



SECURE CONNECTIONS
FOR A SMARTER WORLD

Rapid Prototyping Embedded Designs using NXP Model-Based Design Toolbox: A Battery Management System Application

Irina Costachescu



Marius-Lucian Andrei

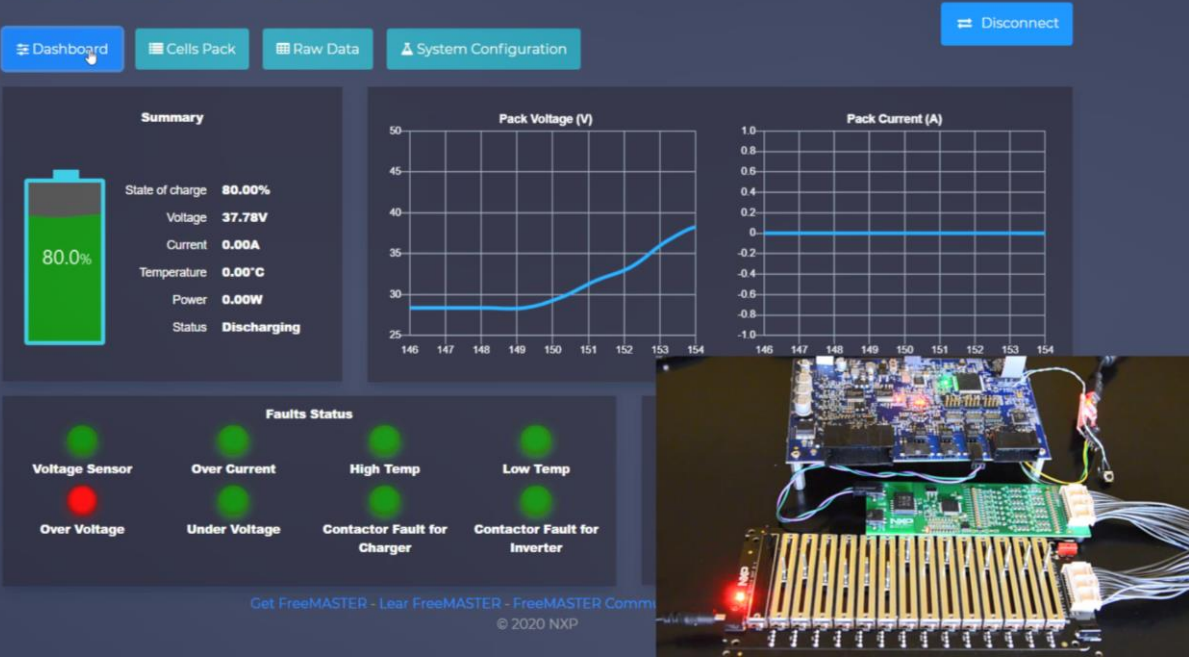


MATLAB EXPO

AGENDA

- 1. NXP Ecosystem
- 2. NXP's Model-Based Design Toolbox Introduction
- 3. Model-Based Design Toolbox for **Battery Management Systems**

Battery Management System



NXP SEMICONDUCTORS WORLDWIDE

Together with our valued customers, we're not just advancing technology, we're advancing society.



AUTOMOTIVE

Enabling carmakers to develop smarter solutions for complex autonomy, connectivity, and electrification challenges

Accelerating the shift to greater mobility



INDUSTRIAL

Reducing wasted time, money, and effort by helping business run more efficiently.

Enabling more efficient data processing



MOBILE

Giving wearable and mobile devices easier access to the services that make modern life more convenient without compromising security and safety.

Transforming how people and devices connect



SMART HOME

Solutions that listen, learn, and adapt into the places we call home for more comfort, affordability, safety, and convenience.

Powering the intelligence behind the technologies



SMART CITY

Simplifying how people access and interact with local services to achieve new standards of sustainability, efficiency, mobility, and economic growth.

Anticipating the demands of tomorrow



COMMUNICATION INFRASTRUCTURE

Powering insights and inspiring performance with hardware solutions for handling 5G connectivity across the emerging communications spectrum.

Delivering real-time responsiveness at the speed of 5G

60 years of combined experience and expertise
Operations in more than **30 countries** worldwide

Approximately **31,000 employees**

Headquarters in The Netherlands – **Eindhoven**



NXP ECOSYSTEM

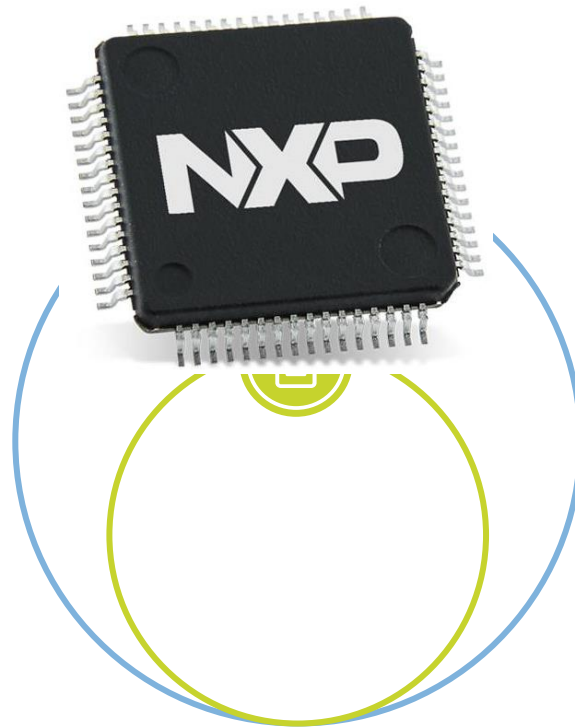


DRIVERS, MIDDLEWARE, LIBRARIES

- Simplify hardware access by using hardware optimized software



BUILD, DEBUG, CONFIGURATION TOOLS



BUILD, DEBUG, CONFIGURATION TOOLS

The screenshot displays the NXP S32V2 configuration tool interface, showing a clock diagram, a clock table, and code generation results.

Diagram: The main diagram illustrates the clock distribution architecture. It shows the derivation of various system clocks from primary sources like FXOSC (48 MHz) and XTAL (40 MHz). Key components include ARMPLL (ARM PLL) and VIDEOLL (Video PLL) blocks, which are divided to produce clocks such as SYSCLK0 (265.97 MHz), SYSCLK1 (800 MHz), and SYSCLK2 (399.22 MHz). These system clocks are further divided to generate peripheral clocks like SYS3_CLK, SYS6_CLK, CORE_CLK, GPU_CLK, and MIPI_LL_CLK. Video-related clocks like VIDEOLL_CLKOUT (48 MHz) and APEX_SYS_CLK (24 MHz) are also shown.

Clocks Table: A table listing clock sources and their configurations. The 'FXOSC source' is highlighted with a yellow background. It specifies the type, enable/disable status, constraints (output frequency range: 8 MHz - 40 MHz), and other run modes (RUN0-RUN3) with their respective frequency ranges. The current value is set to 40 MHz.

Name	C...	Val
Internal		
Fast IRC		48 MHz
IRCOSC Power		Power
External		
FXOSC source		40 MHz
FXOSC Op...ion Mode		Osc
ftm_0_ext...rence clock		Inac
ftm_1_ext...rence clock		Inac

Code Preview: The code preview shows the generated C code for the clock configuration, including the definition of clock names and the configuration of the MC (Microcontroller) run peripheral configuration.

```

clock_config.c clock_config.h
524         .clockName = DEC ^
525         .mc_me_RunPeriphConfig = MC_
526     },
527     {
528         .clockName = CRC
529         .mc_me_RunPeriphConfig = MC_
530     },
531     {

```

Code successfully generated.

Problems: The problems panel is currently empty, indicating no issues were detected during the configuration process.

