



MATLAB AUTOMOTIVE 2022: AUTOMOTIVE DEVOPS FOR MODEL-BASED DESIGN

Curt Hillier, Technical Director,
Automotive Systems Engineering

APRIL 5, 2022



SECURE CONNECTIONS
FOR A SMARTER WORLD

PUBLIC

NXP, THE NXP LOGO AND NXP SECURE CONNECTIONS FOR A SMARTER WORLD ARE TRADEMARKS OF NXP B.V.
ALL OTHER PRODUCT OR SERVICE NAMES ARE THE PROPERTY OF THEIR RESPECTIVE OWNERS. © 2022 NXP B.V.



Vehicles are undergoing a transformation from mostly mechanically defined features and capabilities to software defined.

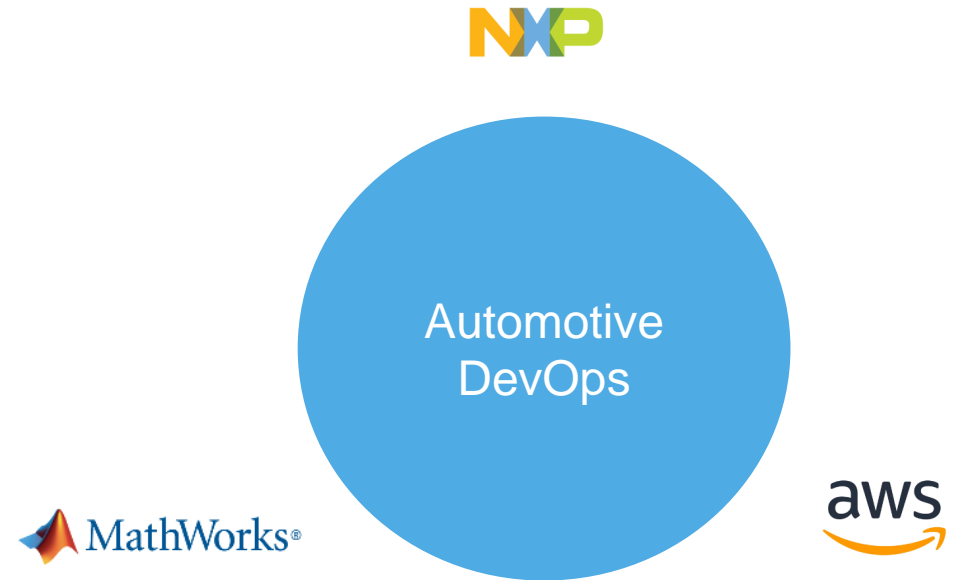
The key automotive industry trends are:

- transition to Agile development
- increased size of software development teams
- migration of tools and workflows to the cloud
- continued adoption of model-based design engineering

NXP, the MathWorks, and AWS have collaborated to build an example Automotive DevOps solution which can enable the future of Model-based design

Let coders do more coding!

Mechanical  Software

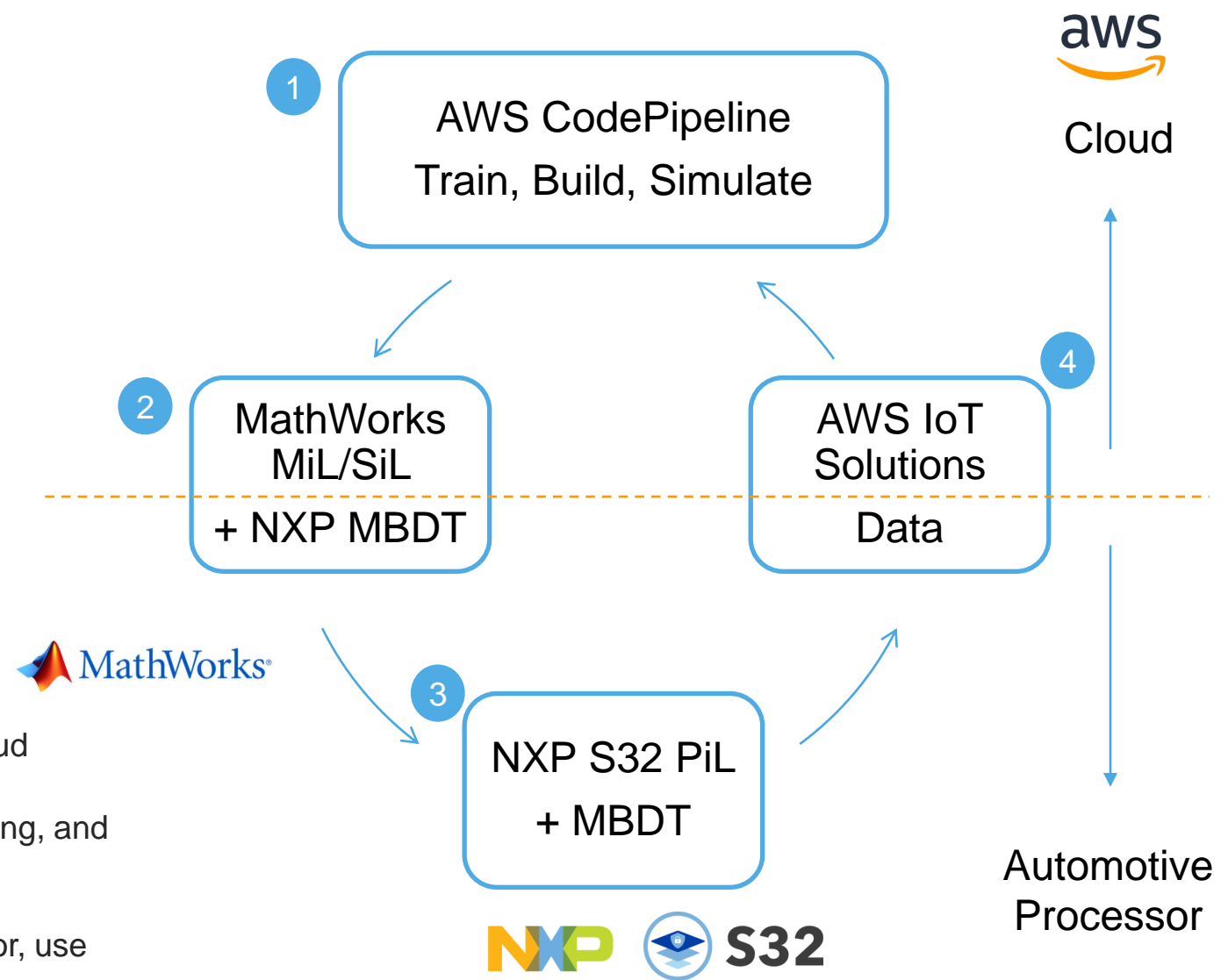


OVERVIEW

- The Automotive DevOps model-based design solution incorporates:
 - AWS CodeSuite services
 - MathWorks® model-based design tools
 - Advanced vehicle control algorithms executing on NXP Automotive processors
- The solution allows users to develop and simulate in the cloud, and then easily deploy to Automotive silicon for algorithm validation.

Major components supporting the solution include:

- 1 **AWS CodePipeline:** Build and simulate models in the cloud
- 2 **MathWorks with NXP MBDT:** tools for designing, simulating, and implementing automotive software and system models
- 3 **NXP GoldBox:** execute algorithm on Automotive processor, use profiler to measure execution time
- 4 **AWS IoT Solutions:** publish data to the cloud



MiL = Model in the Loop
SiL = Software in the Loop
PiL = Processor in the Loop
MBDT = Model-based Design Toolbox Add On

Solution Products



SECURE CONNECTIONS
FOR A SMARTER WORLD



PUBLIC

NXP, THE NXP LOGO AND NXP SECURE CONNECTIONS FOR A SMARTER WORLD ARE TRADEMARKS OF NXP B.V.
ALL OTHER PRODUCT OR SERVICE NAMES ARE THE PROPERTY OF THEIR RESPECTIVE OWNERS. © 2022 NXP B.V.



AWS PRODUCTS: CODEPIPELINE

Why did NXP choose to work with AWS?

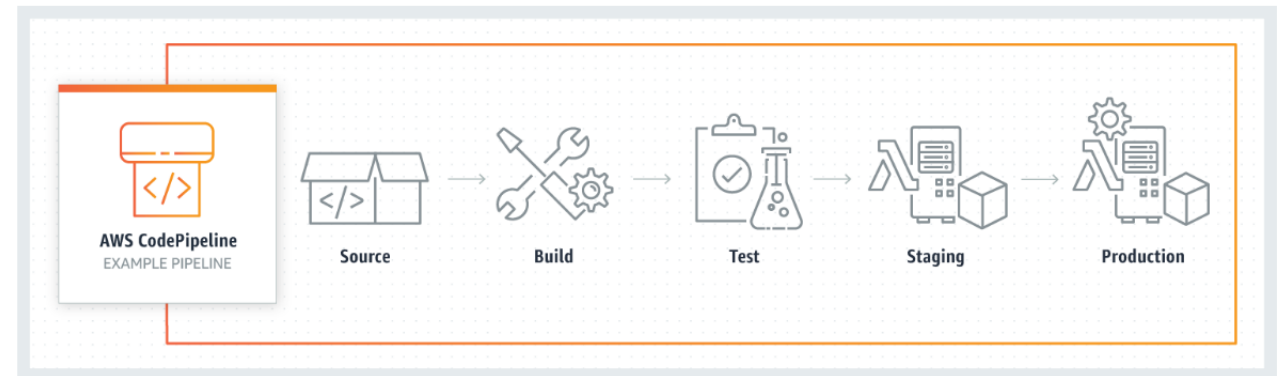
- Numerous solutions for connected vehicles
- Strong collaboration
- Developing leading edge solutions
- Easy to use



AWS CodePipeline:

