MATLAB EXPO 2017

Generating Optimized Code for Embedded Microcontroller Algorithms

Gaurav Dubey

Senior Team Lead, Pilot Engineering

Gaurav.Dubey@mathworks.in

Key Takeaways

1. Reduce costs by minimizing hardware resources
2. Create innovative products by maximizing algorithm content
3. Expand code generation use to more applications (e.g., 8-16 bit)

*"Embedded Coder generates **optimized code** that is as good as we can write, and we've never had any problems with defects in the generated code."*
Dr. Robert Turner, ABB

ABB Accelerates the Delivery of **Large-Scale, Grid-Connected Inverter** Products with Model-Based Design

Challenges

- Difficult to fit modern algorithms into low-cost production hardware
  - Limited ROM, RAM, stack, and speed
- Not known *a priori* during design, what embedded device is required
  - Need optimal implementation
- Hand coding is process bottleneck
  - Adds bugs, delays, iterations

*"The **advantages of Model-Based Design** over hand-coding in C can't be overestimated."   Kazuhiro Ichikawa, Ono Sokki*

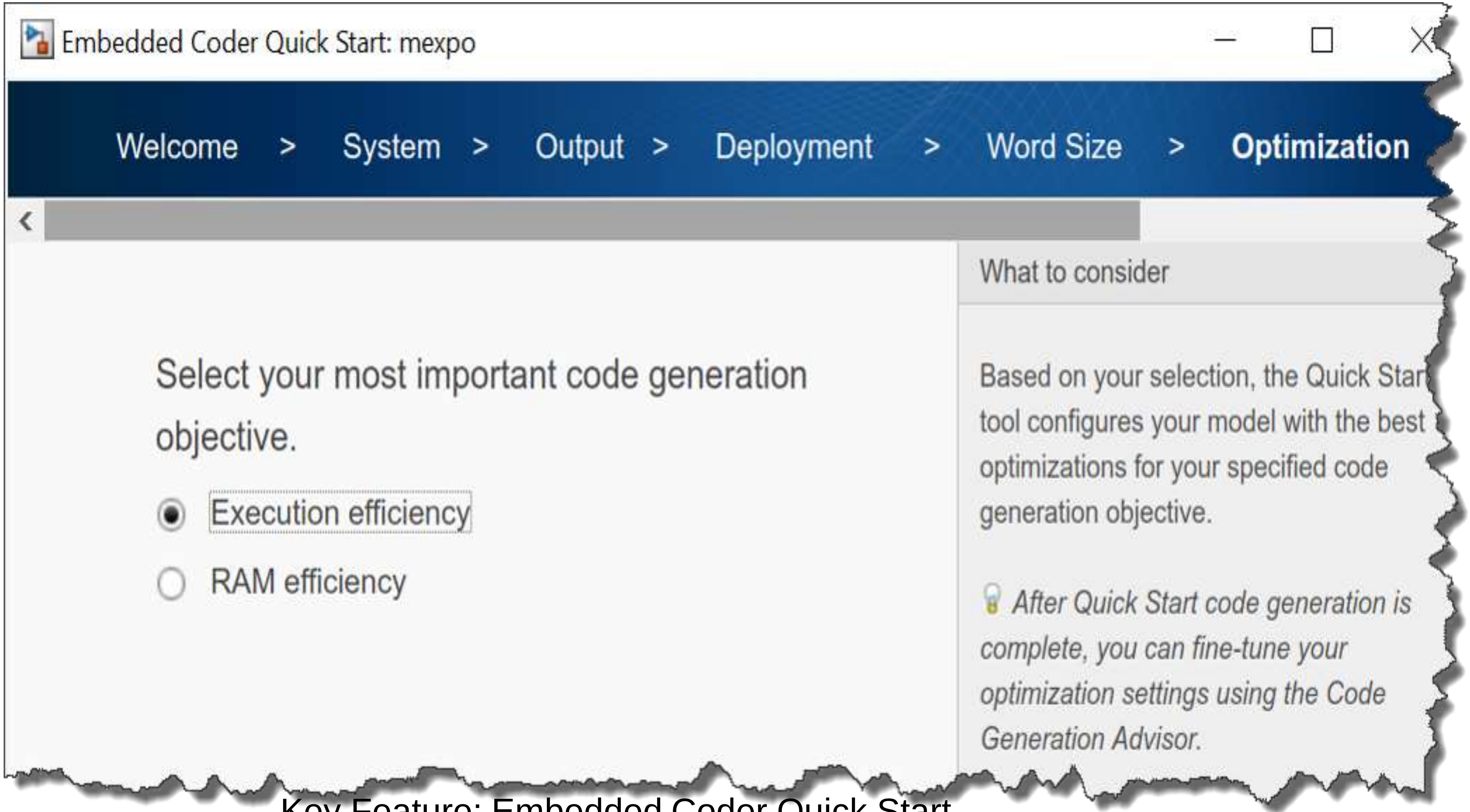
Ono Sokki Reduces Development Time for **Precision Automotive Speed** Measurement Device

