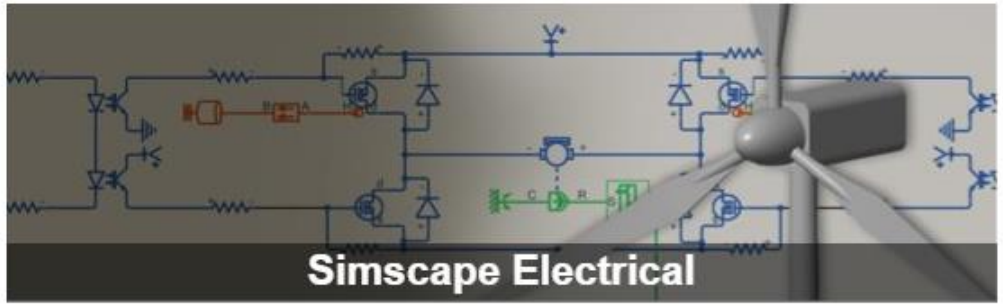


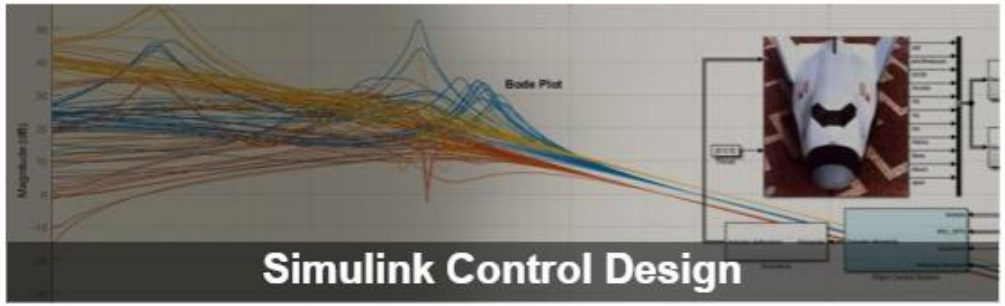


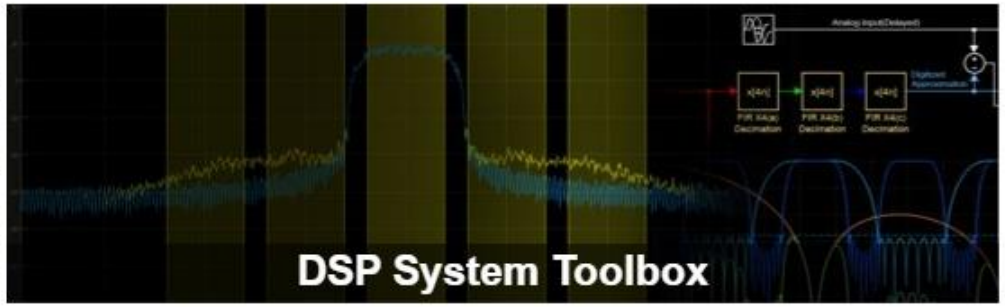
Simulink Online: Supported Products







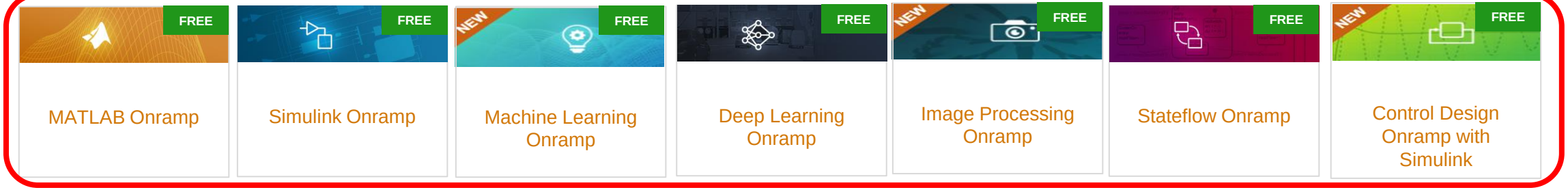




https://www.mathworks.com/products/matlab-online/limitations.html

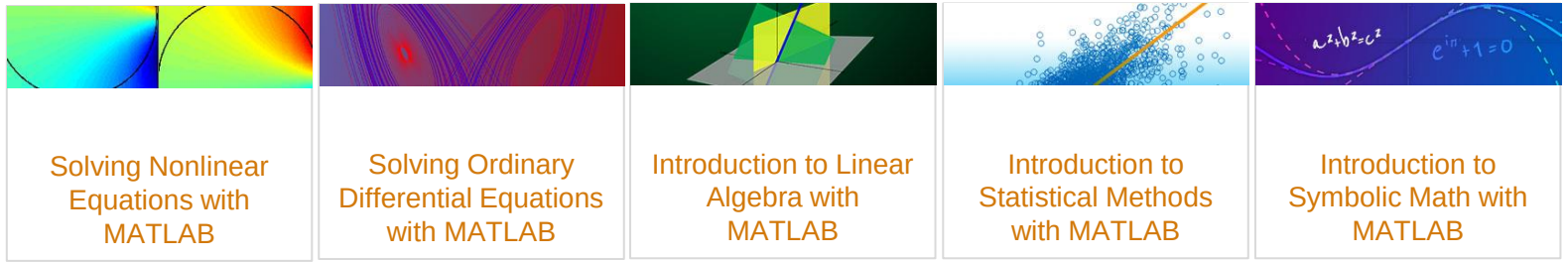
Self-Paced Courses

FREE "getting started" content – available for everyone