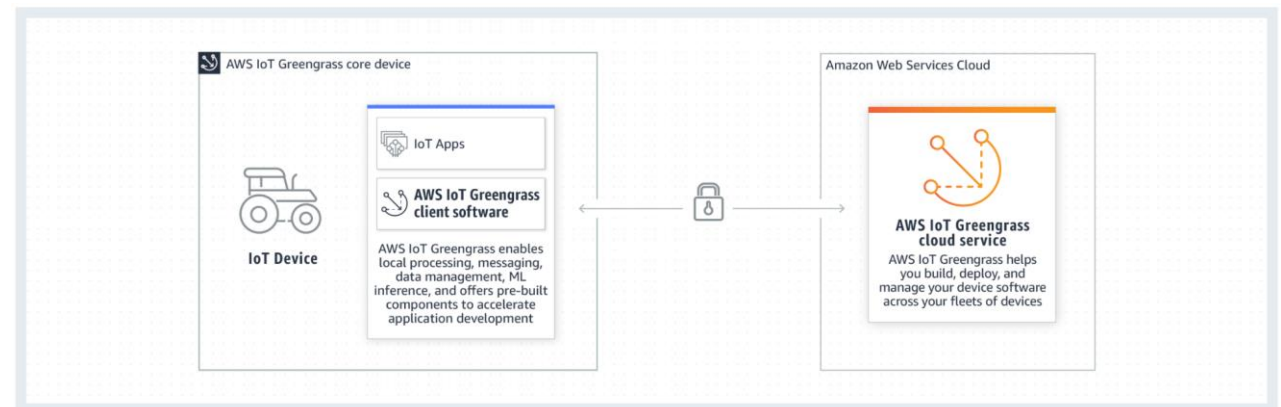
- fully managed [continuous delivery](#) service that helps you automate your release pipelines for fast and reliable application and infrastructure updates.

How it works



AWS Greengrass & IoT Solutions:

- open-source edge runtime and cloud service for building, deploying, and managing device software.



MATHWORKS PRODUCTS

Why did we choose to work with the MathWorks?

- Domain Expertise – across numerous domains
- toolchain for software and simulation development
- Quick to get started: pre-built examples, quality, documentation.

The HEV Model Predictive Control application uses these key MathWorks products:

- Simulink
- Powertrain Blockset
- Vehicle Dynamics Blockset
- Embedded Coder

Allows us to build a car (use a pre-built model), generate C code, and run simulation



Powertrain Blockset

Search MathWorks.com

Powertrain Blockset

Model and simulate automotive powertrain systems

[Request a free trial](#) [Request a quote](#)

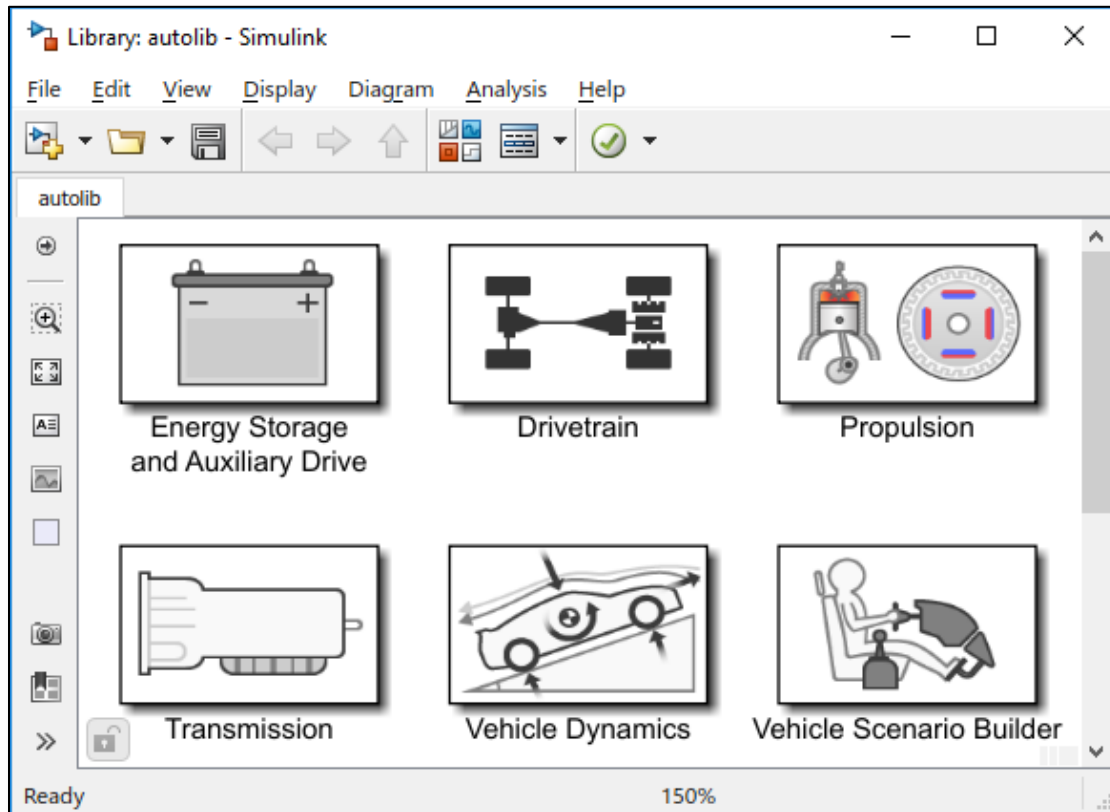
Powertrain Blockset™ provides fully assembled reference application models of automotive powertrains, including gasoline, diesel, hybrid, and electric systems. It includes a component library for simulating engine subsystems, transmission assemblies, traction motors, battery packs, and controller models. Powertrain Blockset also includes a dynamometer model for virtual testing. MDF file support provides a standards-based interface to calibration tools for data import.