Solutions

Optimization Techniques:

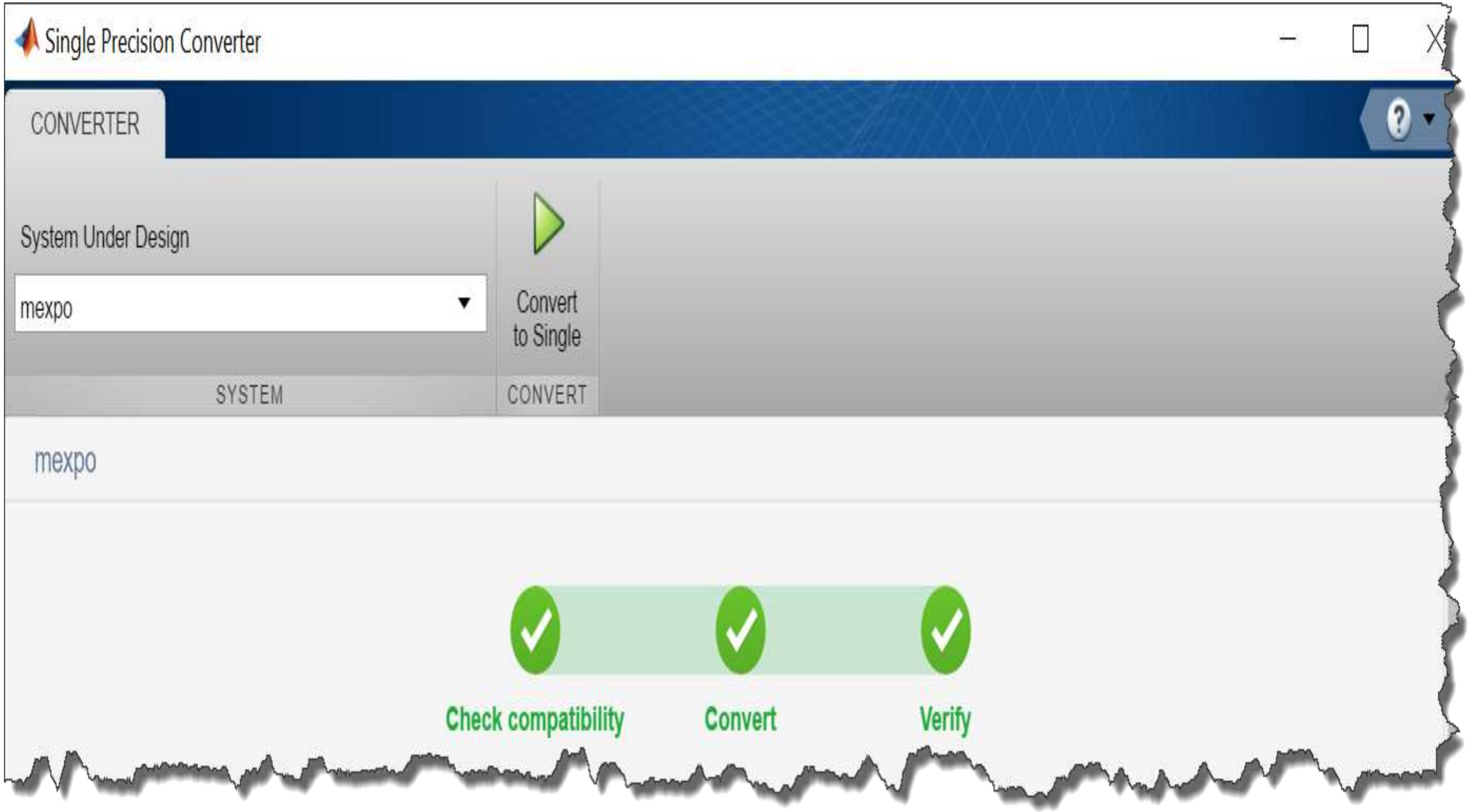
1. Use optimal settings
2. Minimize data sizes
3. Target vector engines
4. Select best processor(s)
5. Reduce data copies
6. Optimize Using Min & Max Values
7. Reuse components
8. Identifying clones in model