Computational Mathematics

*Available only to users at universities that offer campus-wide training access.



Other Resources:

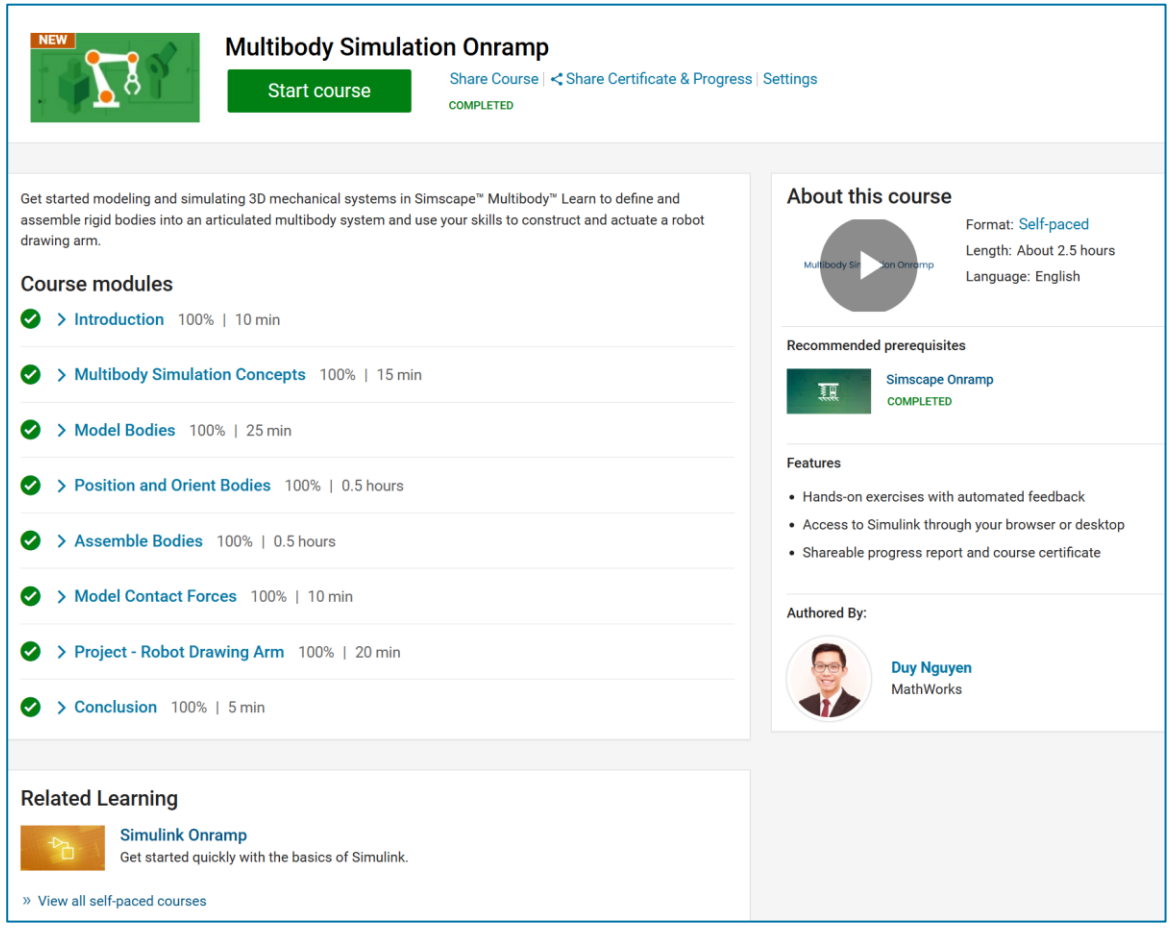
- Introduction to Model-Based Design with Simulink
- Physical Modeling with Simscape
- Simulink Control Design
- Robotics and Autonomous Systems
- (NEW) Multibody Simulation
- (NEW) Battery Systems