1:52

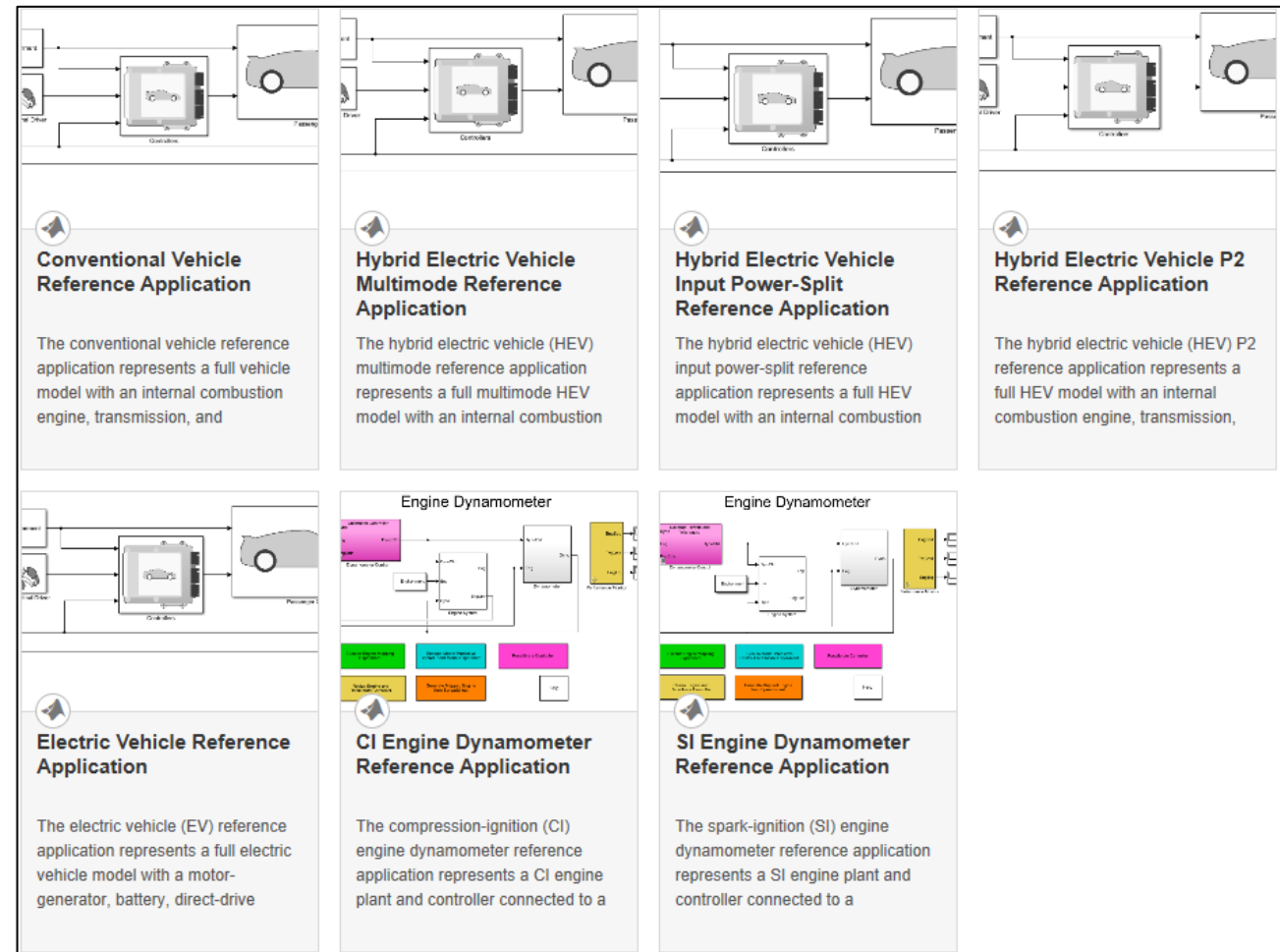


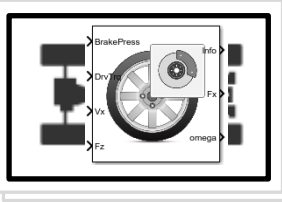
Powertrain Blockset™

Library of blocks

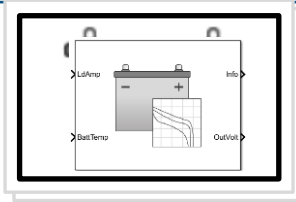


Pre-built reference applications

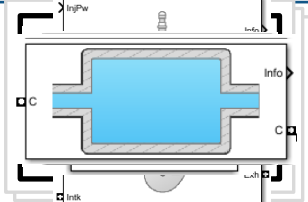




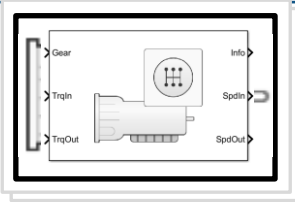
Drivetrain



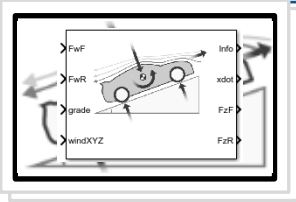
Energy Storage and Auxiliary Drive



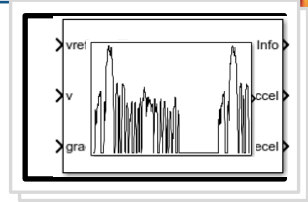
Propulsion



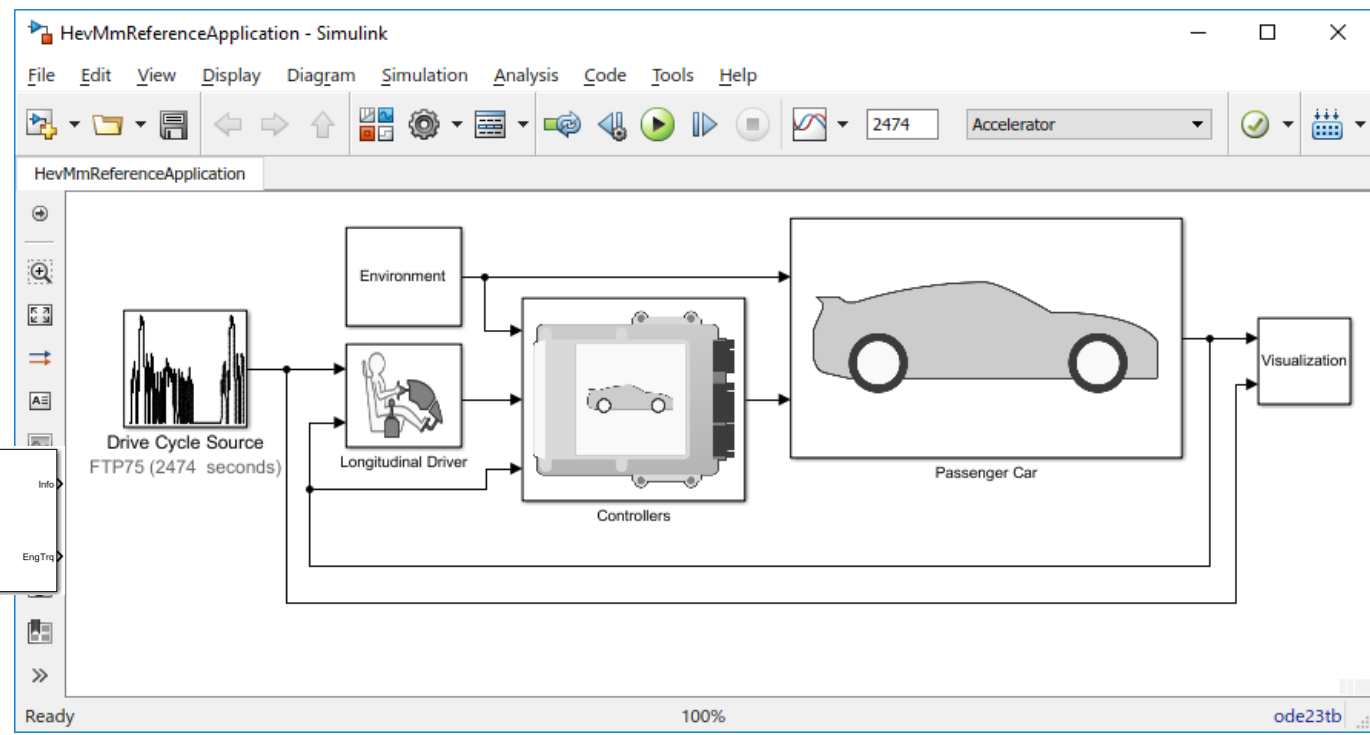
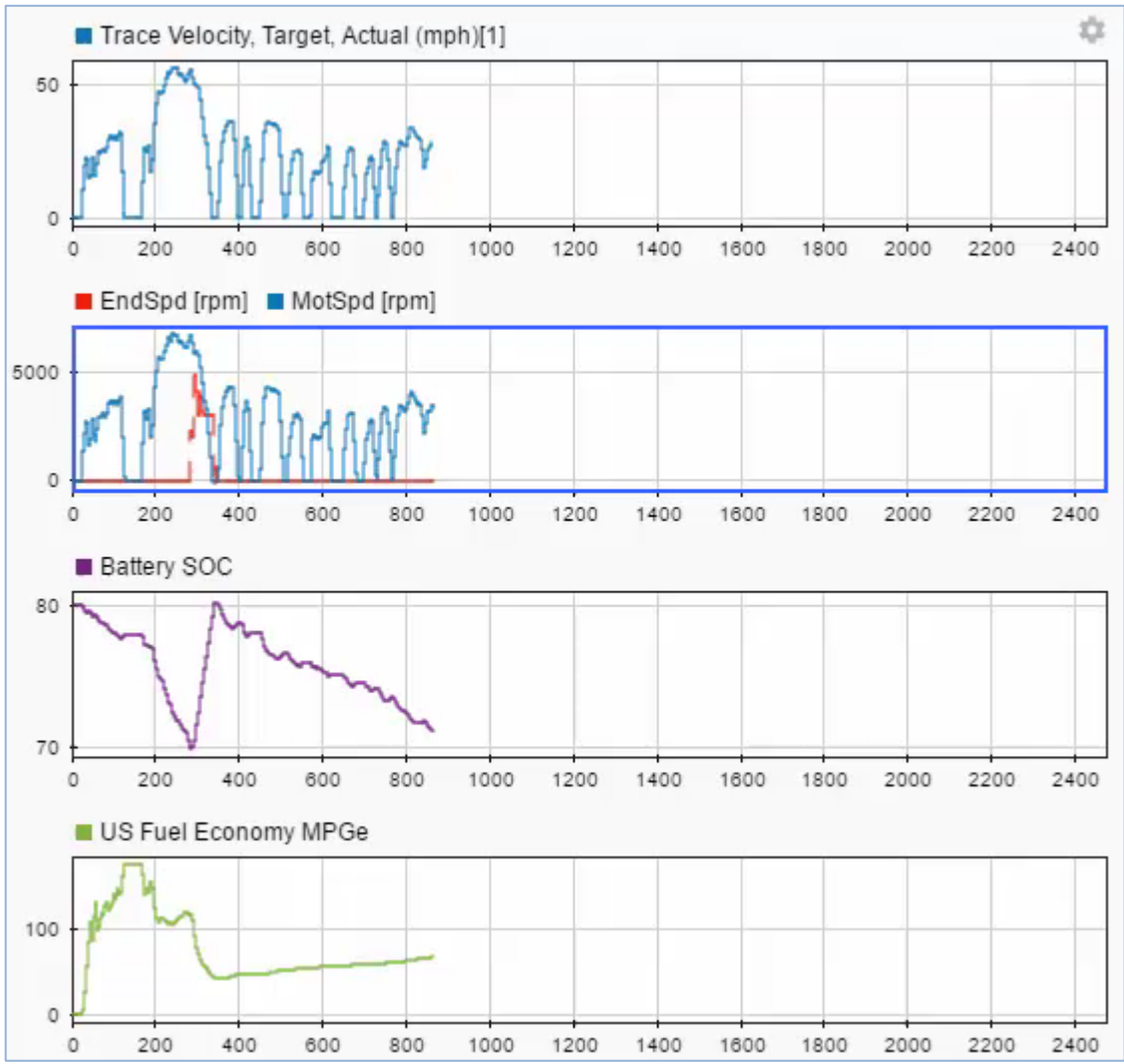
Transmission



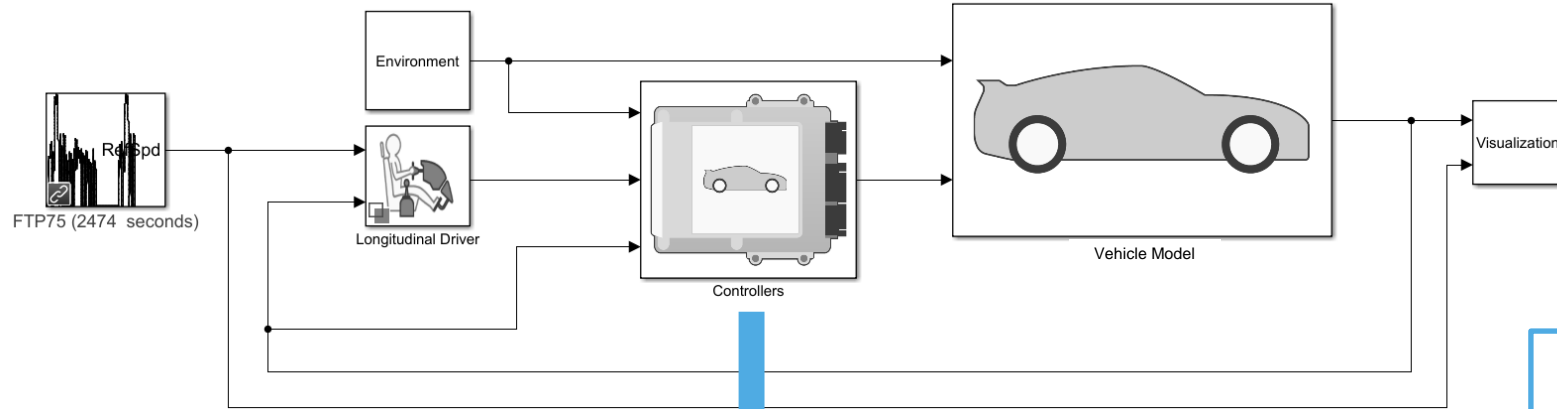
Vehicle Dynamics



Vehicle Scenario Builder



MBDT OVERVIEW: PROCESSOR IN THE LOOP (PIL)