1. Use Optimal Settings



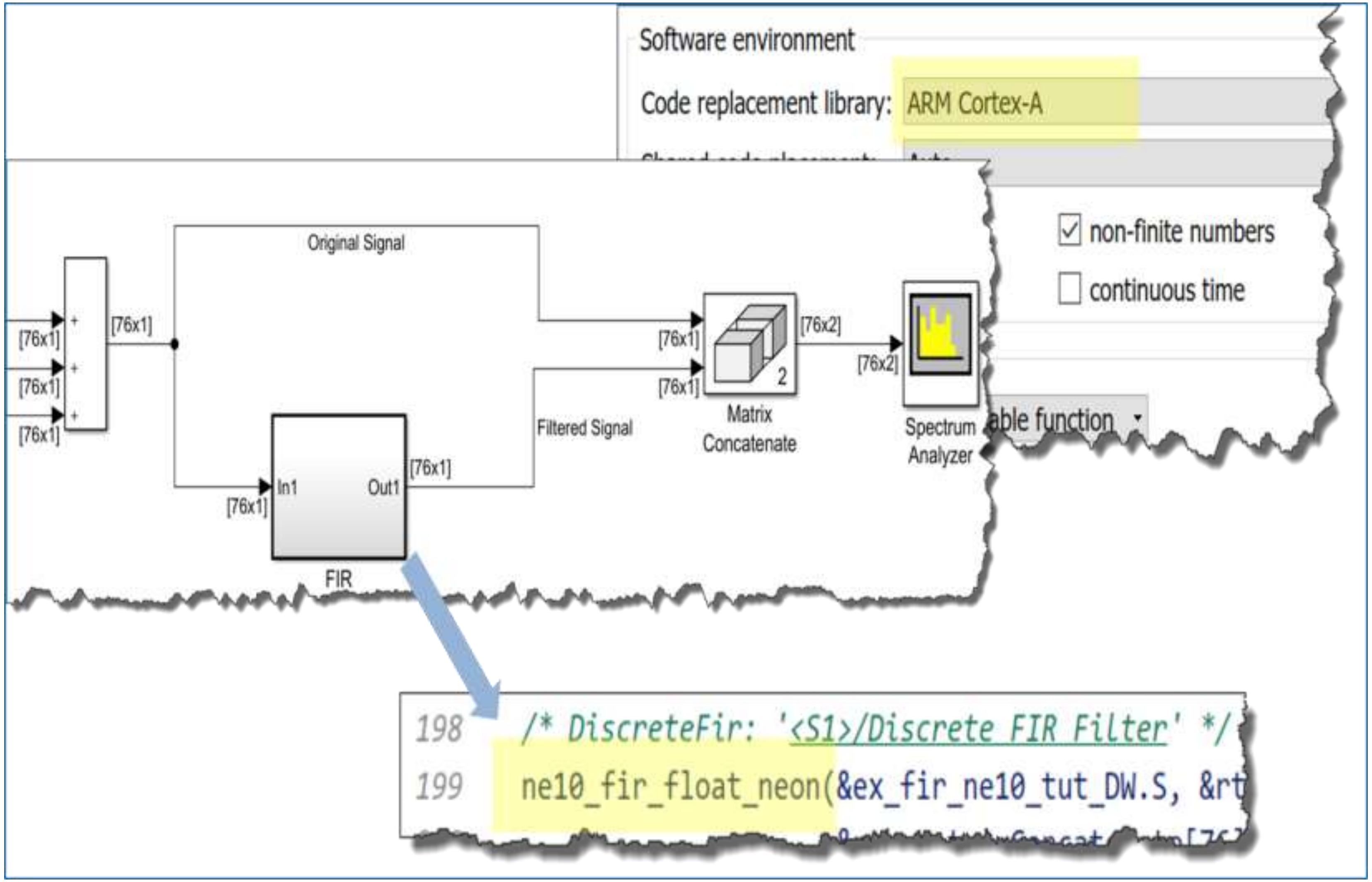
Key Feature: Embedded Coder Quick Start

2. Optimize Data Types