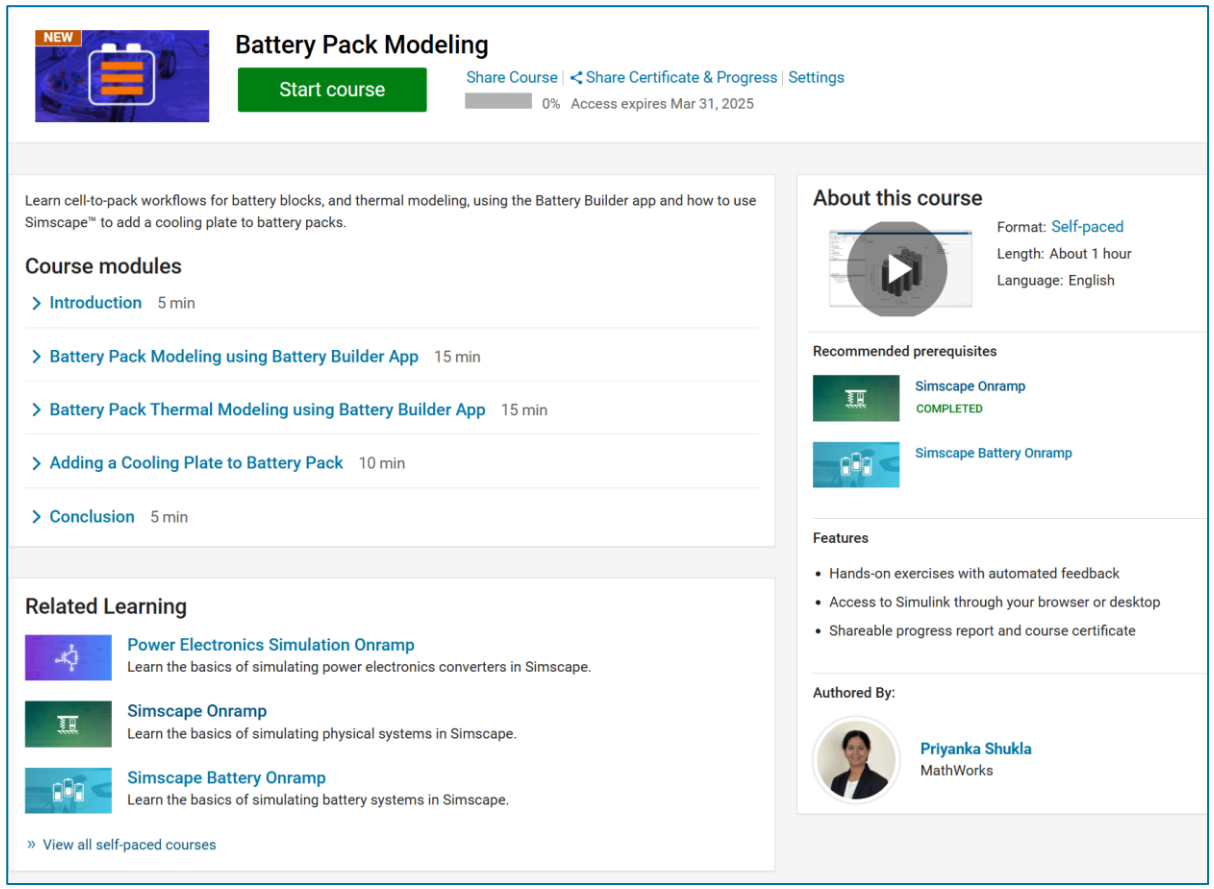
Core MATLAB

Data Science

Self-Paced Courses

FREE “getting started” content – available for everyone



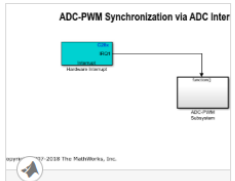


https://matlabacademy.mathworks.com/

Try an Example

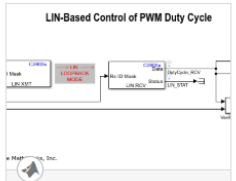
Modeling — Examples

R2019a



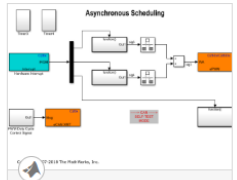
ADC-PWM Synchronization Using ADC Interrupt

Use the ADC block to sample an analog voltage and use the PWM block to generate a pulse waveform. This example also shows how to use



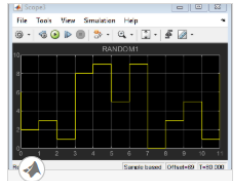
LIN-Based Control of PWM Duty Cycle

Use the C28035 LIN Receive, LIN Transmit, and PWM blocks to generate a pulse waveform.



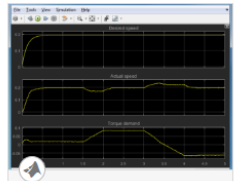
Asynchronous Scheduling

Use the Texas Instruments™ c28x peripherals and Hardware Interrupt blocks to control the real-time execution of Simulink® function-call