NXP ECOSYSTEM



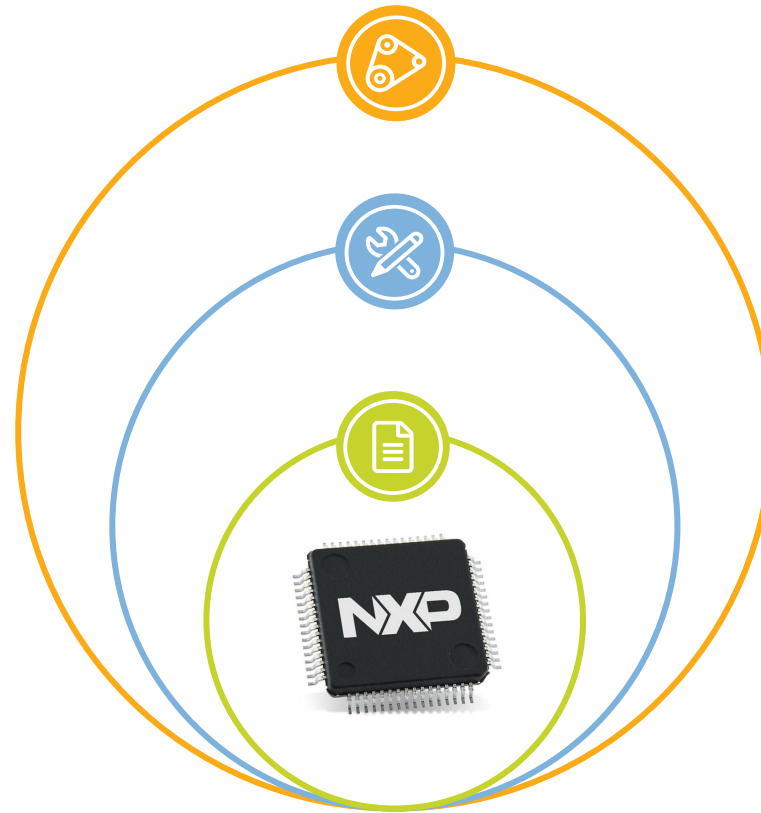
DRIVERS, MIDDLEWARE, LIBRARIES

- Simplify hardware access by using hardware optimized software



BUILD, DEBUG, CONFIGURATION TOOLS

- Application development inside an IDE
- Build Tools, Debug Tools and Configuration Tools integrated within the IDE
- Drivers, Middleware and Libraries configuration and initialization in a graphical environment



REAL TIME MONITOR, DEMO TOOLS



REAL TIME MONITOR, DEMO TOOLS

The screenshot displays the FreeMASTER interface for a PMSM control system. The main window shows a detailed block diagram of the control system, including a Slow Control Loop with PI controllers for speed and current, a Fast Control Loop with dq/αβ and SVM blocks, and observers for back-EMF and tracking. Below the diagram is an image of the NXP hardware board and a PMSM motor.

On the right, an oscilloscope window titled "Oscilloscope - Speed" shows three traces: Speed Required (blue), SpeedRampOut (purple), and Speed Actual (green). The speed starts at 1500 rpm, drops to 500 rpm, and then returns to 1500 rpm. A second trace shows current (i_q and i_q required) in Amperes, showing a corresponding drop and recovery.

At the bottom, a Variable Watch table provides real-time data for various system parameters:

Name	Value	Unit	Period
Control Mode	VECTOR CONTROL [4]	ENUM	1000
Vector Control Type	Speed [1]	ENUM	1000
Speed Required	1500	rpm	500
Speed Actual	1501.97	rpm	1000
U _q	0	Volt	100
Scalar Frequency Required	0	Hz	100
I _q Required	0	Amps	1000
I _q	0.265763	Amps	1000
U _d _req	-0.650648	Volts	1000
U _q _req	4.05735	Volts	1000
DCB Voltage	23.9744	Volt	200

NXP ECOSYSTEM



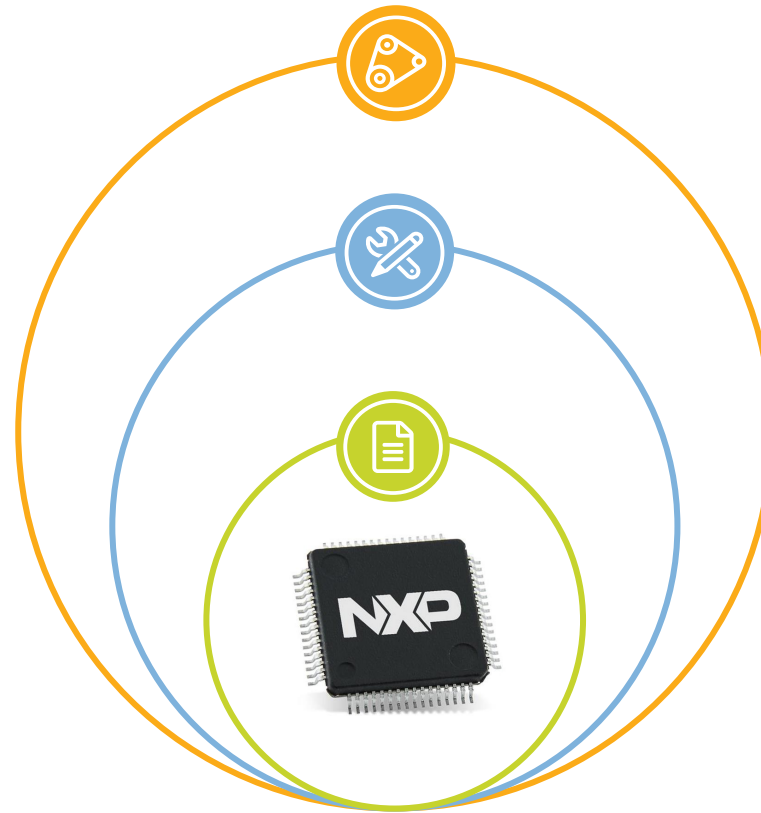
DRIVERS, MIDDLEWARE, LIBRARIES

- Simplify hardware access by using hardware optimized software



BUILD, DEBUG, CONFIGURATION TOOLS

- Application development inside an IDE
- Build Tools, Debug Tools and Configuration Tools integrated within the IDE
- Drivers, Middleware and Libraries configuration and initialization in a graphical environment



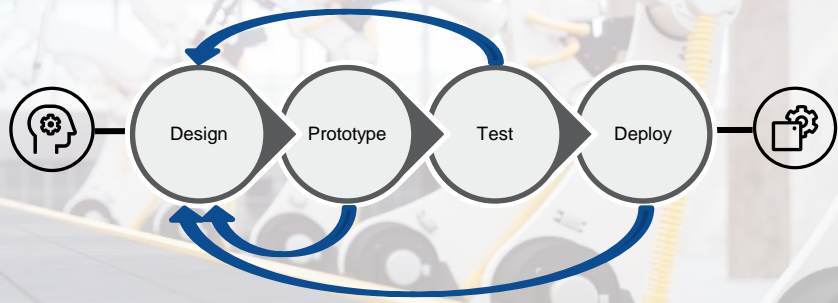
REAL TIME MONITOR, DEMO TOOLS



- Check the status of the running on target application in real time using **FreeMASTER**
- Write and read variables, registers, memory locations
- Monitor signals on the embedded target
- Fast demo design



FROM IDEA TO APPLICATION



DRIVERS, MIDDLEWARE, LIBRARIES

- Simplify hardware access by using hardware optimized software



BUILD, DEBUG, CONFIGURATION TOOLS

- Application development inside an IDE
- Build Tools, Debug Tools and Configuration Tools integrated within the IDE
- Drivers, Middleware and Libraries configuration and initialization in a graphical environment



REAL TIME MONITOR, DEMO TOOLS

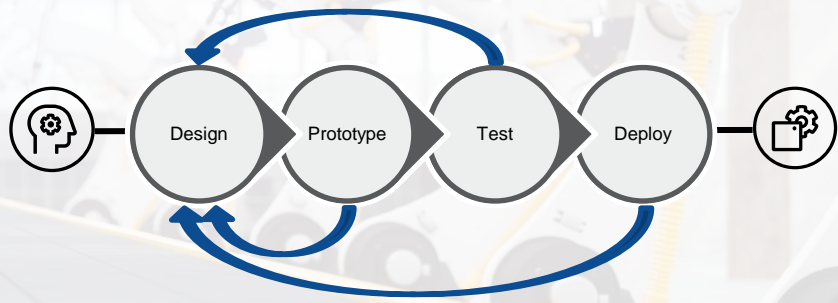
- Check the status of the running on target application in real time using **FreeMASTER**
- Write and read variables, registers, memory locations
- Monitor signals on the embedded target
- Fast demo design

MODEL-BASED DESIGN

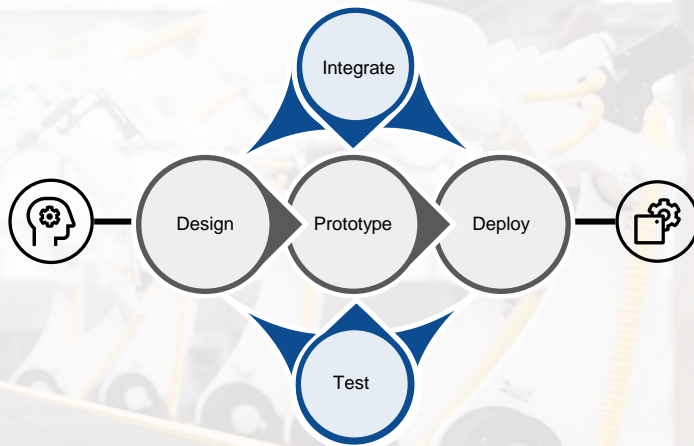




FROM IDEA TO APPLICATION



FROM IDEA TO APPLICATION



- ✓ **FAST** – Time To Market
- ✓ Hardware **independent** simulations
- ✓ Easy To **Use-Reuse**

MBDT

MODEL-BASED DESIGN TOOLBOX

- Collection of Drivers, Libraries and Tools
- Embedded systems design and deployment on NXP MCUs directly from **Simulink**