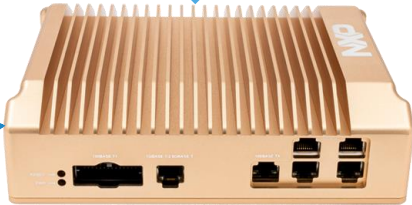
NXP MBDT:

- brings the power of Model-based design to NXP Automotive real-time processors and vehicle network processors
- Excellent for early prototyping and performance measurements

NXP MBDT "HCP"

Service Oriented Gateway

- in-vehicle secure networking
- edge processing
- cloud interface



NXP **S32G2** GoldBox Vehicle Network Processor



NXP **S32S** GreenBox 2 Real-Time Controller

xEV Propulsion Controller

- Battery SoX algorithms
- Energy Management
- Motor control



Automotive DevOps Demonstration



SECURE CONNECTIONS
FOR A SMARTER WORLD

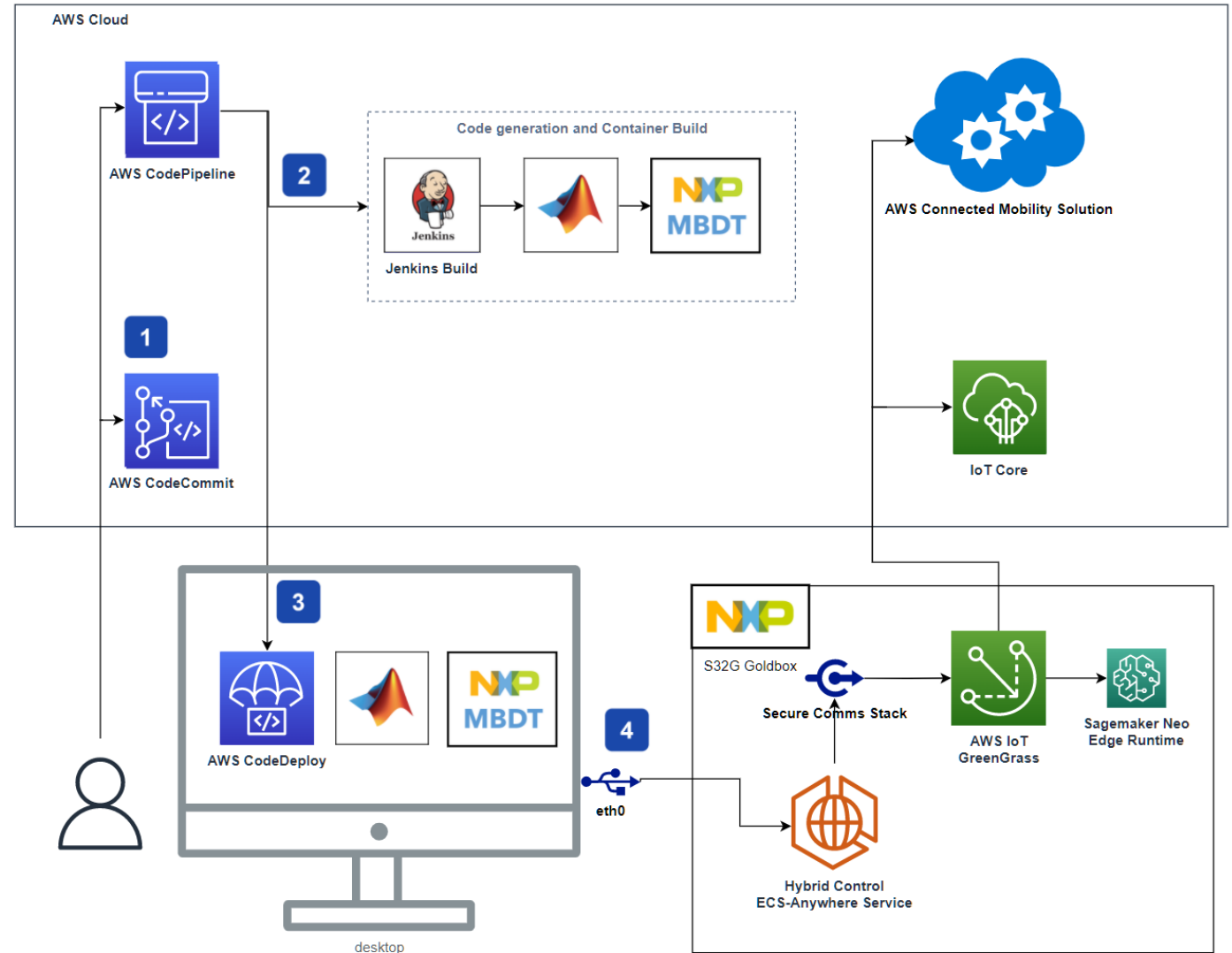


PUBLIC



LET'S WALK THROUGH THE DEMONSTRATION: "LOCAL DESKTOP"

- 1 Code Commit
- 2 Build
- 3 Deploy to target
- 4 PiL Simulation with GUIs / metrics and performance profiling



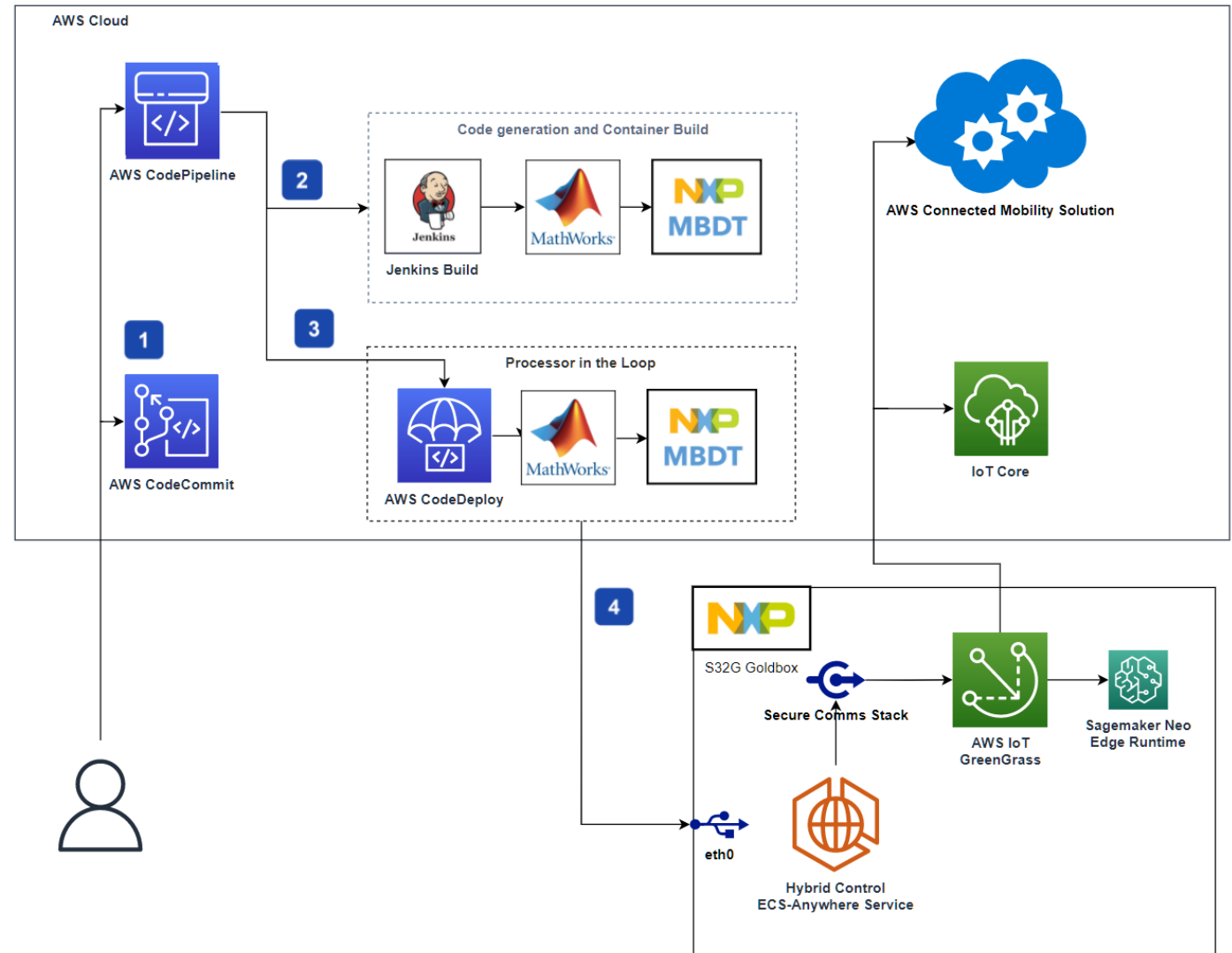
LET'S WALK THROUGH THE DEMONSTRATION: "ALL CLOUD TOOLCHAIN"