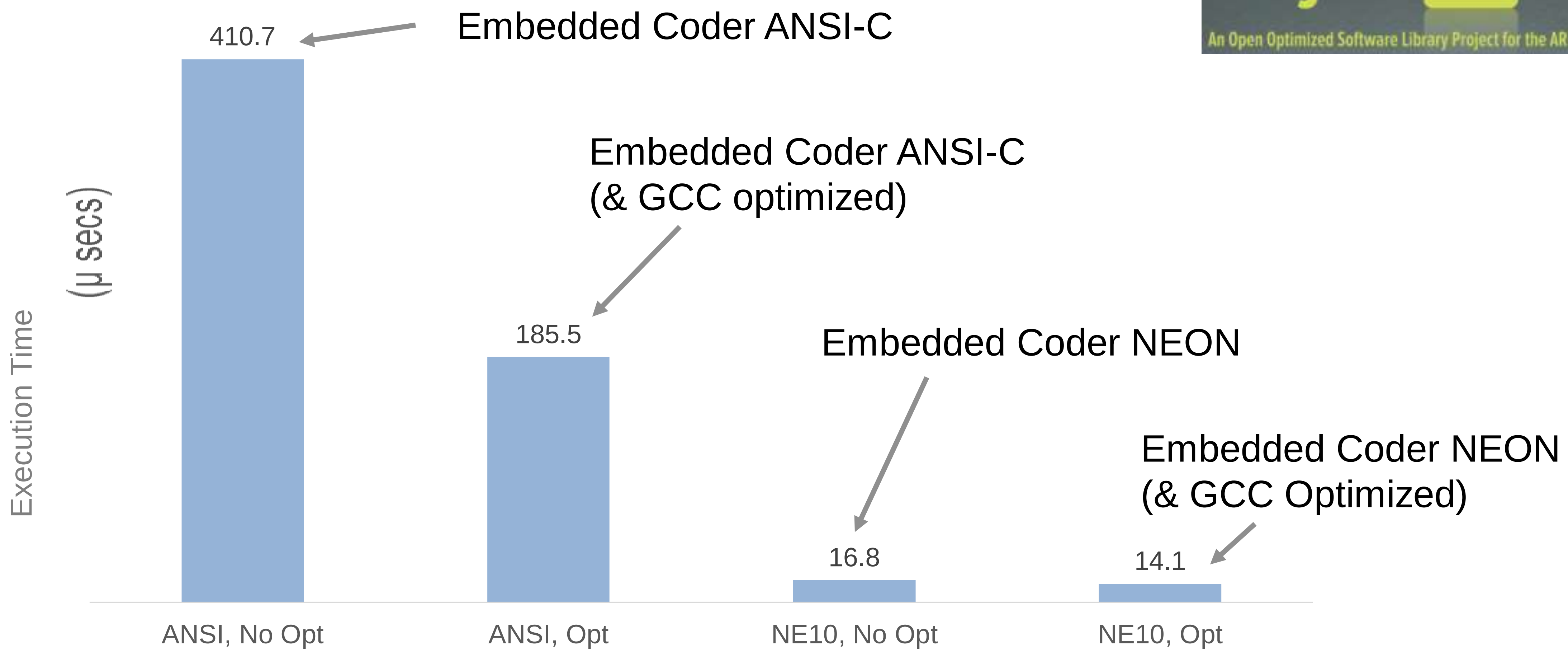
Key Feature: Single Precision Converter

3. Target vector engines



Key Feature: Code Replacements

PIL Benchmark Results for ARM Cortex-A