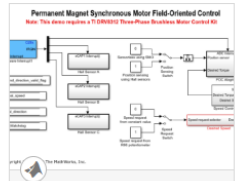
Using CAN Calibration Protocol in External Mode

Use the CAN Calibration Protocol (CCP) block to monitor model signals and tune parameter values in the application code running on



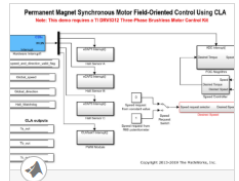
Simulation of FOC Using PMSM Model

Simulate field-oriented control (FOC) using a Permanent Magnet Synchronous Machine (PMSM) model. The model is created using



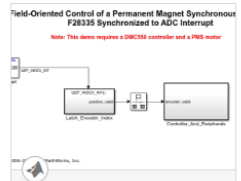
Permanent Magnet Synchronous Motor Field-Oriented Control

Control the speed of a three-phase Permanent Magnet Synchronous Motor in a closed-loop fashion via Field-Oriented Control (FOC)using



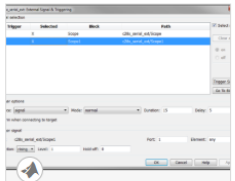
Using the Control Law Accelerator (CLA)

Use the Control Law Accelerator (CLA) available on some of the TI Piccolo processors.