1 Code Commit

2 Build

3 Deploy to target

4 PiL Simulation with GUIs /
metrics and performance
profiling

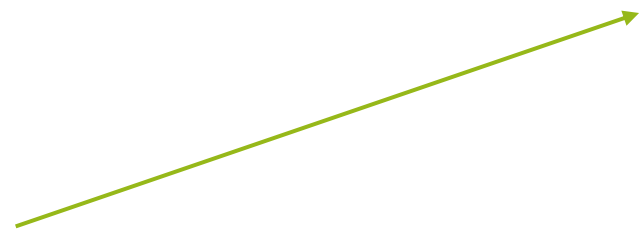


AWS CODEPIPELINE EXAMPLE

The screenshot displays the AWS CodePipeline console for a pipeline named 'nxp-mathworks'. The pipeline execution ID is 'ea0bac40-661c-44d2-aab6-baee969ed6b2'. The pipeline consists of three stages, all of which have succeeded:

- Source** (Succeeded):
 - CodeCommit_Source (AWS CodeCommit) succeeded 5 minutes ago with ID 'c6359066'.
 - CodeCommit_Source: clear mex
- JenkinsBuild** (Succeeded):
 - JenkinsBuild (Custom MyJenkinsProvider (Version: 3)) succeeded just now.
 - CodeCommit_Source: clear mex
- Deploy** (Succeeded):
 - CodeDeploy (AWS CodeDeploy) succeeded just now.
 - CodeCommit_Source: clear mex

Code commit

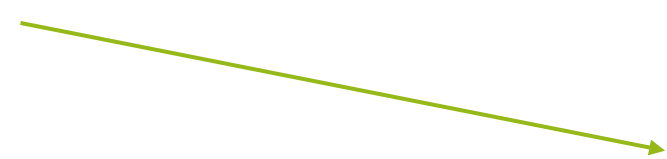


Code build

(MATLAB codegen and NXP MBDT)



Code deploy



JENKINS BUILD OUTPUT

- Create elf file for target execution on S32G

```
### Created: ./MPC_DBL_Target.elf
### Invoking postbuild tool "Code Size" ...
aarch64-fsl-linux-size --format=berkeley ./MPC_DBL_Target.elf
  text    data    bss    dec    hex filename
 78301   2000   416952  497253  79665 ./MPC_DBL_Target.elf
### Done invoking postbuild tool.
### Successfully generated all binary outputs.
```

- AWS CodePipeline Build successful

```
[AWS CodePipeline Plugin] Upload successful
[AWS CodePipeline Plugin] Build succeeded, calling PutJobSuccessResult
Finished: SUCCESS
```

The screenshot shows a Jenkins console window titled 'localhost:8080/job/MATLAB_Build/67/console'. The output is for build #67. It shows the compilation of MATLAB code for an ARM target. A green box highlights the successful creation of the ELF file and the 'Code Size' report. Another green box highlights a warning about algebraic loops in the Simulink model 'simHEV'. The build concludes with successful artifact upload to AWS CodePipeline.

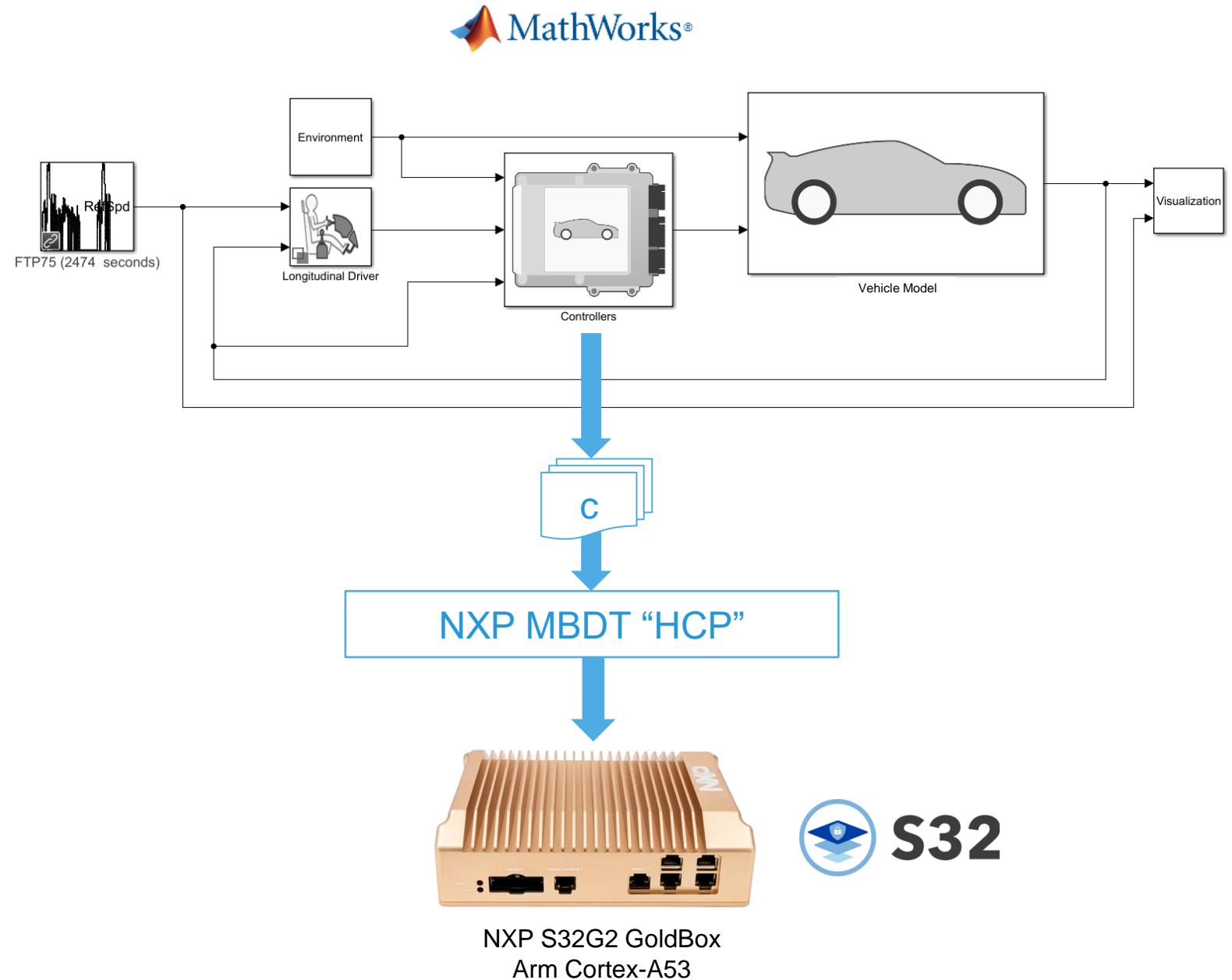
```
localhost:8080/job/MATLAB_Build/67/console
d #67
...
-o./MPC_DBL_Target.elf xil_interface_lib.o xil_data_stream.o xil_services.o xil_interface.o xilcomma_rtiostream.o xil_rtiostream.o rtiostream_utils.o
coder_assumptions_app.o coder_assumptions_data_stream.o coder_assumptions_rtiostream.o pil_main.o rtiostream_topip.o ../../../../../../slprj/ert/MPC_DBL_Target
/MPC_DBL_Target_rtwlib.a C:/Users/Administrator/Documents/MATLAB_Jenkins/ISC_Task4_S32G2/Work/slprj/ert/_sharedutils/rtwshared.a C:/Users/Administrator
/Documents/MATLAB_Jenkins/ISC_Task4_S32G2/Work/slprj/ert/MPC_DBL_Target/coderassumptions/lib/MPC_DBL_Target_ca.a -lm -lpthread

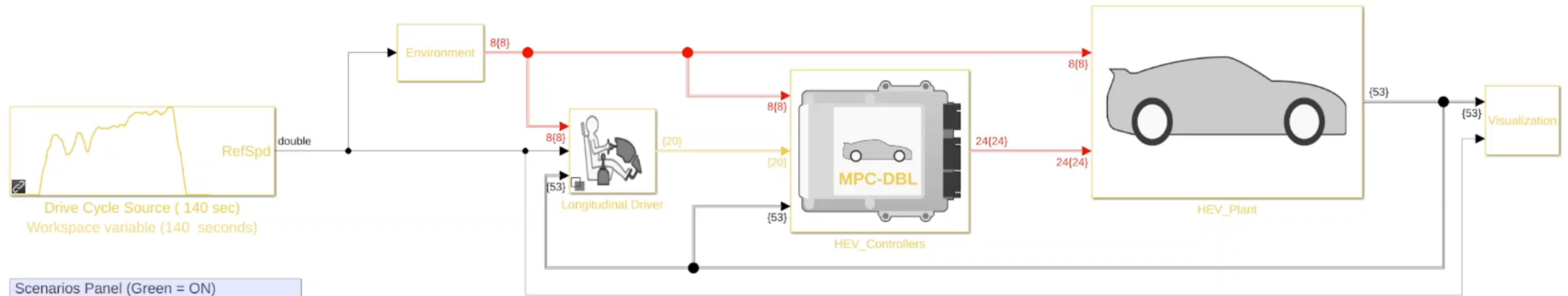
### Created: ./MPC_DBL_Target.elf
### Invoking postbuild tool "Code Size" ...
aarch64-fsl-linux-size --format=berkeley ./MPC_DBL_Target.elf
  text    data    bss    dec    hex filename
 78301   2000   416952  497253  79665 ./MPC_DBL_Target.elf
### Done invoking postbuild tool.
### Successfully generated all binary outputs.

C:\Users\Administrator\Documents\MATLAB_Jenkins\ISC_Task4_S32G2\Work\slprj\ert\MPC_DBL_Target\pil>exit 0
### Updating code generation report with PIL files ...
Caught throwable while adding doc set item to doc set builder: null
[Warning: Block diagram 'simHEV' contains 2 algebraic loop(s). To see more details about the loops use the
command <a
href="matlab:Simulink.BlockDiagram.getAlgebraicLoops(bdroot);">Simulink.BlockDiagram.getAlgebraicLoops('simHEV')
</a> or the command line Simulink debugger by typing <a href="matlab:sldbg(bdroot);">sldbg('simHEV')
</a> in the MATLAB command window. To eliminate this message, set Algebraic loop to 'none'.]
[> In build_model (line 4)]
Found algebraic loop containing:
simHEV/HEV_Plant/HEV_Mathworks/Electric Plant/BTM
simHEV/HEV_Plant/HEV_Mathworks/Electric Plant/Battery (algebraic variable) (discontinuity)
This algebraic loop may be resolved when the subsystem option 'Minimize algebraic loop occurrences' is selected on some or all of the atomic and enabled
subsystems in this algebraic loop or if 'Minimize algebraic loop occurrences' is selected on the Model Referencing pane of referenced models in this
algebraic loop
Found algebraic loop containing:
simHEV/Environment1/Switch3 (algebraic variable) (discontinuity)
[Warning: Discontinuities detected within algebraic loop(s), may have trouble solving]
[> In build_model (line 4)]
### Starting application: 'Work\slprj\ert\MPC_DBL_Target\pil\MPC_DBL_Target.elf'
[Warning: Connection to box failed, if this is build server then that is ok!]
[> In build_model (line 7)]
[Warning: Cannot close the model 'simHEV' because it has been changed. Use the command 'save_system' to
first save the model]
[Warning: Cannot close the model 'autolibshared' because it has been changed. Use the command 'save_system'
to first save the model]
[Warning: Cannot close the model 'autolibsharedcommon' because it has been changed. Use the command
'save_system' to first save the model]
[Warning: Cannot close the model 'autolibutils' because it has been changed. Use the command 'save_system'
to first save the model]
[Warning: Cannot close the model 'autolibengctrlr' because it has been changed. Use the command
'save_system' to first save the model]
[Warning: Cannot close the model 'autolibshareddrivetraincommon' because it has been changed. Use the
command 'save_system' to first save the model]
[Warning: Cannot close the model 'autolibdrivetraincommon' because it has been changed. Use the command
'save_system' to first save the model]
C:\Users\Administrator\Documents\MATLAB_Jenkins\ISC_Task4_S32G2>exit 0
[AWS CodePipeline Plugin] Publishing artifacts
[AWS CodePipeline Plugin] Compressing directory 'C:\Users\Administrator\Documents\MATLAB_Jenkins' as a 'Zip' archive
[AWS CodePipeline Plugin] Uploading artifact: (Name: Artifact_JenkinsBuild_JenkinsBuild,Location: (Type: S3,S3Location: (BucketName:
nxpmathworksoidstack-nxpmathworkscontainerpipeli-cm281f4b3sl,ObjectKey: nxp-mathworks/Artifact_J/SLwnz28)), file: C:\Users\ADMINI-1\AppData\Local
\Temp\MATLAB_Build-4874275437042705845.zip
[AWS CodePipeline Plugin] Upload successful
[AWS CodePipeline Plugin] Build succeeded, calling PutJobSuccessResult
Finished: SUCCESS
```