MATHWORKS ECOSYSTEM MATLAB/SIMULINK

- Model-Based Design
- Simulation
- Automatic Code Generation
- Verification and Validation

MODEL-BASED DESIGN TOOLBOX – SUPPORTED PLATFORMS



S32K1xx

S32K3xx

MPC57xx

S12ZVMx

S32G2

S32S2

AUTOMOTIVE

IMXRT1xxx

KVx

DSC

**INDUSTRIAL
& IoT**

S32V234

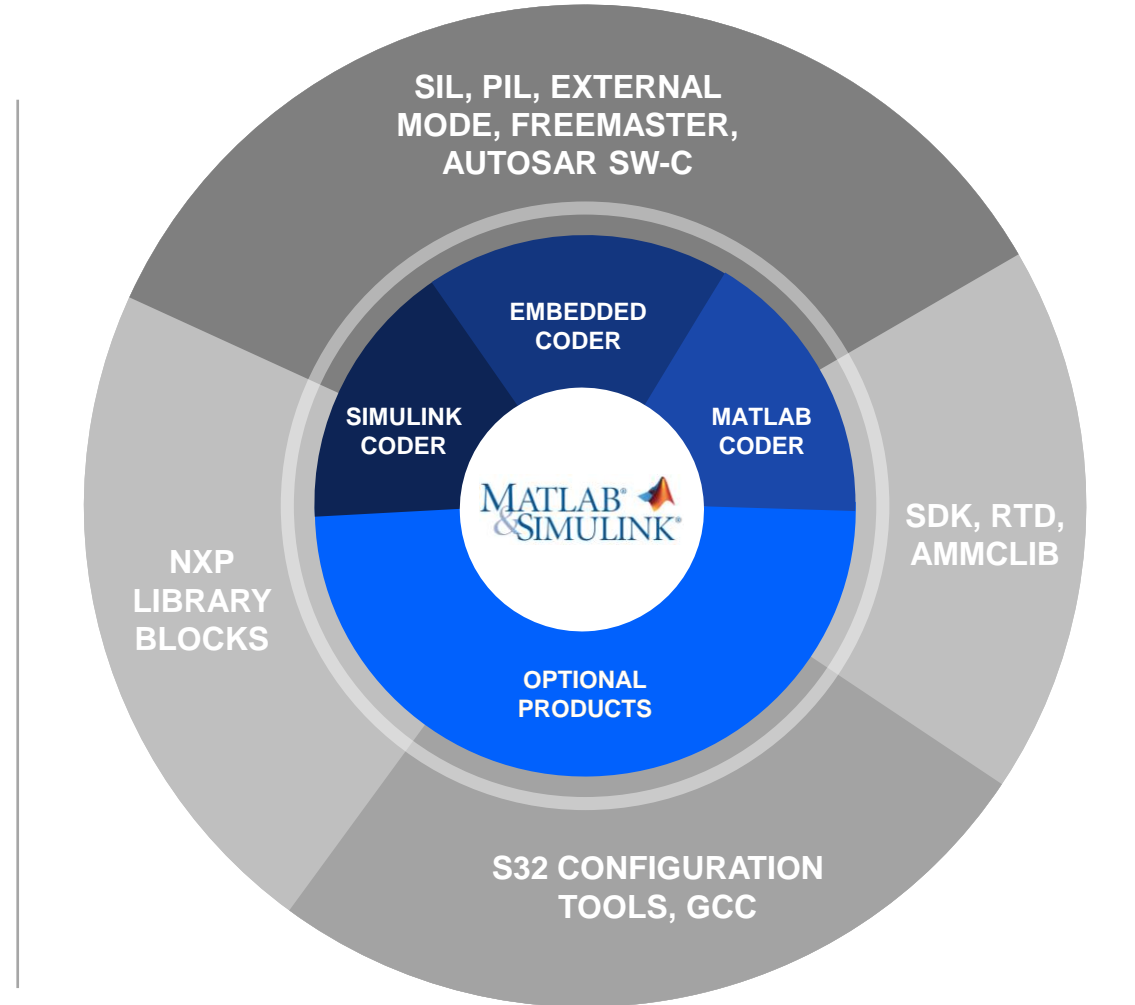
VISION

S32R41

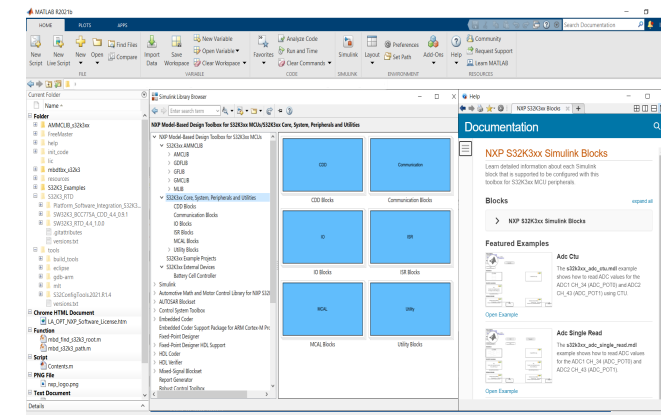
RADAR

MODEL-BASED DESIGN TOOLBOX – SOFTWARE DEVELOPMENT ENVIRONMENT AND MODULES

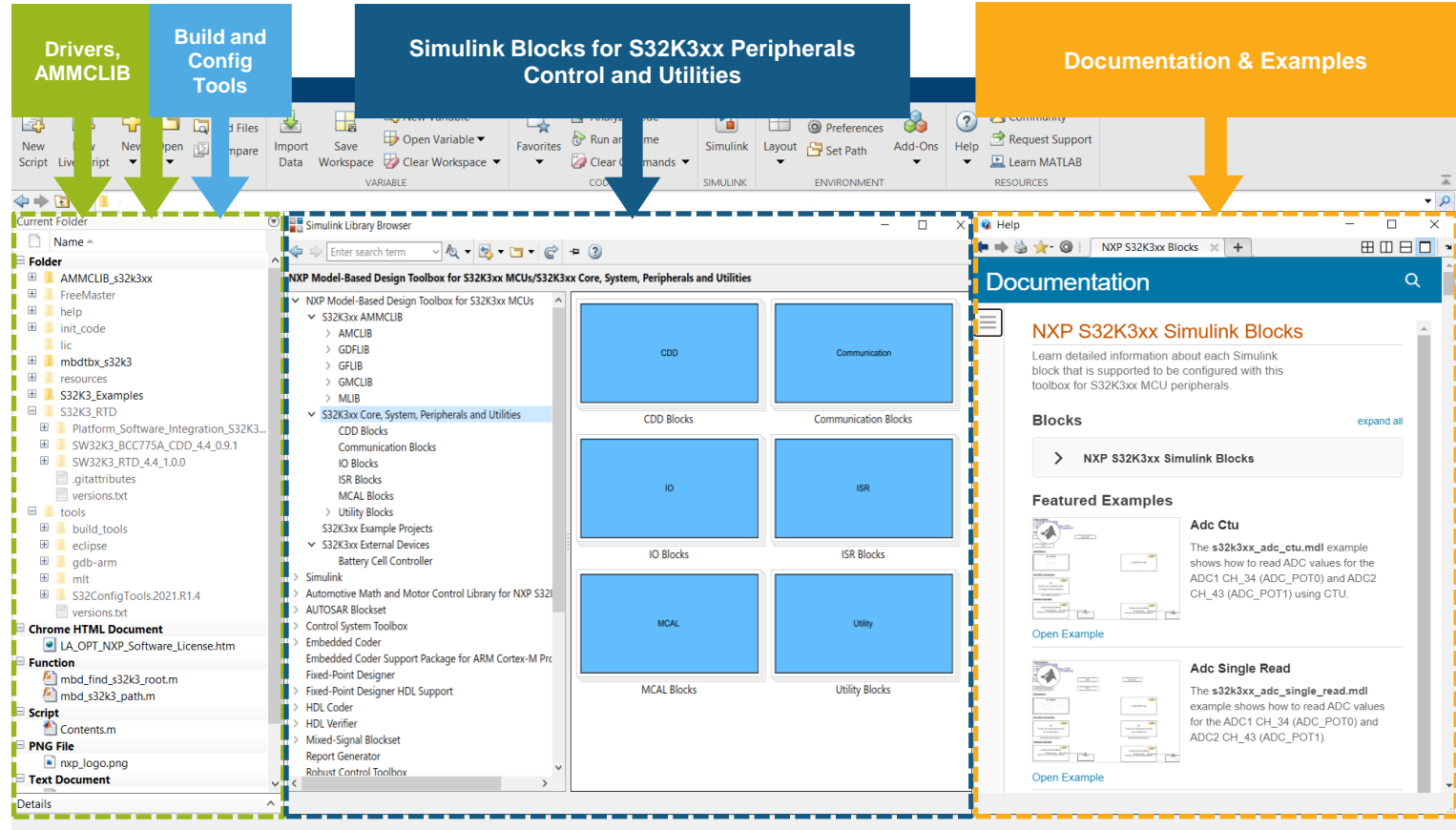
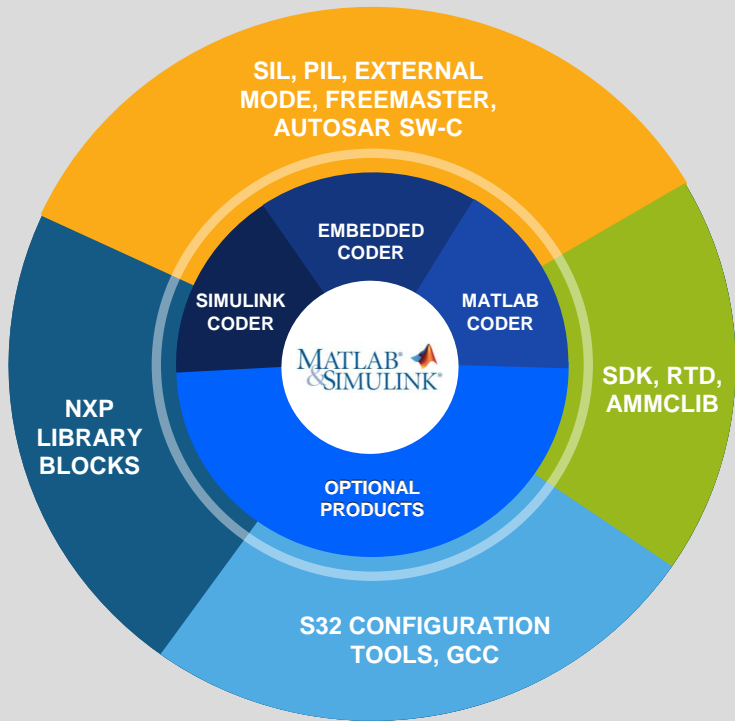
- Stateflow
- Simscape
- Motor Control Blockset
- AUTOSAR Blockset
- Deep Learning Toolbox
- Vehicle Network Toolbox