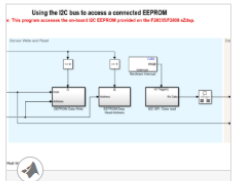
Schedule a Multi-Rate Controller for a Permanent Magnet Synchronous...

Create a real-time executable for a Texas Instruments F28335 embedded target. You will build upon the algorithm specified in the



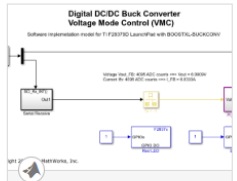
Parameter Tuning and Signal Logging with Serial External Mode

Perform parameter tuning and data logging with a Simulink® model running in External mode on the Texas Instruments™ C2000 targets.



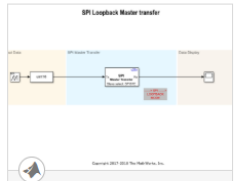
Using the I2C Bus to Access Sensors

Use the I2C blocks to communicate with I2C based devices.



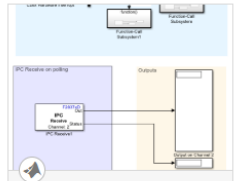
Modeling a Voltage Controller for the DC/DC Buck Converter

Model a controller for the DC/DC buck converter using the Embedded Coder Support Package for Texas Instruments® C2000 Processors.



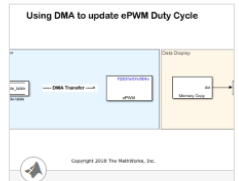
Using SPI to Read and Write Data to SPI EEPROM

Configure and use SPI blocks to read and write data.



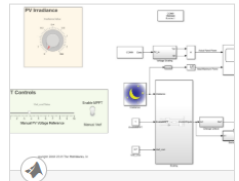
Inter-Processor Communication Using IPC Blocks

Use the IPC blocks to communicate between the two CPUs of Texas Instruments™ Delfino F2837xD using Simulink® models.



Modify Duty Cycle of ePWM Using DMA

Configure the direct memory access (DMA) parameters to modify the ePWM duty cycle. Using DMA, the sine wave data is copied from a



Photovoltaic Inverter with MPPT Using Solar Explorer Kit

Implement a photovoltaic (PV) inverter system using the Embedded Coder® Support Package for Texas Instruments™ C2000™ Processors.

Real-Time Code Execution Profiling

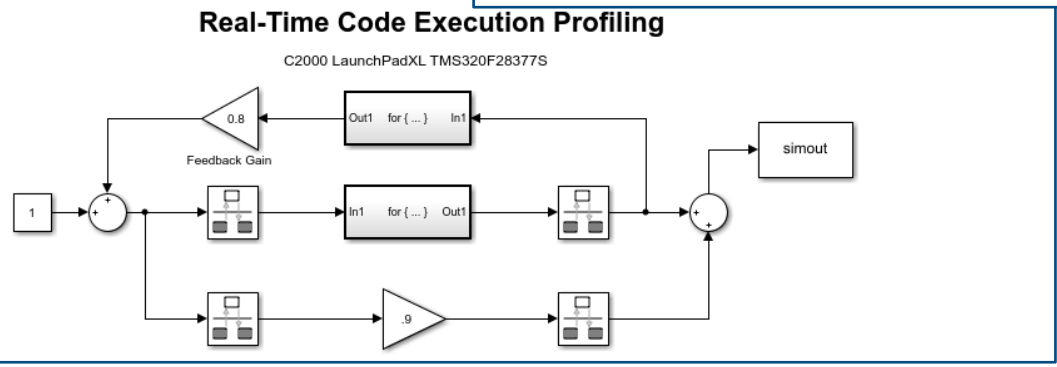
This example shows you how to use Embedded Coder™ Support P
C2000 Processors and Embedded Coder Support Package for Texa
Concerto Processors for real-time execution profiling of generated c