PIL SIMULATION

- ✓ Processor in the Loop
- ✓ Simulate with standard drive cycles (e.g. FTP-75, US06, WLTP III)
- ✓ S32G target profiling supported by NXP MBDT

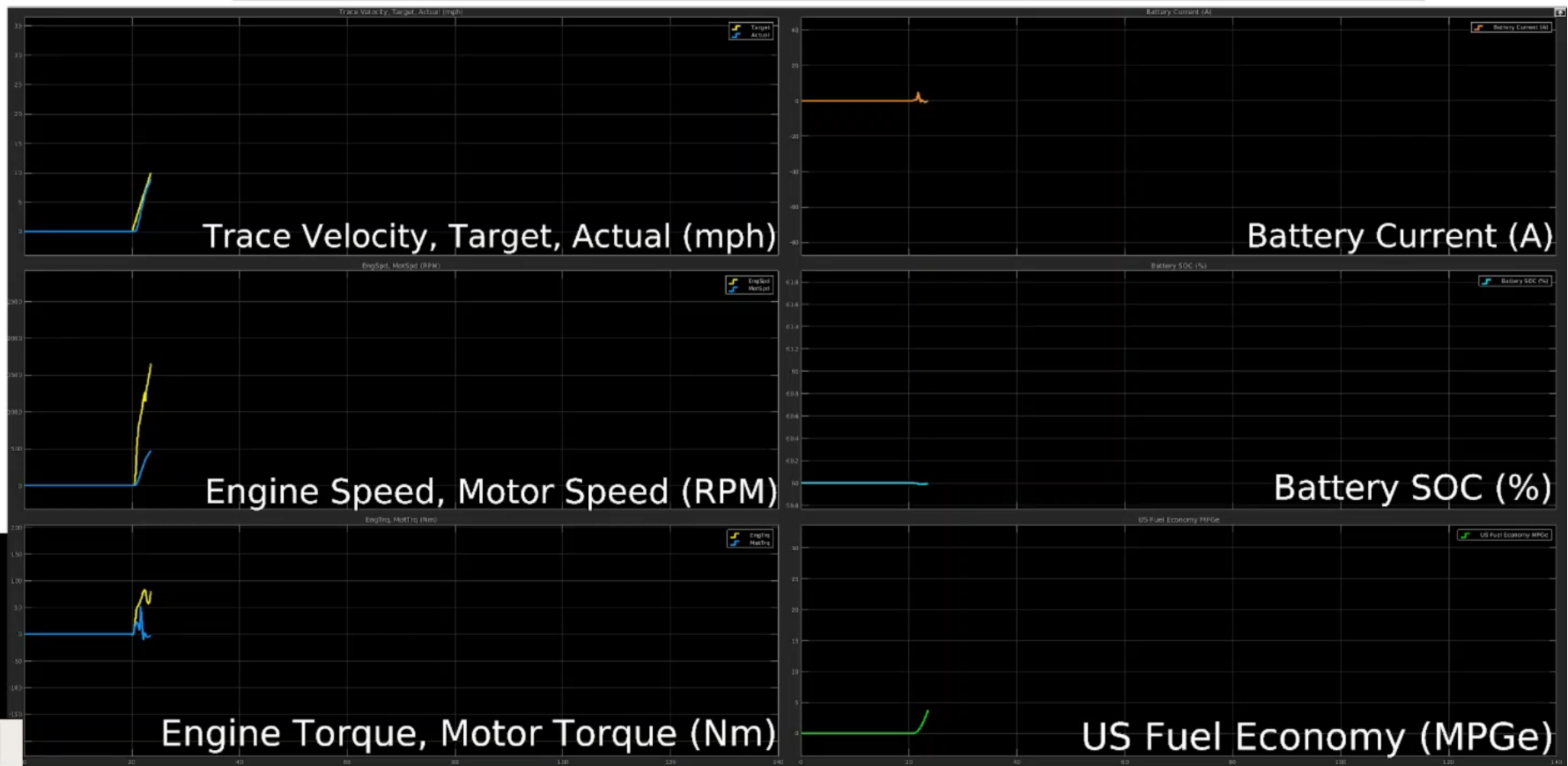




Scenarios Panel (Green = ON)

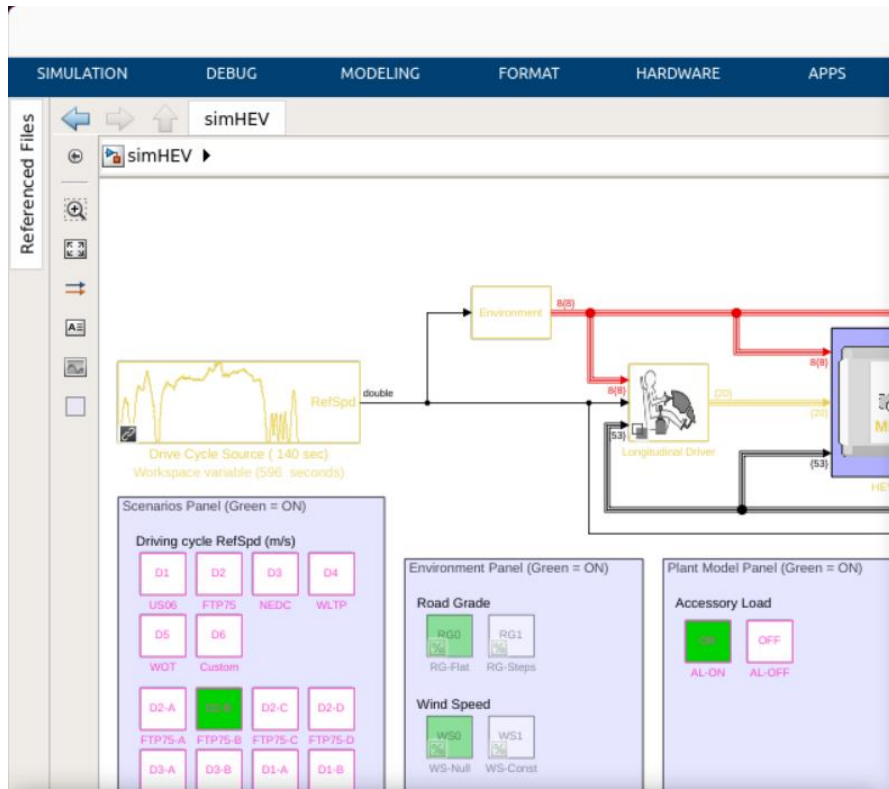
Driving cycle RefSpd (m/s)

D1 US06	D2 FTP75	D3 NEDC	D4 WLTP
D5 WOT	D6 Custom		
D2-A FTP75-A	D2-B FTP75-B	D2-C FTP75-C	D2-D FTP75-D
D3-A NEDC-A	D3-B NEDC-B	D1-A US06-A	D1-B US06-B



T=23.510 16%

PROFILING



Profiling: simHEV/HEV_Controllers/Powertrain Control Module (PCM)/...

Code Execution Profiling

To view execution-time metrics:

- For a profiled component, click a blue-shaded block.
- For top-model tasks, click the blue-shaded background.