Run Format: [ANSI or Ne10], [gcc no opt or gcc -02], ARM 1Ghz Cortex A8

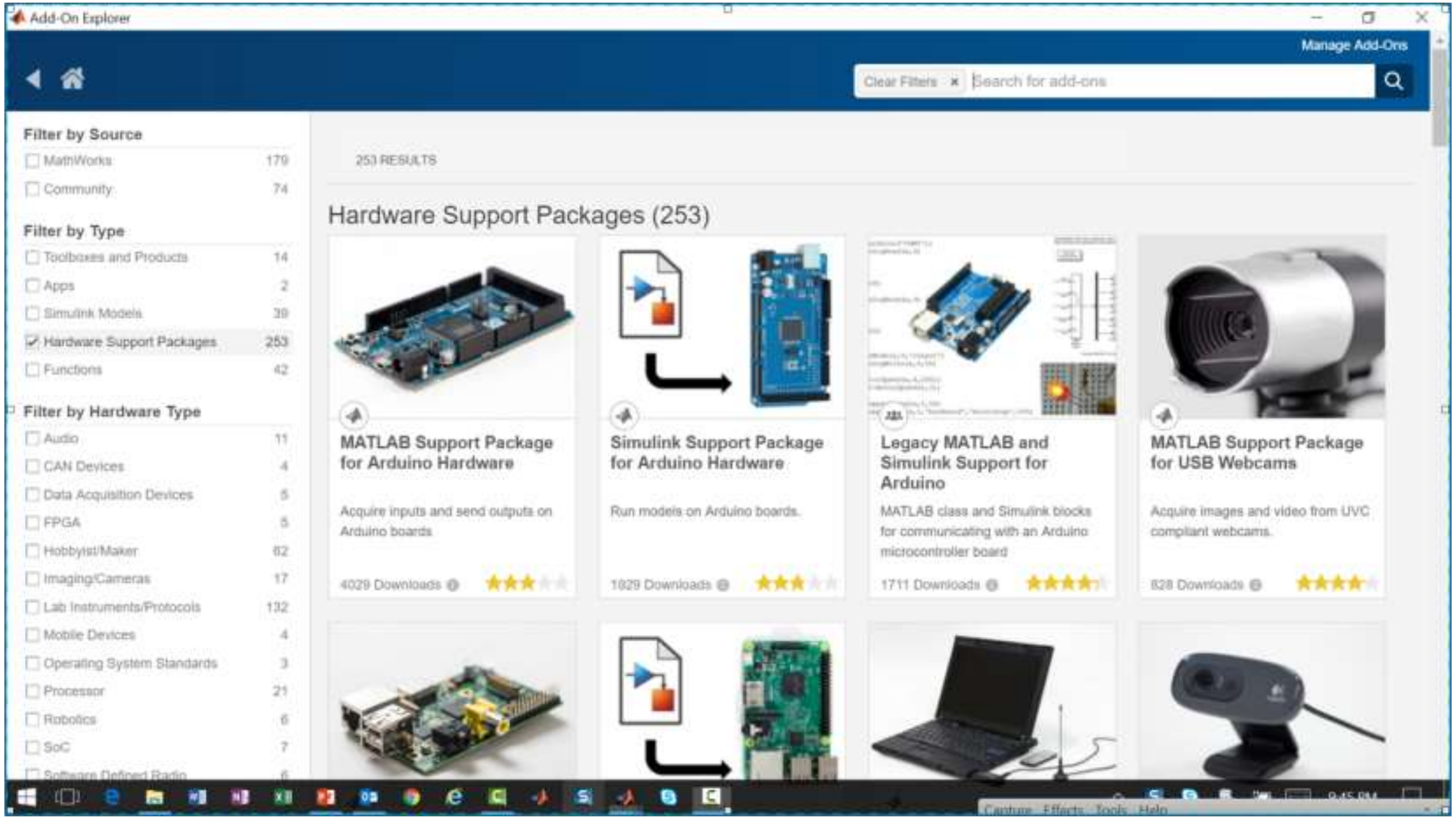
Example: FIR Filter