Available versions of this model:

- C2000 LaunchPadXL TMS320F28377S (LAUNCHXL-F28377S)

Model

The following figure shows the example model:



https://www.mathworks.com/help/simulink/examples.html

Examples of MathWorks Supported Hardware

Arduino

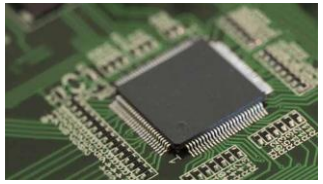
Lego EV3

Raspberry Pi

Android/iOS Devices

Kinect for Windows

BeagleBone Black



Texas Instruments

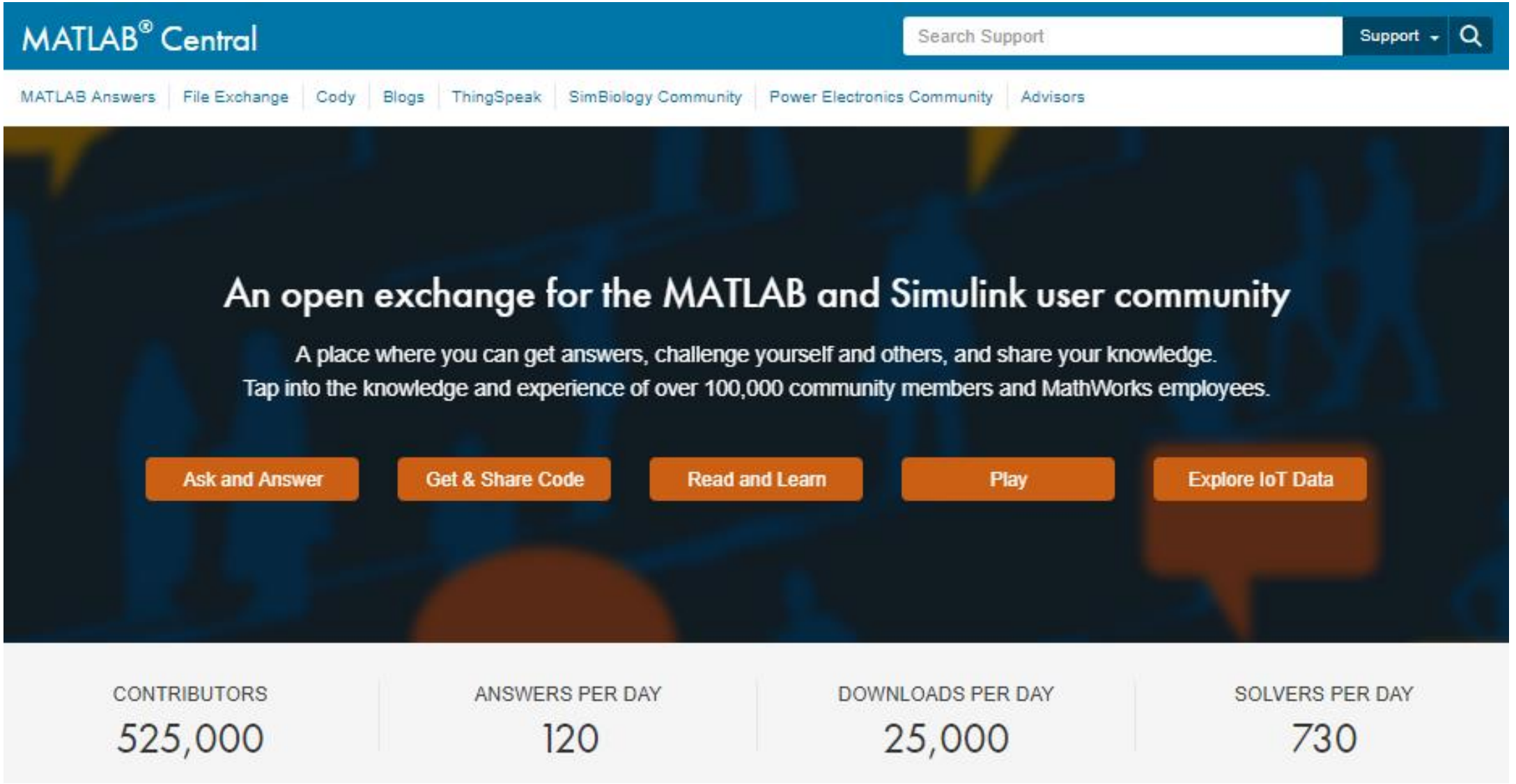
STM Electronics