Alternatively, you can [view the full code execution profiling report.](#)

simHEV - Simulink

Code Execution Profiling Report

Code Execution Profiling Report for simHEV/HEV_Controllers/Powertrain Control Module (PCM)/Hybrid Control Module (HCM)/Hybrid Control Module (HCM) - Optimal/HEV Energy Management and Battery Thermal Management/MPC-DBL

The code execution profiling report provides metrics based on data collected from a SIL or PIL execution. Execution times are calculated from data recorded by instrumentation probes added to the SIL or PIL test harness or inside the code generated for each component. See [Code Execution Profiling](#) for more information.

1. Summary

Total time	15083965022
Unit of time	Ticks
Command	report(executionProfile, 'Units', 'Ticks');
Timer frequency (ticks per second)	unavailable
Profiling data created	22-Mar-2022 18:35:31

2. Profiled Sections of Code

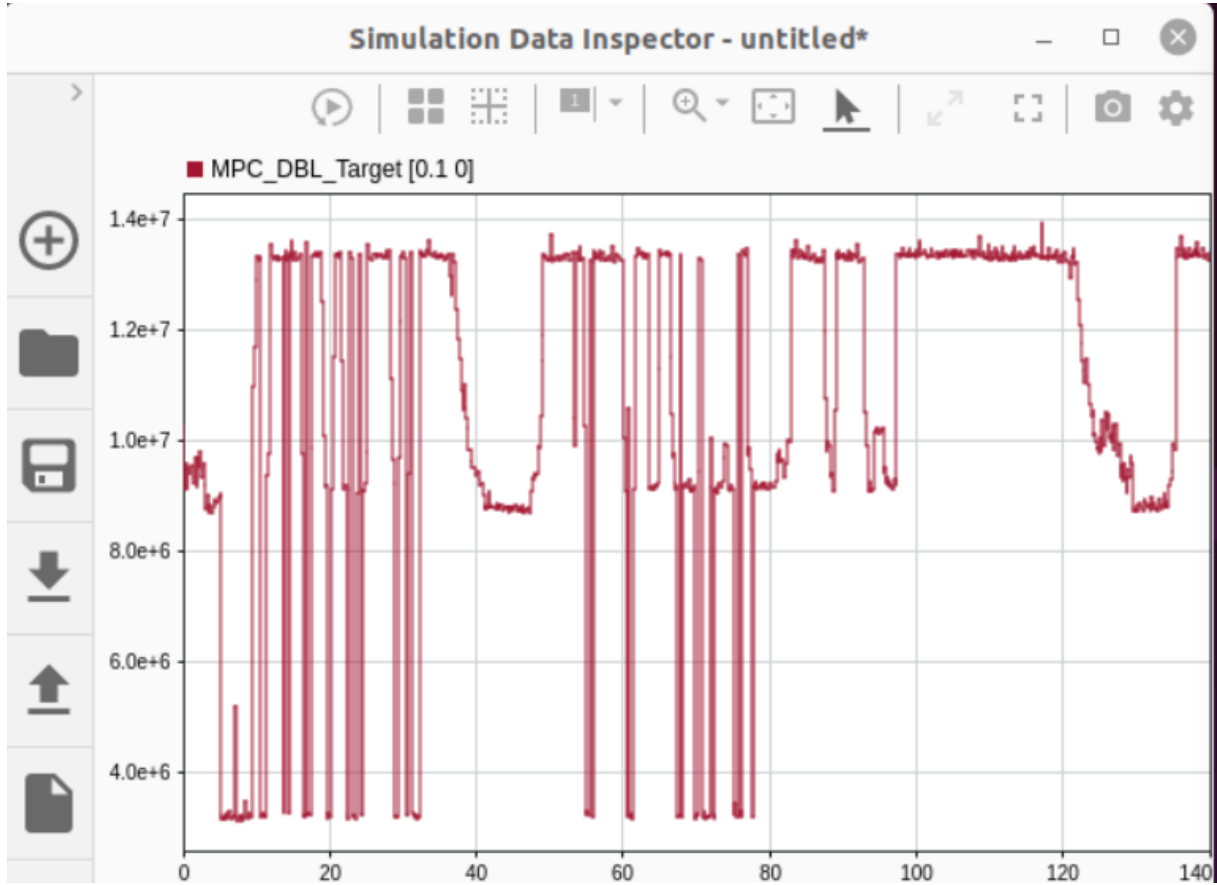
Section	Maximum Execution Time in ticks	Average Execution Time in ticks	Maximum Self Time in ticks	Average Self Time in ticks	Calls
MPC_DBL_Target_initialize	2200	2200	2200	2200	1
MPC_DBL_Target_Init	297611	297611	297611	297611	1
[+] MPC_DBL_Target [0.1 0]	13944124	10766356	2024476	1612268	1401
MPC_DBL_Target_Term	1000	1000	1000	1000	1

3. Definitions

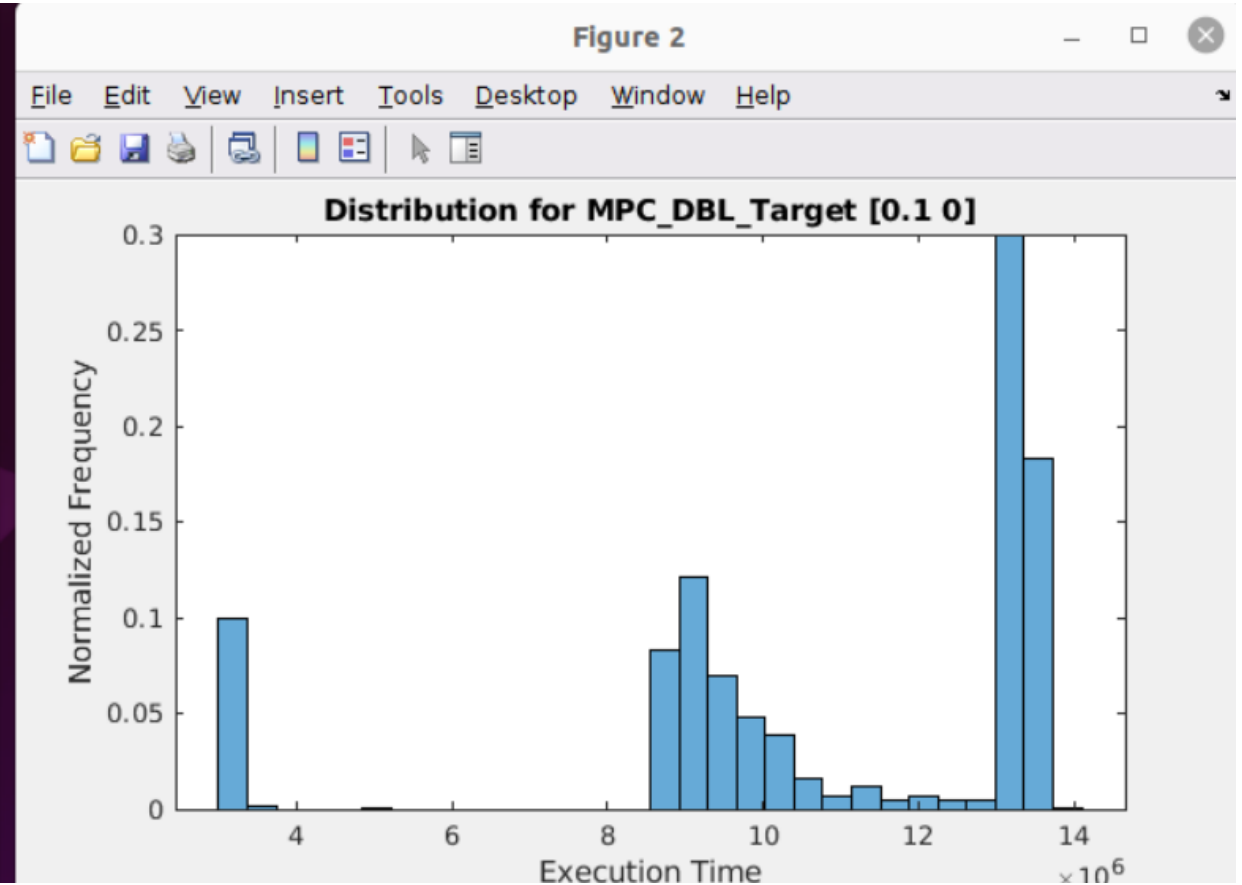
Execution Time: Time between start and end of code section.