MODEL-BASED DESIGN TOOLBOX OVERVIEW



MODEL-BASED DESIGN TOOLBOX - OVERVIEW



- ✓ MCU Peripherals **Configuration & Control** using NXP's **Real-Time Drivers (RTD)** / **Software Development Kit (SDK)** APIs
- ✓ **External Tools** integration for peripherals, pins and clocks configuration; **Build Tools**
- ✓ **MBDT** blocks generate code on top of RTD/SDK
- ✓ **Example applications** covering all the toolbox features and functionalities

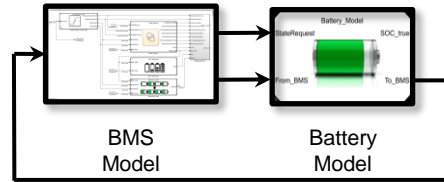
* AMMCLIB – Automotive Math and Motor Control Library

BATTERY MANAGEMENT SYSTEM – DEVELOPMENT FLOW WITH MATHWORKS AND NXP



MODEL-BASED DESIGN DEVELOPMENT FLOW

Idea Incubation

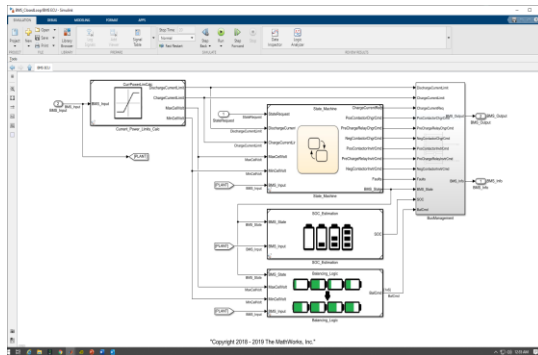


PC Environment

Step 1 – System Requirements

Model-in-the-Loop

- Software requirements
- Control system requirements
- Overall application control strategy

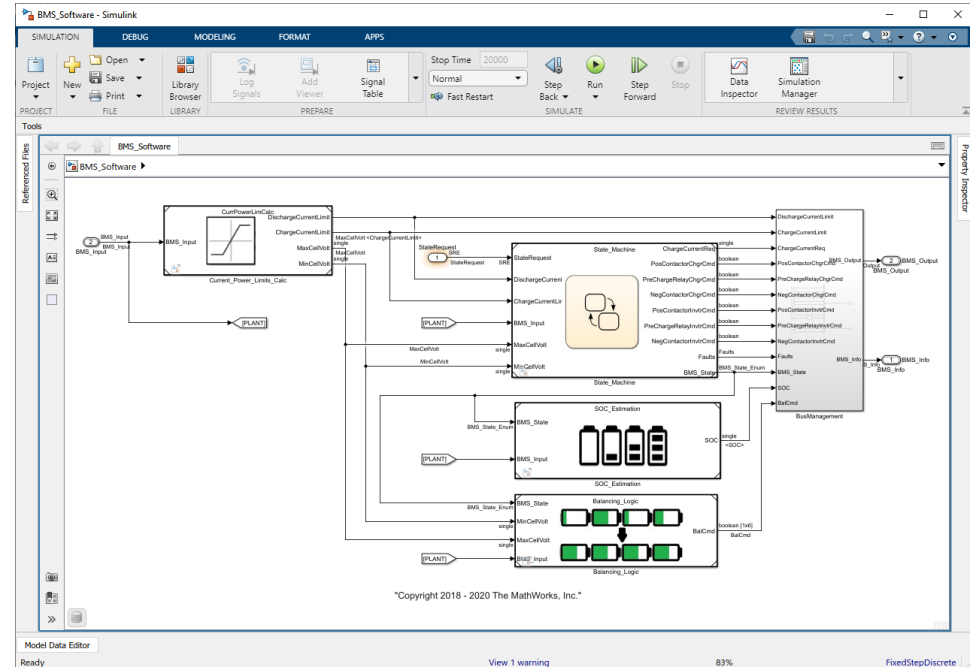
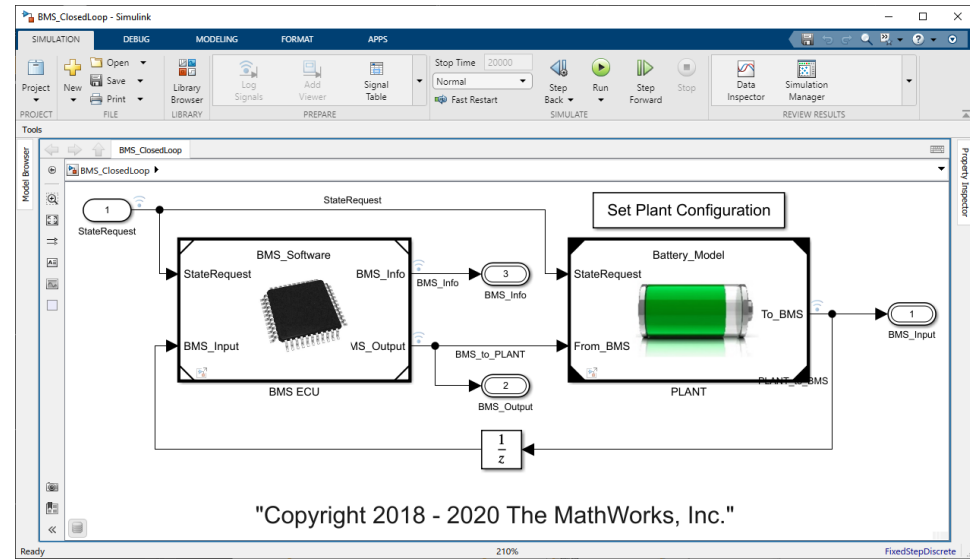
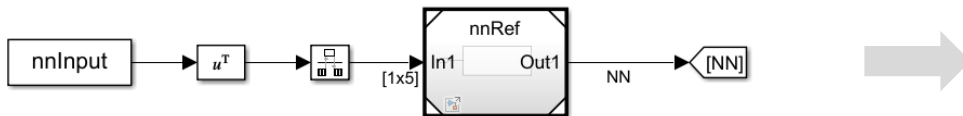


IDEA INCUBATION MODEL-IN-THE-LOOP

MIL

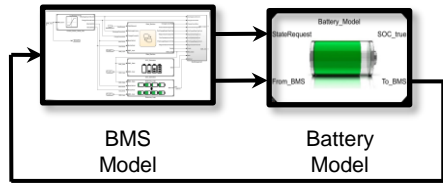
Model-in-the-Loop

- Software requirements
- Control system requirements
- Overall application control strategy
- Model testing