Freescale

Zynq SDR

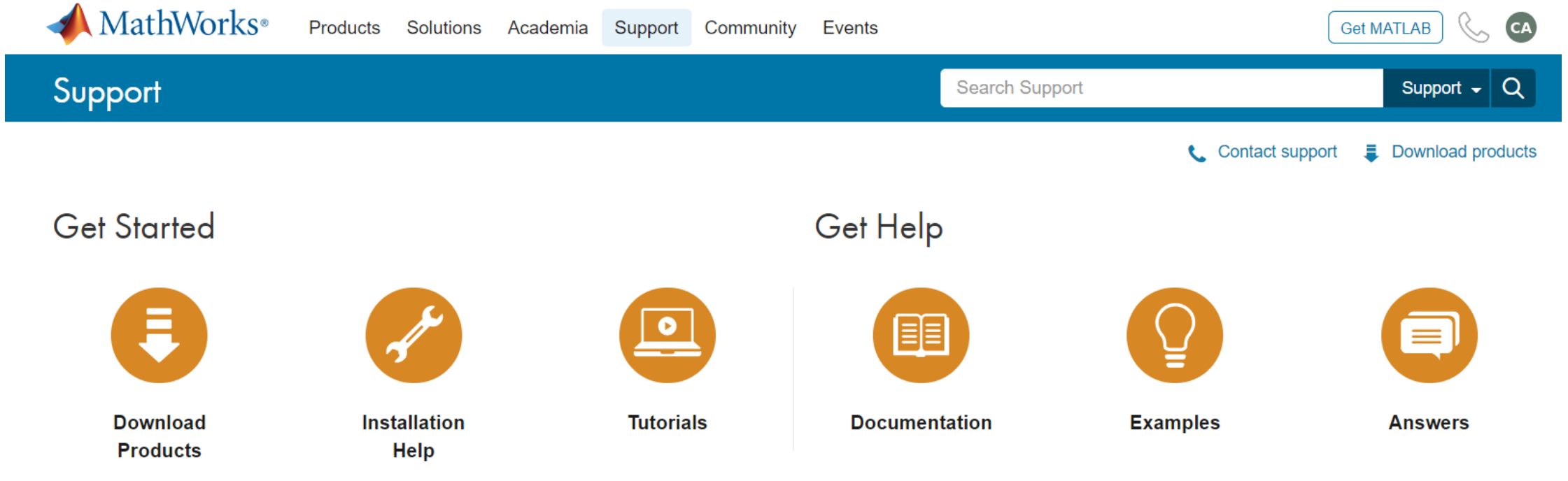
https://www.mathworks.com/hardware-support/home.html

MATLAB Central



https://www.mathworks.com/matlabcentral/?s_tid=gn_mlc

Technical Support



MathWorks Education Application Engineers *nsardesa@mathworks.com*

consult with faculty and researchers to support them with their STEM initiatives,

including integrating computational or systems thinking into their curriculum and research