OK Help

PROFILING RESULTS: TIME SERIES AND HISTOGRAM



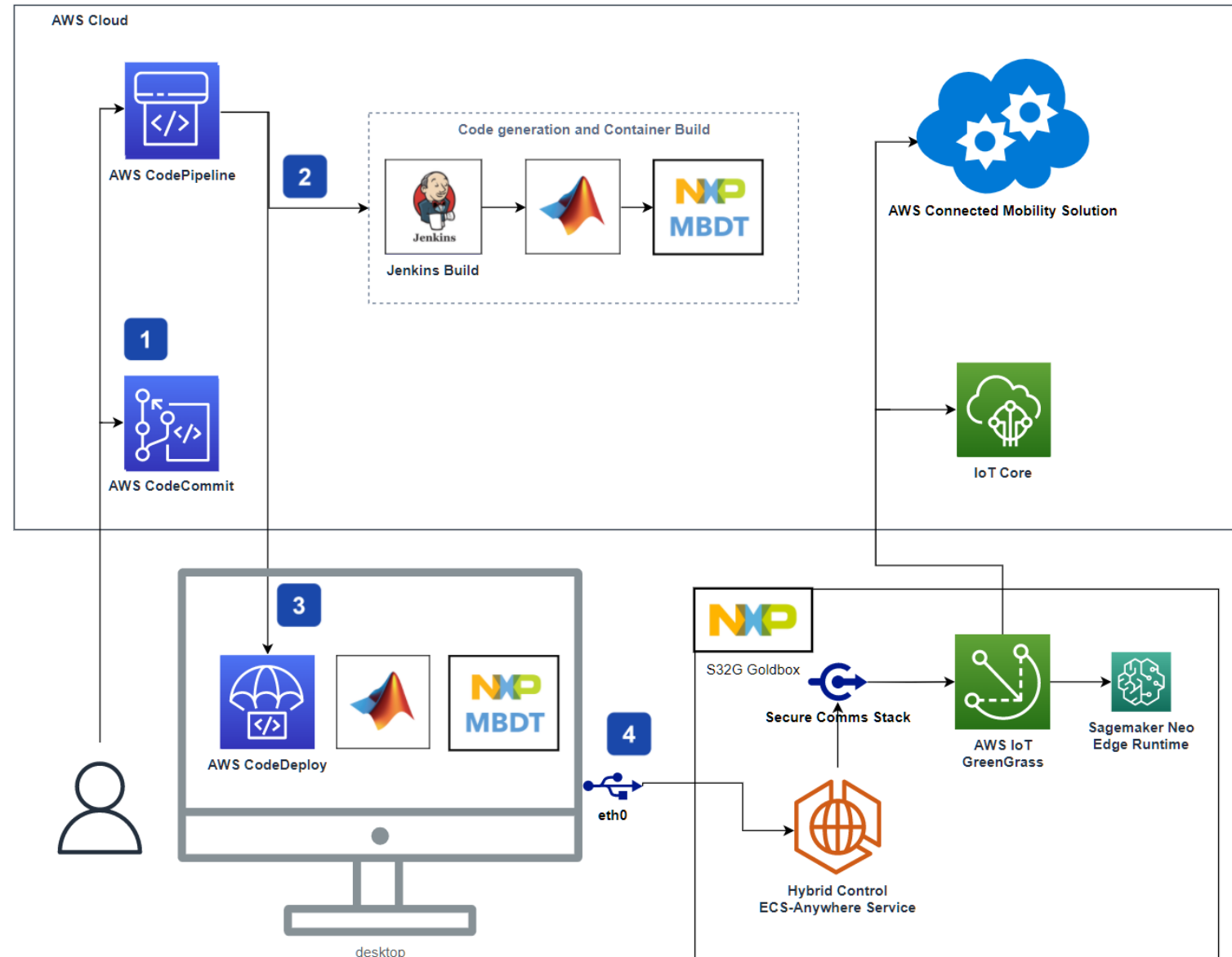
Time Series



Histogram

RECAP: AUTOMOTIVE DEVOPS MODEL-BASED DESIGN EXAMPLE

- 1 Code Commit:** track changes for global user base
- 2 Code Build:** leverage installed tools for multiple users; integrated with MathWorks and NXP model-based design tools
- 3 Deploy:** push compiled code to HIL / PIL systems for testing
- 4 Simulate:** PIL execution with real-time profiling on Automotive processors



AUTOMOTIVE DEVOPS MODEL-BASED DESIGN



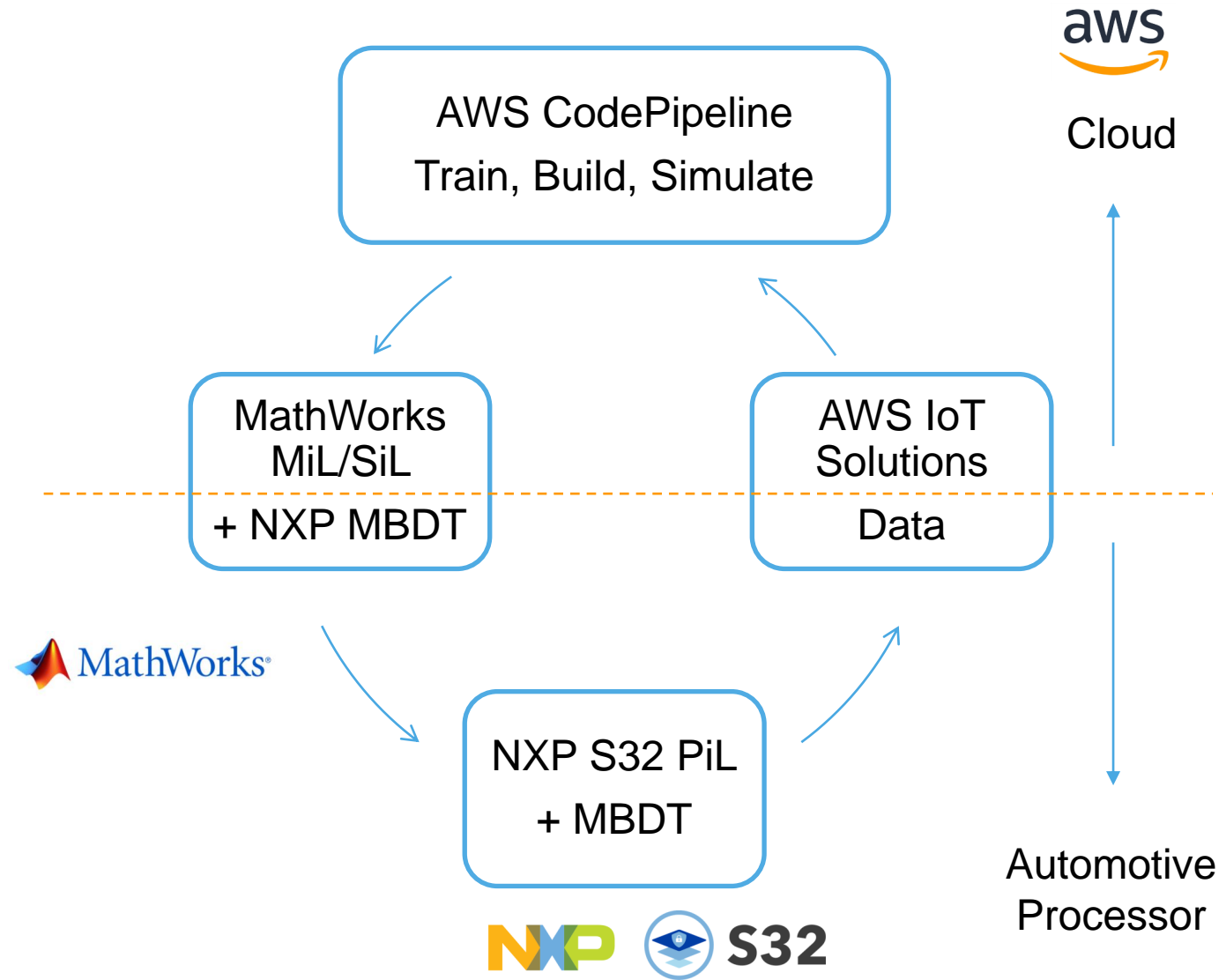
Design, build and simulate in the cloud. Engineers use model-based systems engineering (MBSE) to manage system complexity, improve communication, and produce optimized systems



Deploy to the Automotive Edge. NXP's S32G Vehicle Network Processors interface with all the vehicle functional domains and provide secure processing (AI/ML) and network acceleration for vehicle edge services.



Integration with AWS CodePipeline and AWS IoT Greengrass enables a DevOps workflow built on AWS.



Conclusion



SECURE CONNECTIONS
FOR A SMARTER WORLD



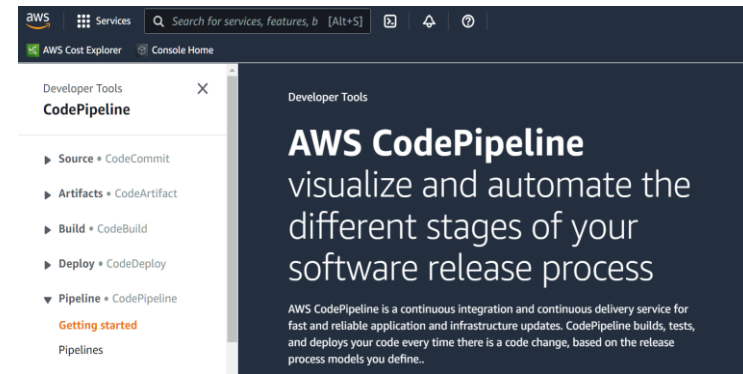
PUBLIC

NXP, THE NXP LOGO AND NXP SECURE CONNECTIONS FOR A SMARTER WORLD ARE TRADEMARKS OF NXP B.V.
ALL OTHER PRODUCT OR SERVICE NAMES ARE THE PROPERTY OF THEIR RESPECTIVE OWNERS. © 2022 NXP B.V.



SUMMARY AND SOLUTION BENEFITS

- Develop in the cloud and deploy to edge
- To meet the automotive trends, we highlighted:
 - transition to a Continuous Integration / Continuous Deployment workflow
→ [AWS CodePipeline](#)
 - increased size of software development teams
→ [cloud migration using AWS solutions](#)
 - migration of tools and workflows to the cloud
→ [AWS hosting MathWorks and NXP MBDT](#)
 - continued adoption of model-based design engineering
→ [supported by MathWorks toolchains and NXP MBDT](#)
- Edge deployment using NXP MBDT and Automotive Real-Time Processors and Vehicle Network Processors:
→ [real world benchmarking on Automotive targets](#)



FOR MORE INFORMATION

- **NXP:**

GoldBox	GreenBox	MBDT
		
www.nxp.com/GoldBox	www.nxp.com/GreenBox	www.nxp.com/MBDT

- [Connected EV Management Demo](#)

- **AWS:**

- [AWS CodePipeline | Continuous Integration & Continuous Delivery \(amazon.com\)](#)
- [Intelligence at the IoT Edge — AWS IoT Greengrass — Amazon Web Services](#)

- **MathWorks:**

- <https://www.mathworks.com/solutions/automotive/virtual-vehicle.html>

- For follow up: curt.hillier@nxp.com



SECURE CONNECTIONS
FOR A SMARTER WORLD