MODEL-BASED DESIGN DEVELOPMENT FLOW

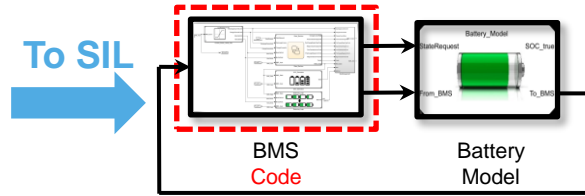
Idea Incubation



Step 1 – System Requirements Model-in-the-Loop

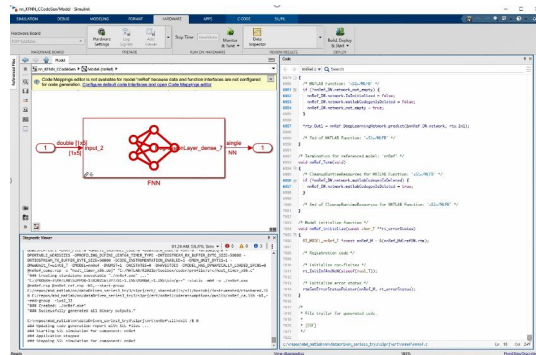
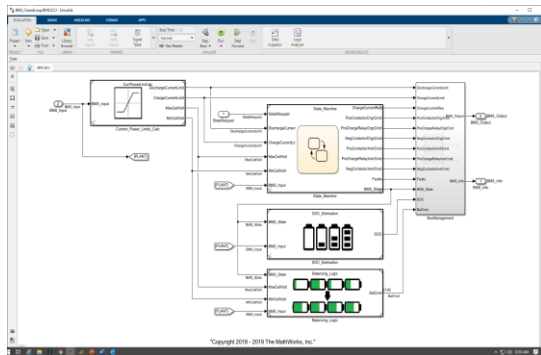
- Software requirements
- Control system requirements
- Overall application control strategy

Automatic Code Generation



Step 2 – Modeling/Simulation Software-in-the-Loop

- Control algorithm design
- Code generation preparation
- Control system design
- Start testing implementation approach

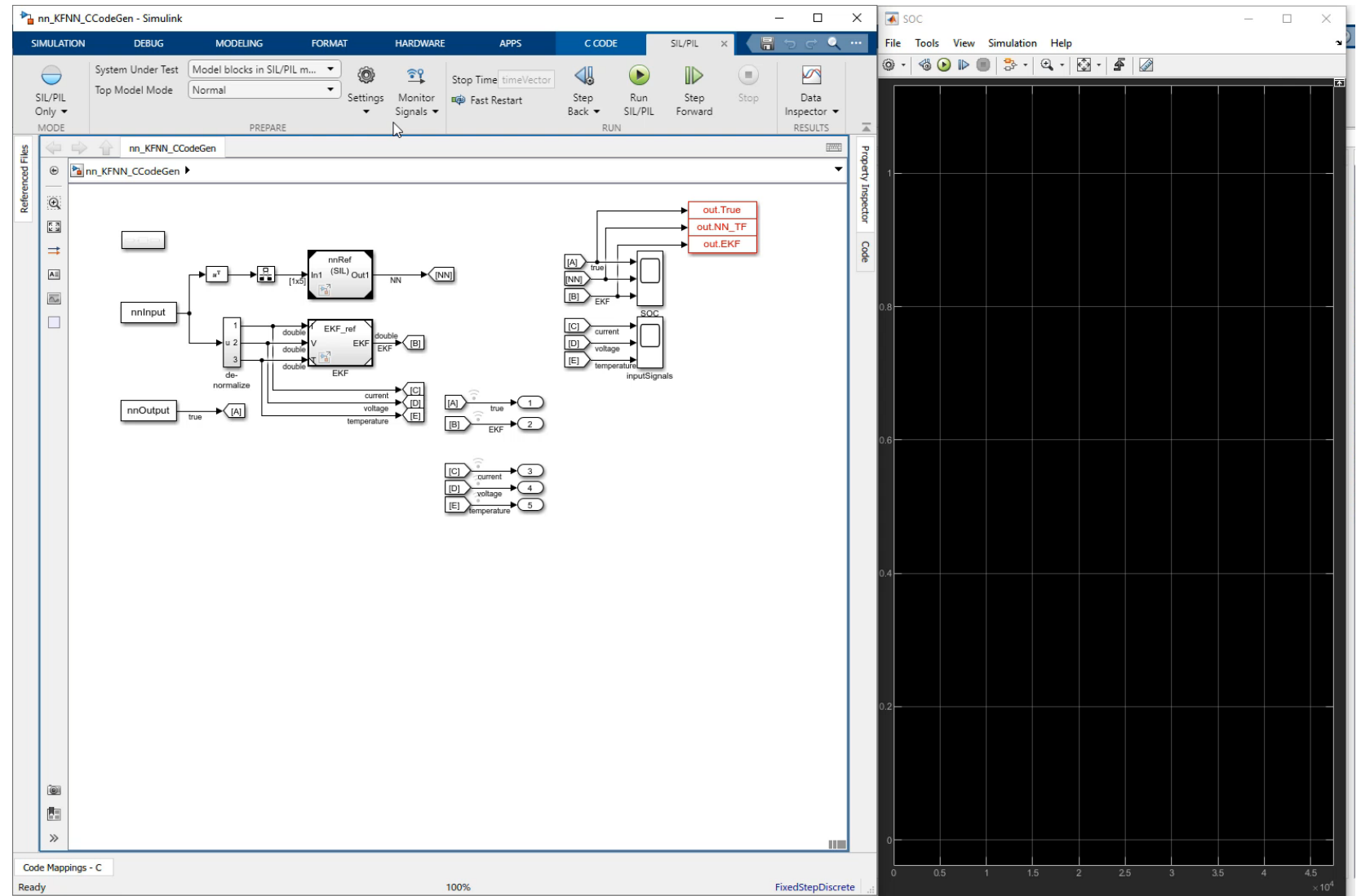


AUTOMATIC CODE GENERATION SOFTWARE-IN-THE-LOOP

SIL

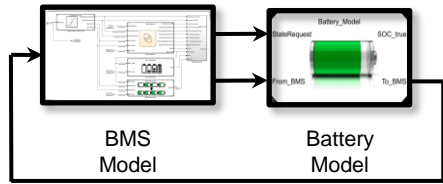
Software-in-the-Loop

- Control algorithm design
- Code generation preparation
- Control system design
- Start testing implementation approach

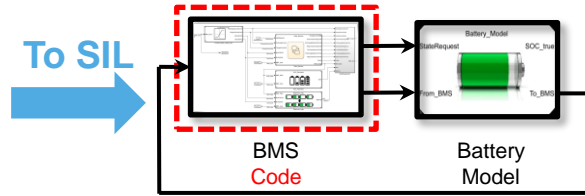


MODEL-BASED DESIGN DEVELOPMENT FLOW

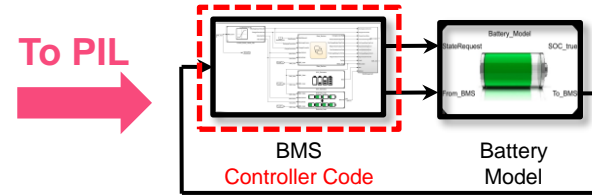
Idea Incubation



Automatic Code Generation



Code Validation



Step 1 – System Requirements Model-in-the-Loop

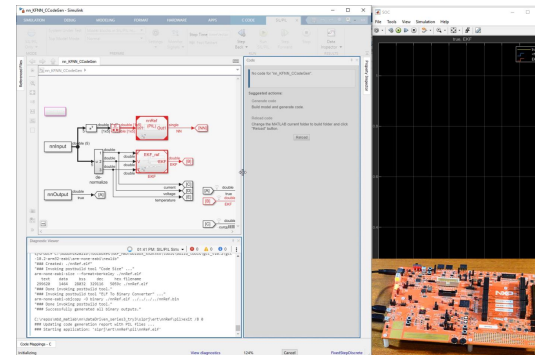
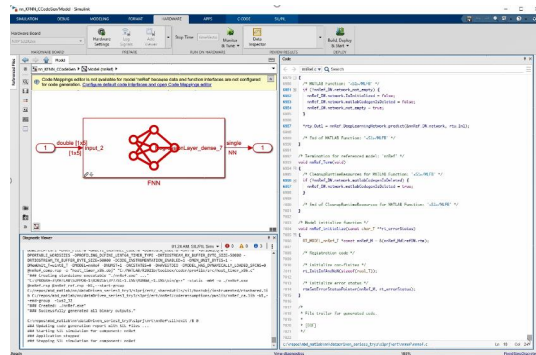
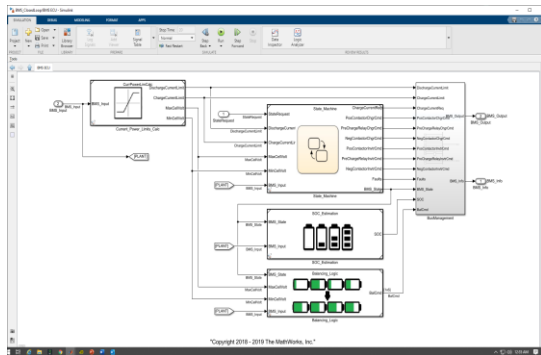
- Software requirements
- Control system requirements
- Overall application control strategy