4. Select best processor(s) for your application

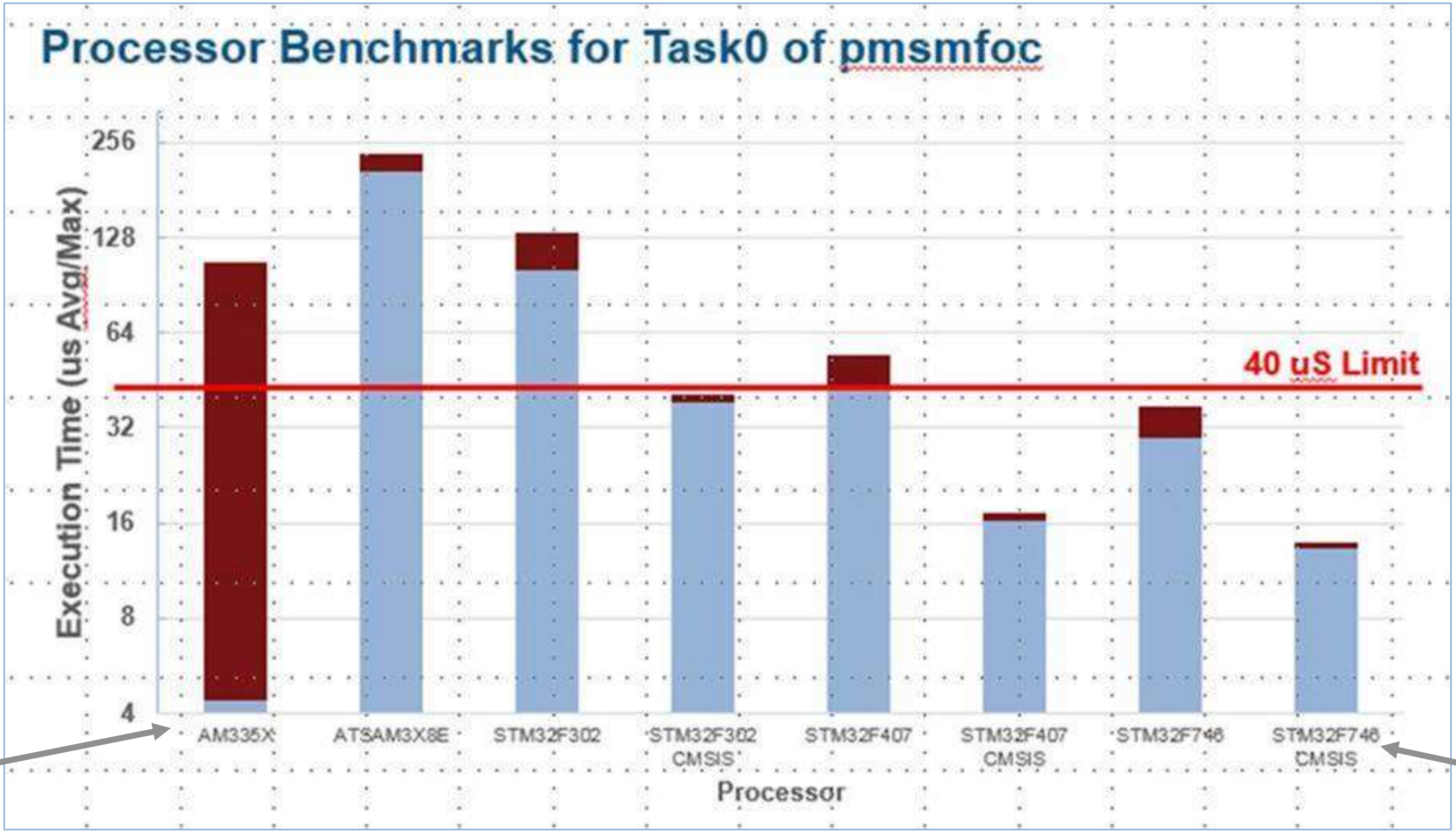
- Portable code: any device for **algorithm code generation**