Step 2 – Modeling/Simulation Software-in-the-Loop

- Control algorithm design
- Code generation preparation
- Control system design
- Start testing implementation approach

Step 3 – Rapid Prototype Processor-in-the-Loop

- Controller code generation
- Determine execution time on MCU
- Verify algorithm on MCU
- Check memory/stack usage on MCU

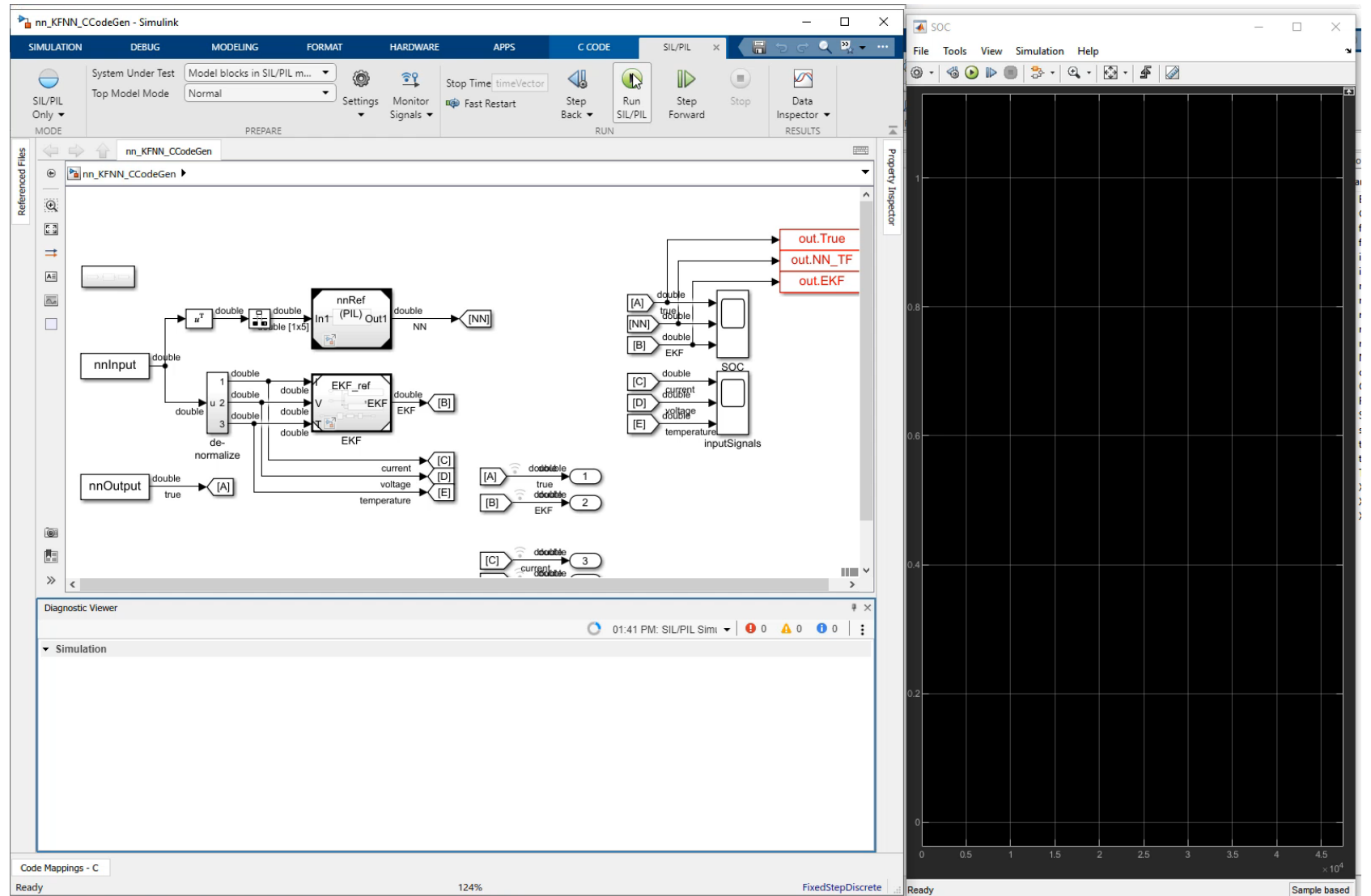


AUTOMATIC CODE GENERATION PROCESSOR-IN-THE-LOOP

PIL

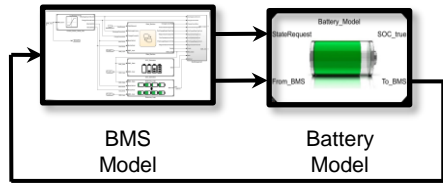
Processor-in-the-Loop

- Controller code generation
- Determine execution time on MCU
- Verify Algorithm on MCU
- Check memory/stack usage on MCU



MODEL-BASED DESIGN DEVELOPMENT FLOW

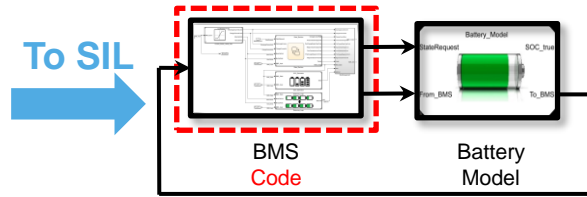
Idea Incubation



Step 1 – System Requirements Model-in-the-Loop

- Software requirements
- Control system requirements
- Overall application control strategy

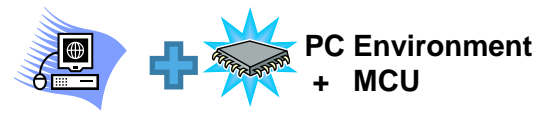
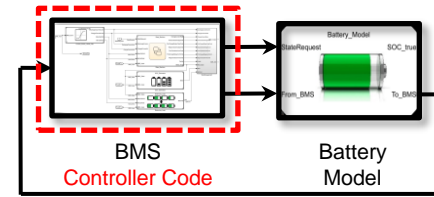
Automatic Code Generation



Step 2 – Modeling/Simulation Software-in-the-Loop

- Control algorithm design
- Code generation preparation
- Control system design
- Start testing implementation approach

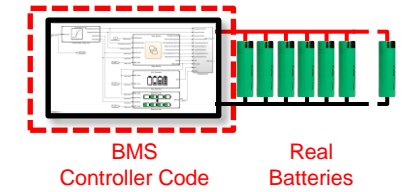
Code Validation



Step 3 – Rapid Prototype Processor-in-the-Loop

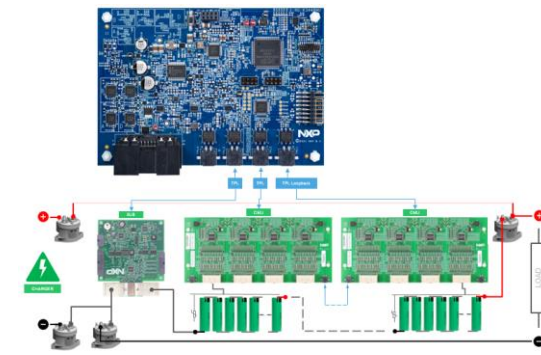
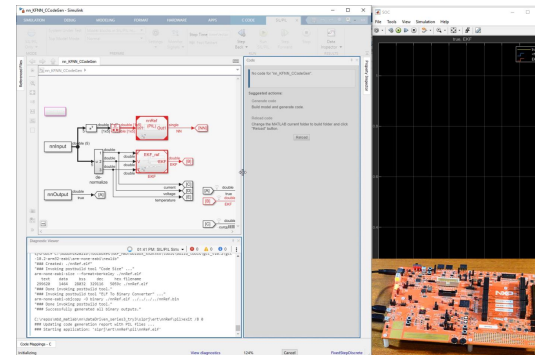
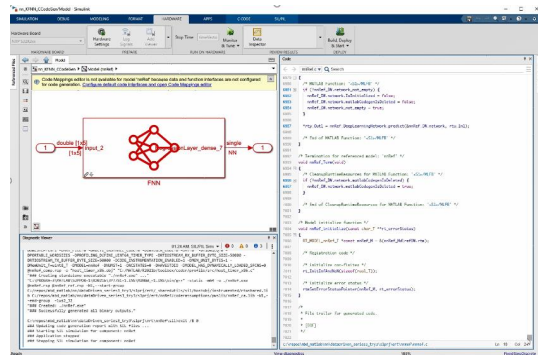
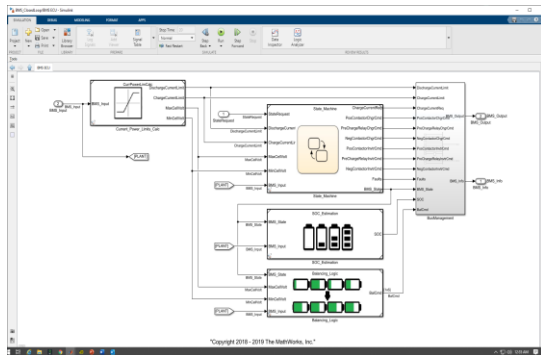
- Controller code generation
- Determine execution time on MCU
- Verify algorithm on MCU
- Check memory/stack usage on MCU

Prototype



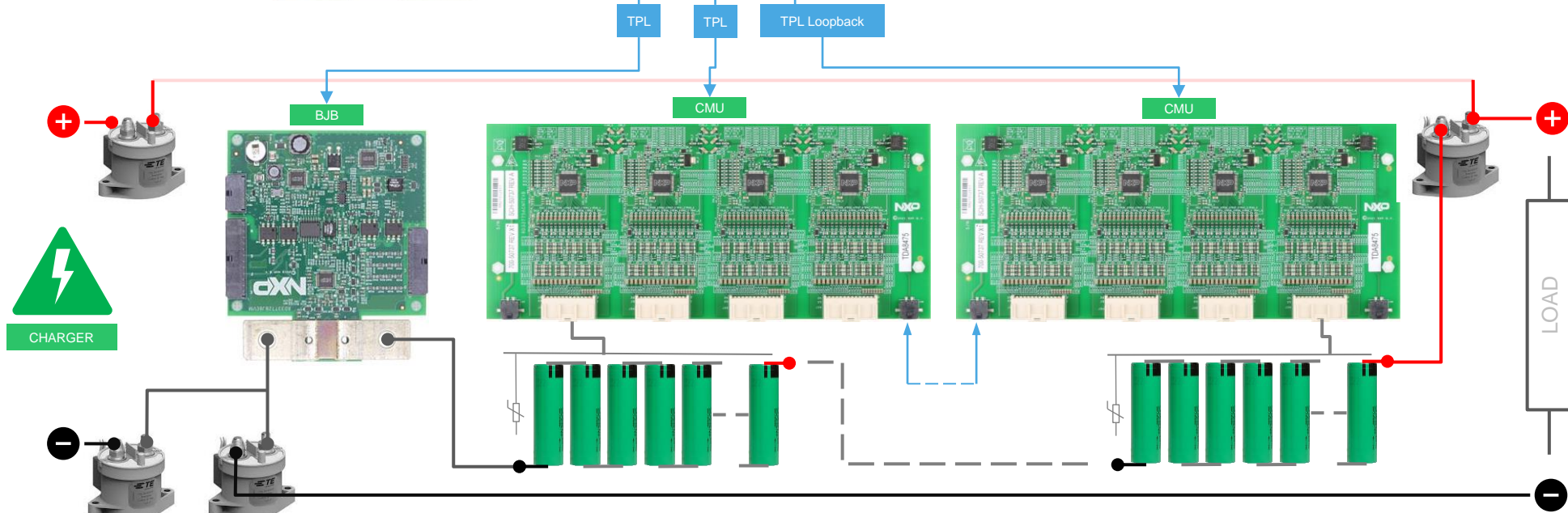
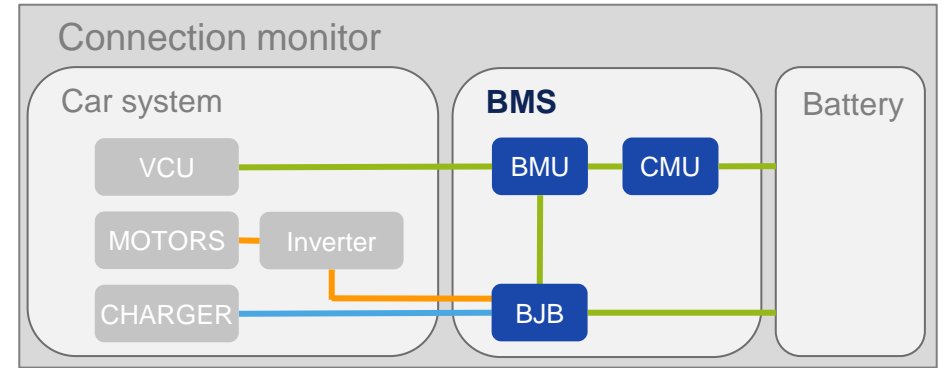
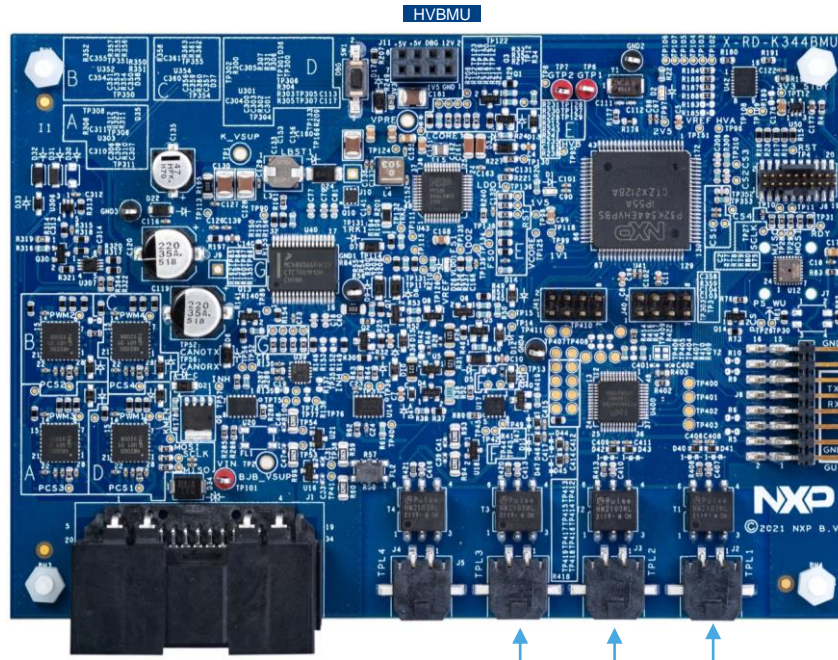
Step 4 – Target MCU Implementation MCU Final Application

- Validation/verification phase
- Controller code generation
- Test system in target environment using tools for data logging and parameter tuning



HIGH VOLTAGE BMS DESIGN

Prototype

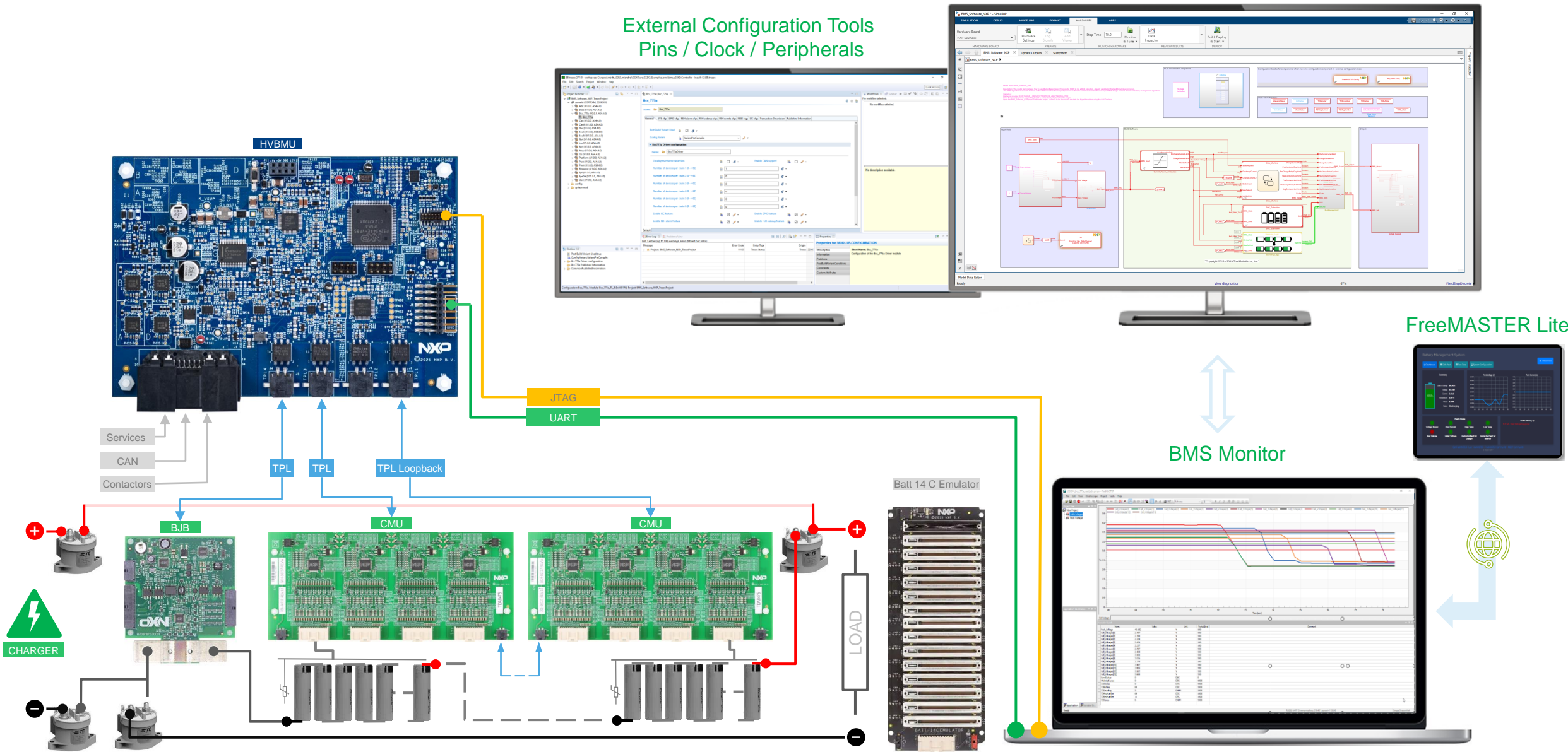


i Images presented are for illustration purposes only and may not be an exact representation of the product – their purpose is just exemplification of a concept.

BMS RAPID PROTOTYPING - MBDT ENVIRONMENT

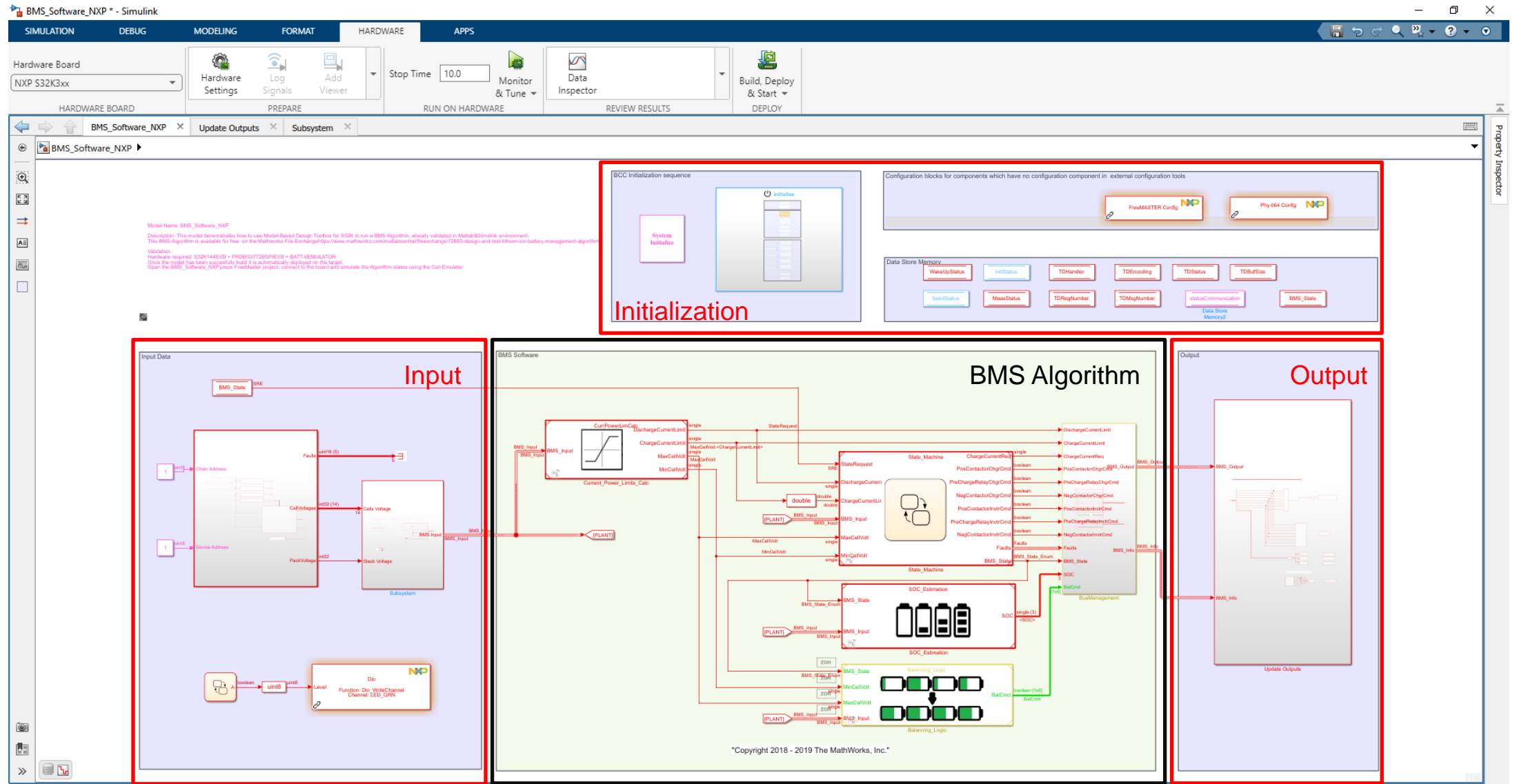
Model-Based Design Tools for Simulink

External Configuration Tools
Pins / Clock / Peripherals



Images presented are for illustration purposes only and may not be an exact representation of the product – their purpose is just exemplification of a concept.

BMS MODEL ON MBDT



Model Data Editor

Ready

View diagnostics

67%

FixedStepDiscrete

FREEMASTER DEMO INTERFACE



ADDITIONAL RESOURCES & SUPPORT



MBDT Beginner's Guide

- W1: MBDT Introduction
- W2: How-To SPI
- W3: How-To CAN
- W4: How-To PWM
- W5: How-To LIN
- W6: How-To PIL
- W7: How-To Timers

Co-hosted Webinars

- Motor Control: S32K
- Motor Control: i.MX RT
- Motor Control: BLDC/PMSM
- Motor Control: Design Application Code Generation and Verification
- Speed Up Applications Development with MBDT
- AUTOSAR SW on S32K1/MPC
- AUTOSAR SW on S32K3
- Deploying BMS algorithm on S32K1
- Deploying Deep Learning SOC algorithm on S32K3
- Vision
- FreeMASTER

Announcement

MBDT for HCP Release Announcement:
NXP Model-Based Design Toolbox for High-Performance Computing Platform (HCP) - version 1.1.0 RFP

Model-Based Design Tools for Matlab and Simulink Support

S32K1xx <ul style="list-style-type: none">How toTutorialsVideosFAQ	MPC57xx <ul style="list-style-type: none">How toTutorialsVideosFAQ	S12ZVM <ul style="list-style-type: none">How toTutorialsVideos
S32K3xx <ul style="list-style-type: none">How toTutorialsVideosFAQ	HCP <ul style="list-style-type: none">How toTutorialsVideos	MathWorks <ul style="list-style-type: none">Motor ControlEmbedded Coder®Stateflow®Simulink™VideosSupport PackagesPolyspace
i.MX RT <ul style="list-style-type: none">How toTutorialsVideosFAQ	Kinetis V <ul style="list-style-type: none">How toTutorialsVideos	DSC <ul style="list-style-type: none">How toTutorialsVideos

The Model Based Design Toolbox provides an integrated development environment and toolchain for configuring and generating all of the necessary software automatically. Learn more.

Discussions

- MPC5748G PIL timeout error**
by [chamioncham](#) yesterday • Latest post 9 hours ago by [chamioncham](#)
- MLIB and SPI compilation error**
by [equiper_alita](#) 3 hours ago • Latest post yesterday by [equiper_alita](#)
- Code generated by mbdtool not excute on the MPC574...**
by [m1387176142](#) on 04-05-2020 04:52 PM • Latest post Tuesday by [eusebio_bivoli](#)

ASK A QUESTION

- 0 1
- 0 1
- 0 9

Contents

- NXP Model-Based Design Tools Knowledge Base 94
- Hotfixes**
- S32K1xx
- MPC57xx
- S12ZVM
- i.MX RT
- Kinetis V

PMSM Control Workshop

Course Main Page

- M1: Environment Setup
- M2: PMSM and FOC
- M3: System Partitioning
- M4: PWM Modulation
- M5: V/f Scalar Control
- M6: Current Sensing
- M7: Torque Control
- M8: Speed Control
- M9: Position Observer
- M10: Sensorless Speed Control

BLDC Control Workshop

Course Main Page

- 1. Introduction
- 2. Application Partitioning
- 3. Input Commands
- 4. BLDC Motor Theory
- 5. Hall Sensors
- 6. Commutation
- 7. Commutation Algorithm
- 8. Power Stage Config
- 9. Open Loop Control
- 10. Speed Estimator
- 11. Closed Loop Control
- 12. Motor Control System



MATLAB EXPO

Thank you



© 2022 The MathWorks, Inc. MATLAB and Simulink are registered trademarks of The MathWorks, Inc. See [mathworks.com/trademarks](https://www.mathworks.com/trademarks) for a list of additional trademarks. Other product or brand names may be trademarks or registered trademarks of their respective holders.