- Support packages for **target-specific** system executable generation
  - ARM … Zynq

- Hardware vendors offer their **own target packages**
  - ADI, Infineon, Microchip, NXP, Renesas, TI, STMicroelectronics



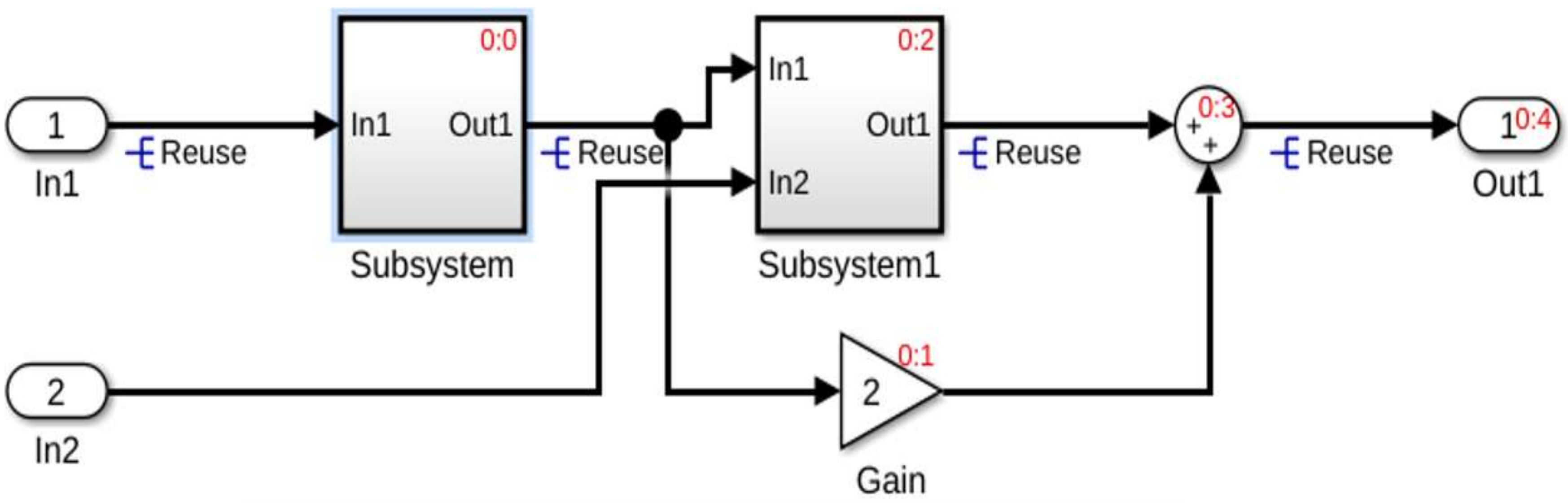
Results for PMSM Motor Control for ARM cores
- Average and Max Execution Time



Cortex-A8,
1 GHz,
Linux OS,
NE10 DSP Libs

Cortex-M7,
216 MHz,
Bare metal,
CMSIS" DSP Libs

5. Reuse data



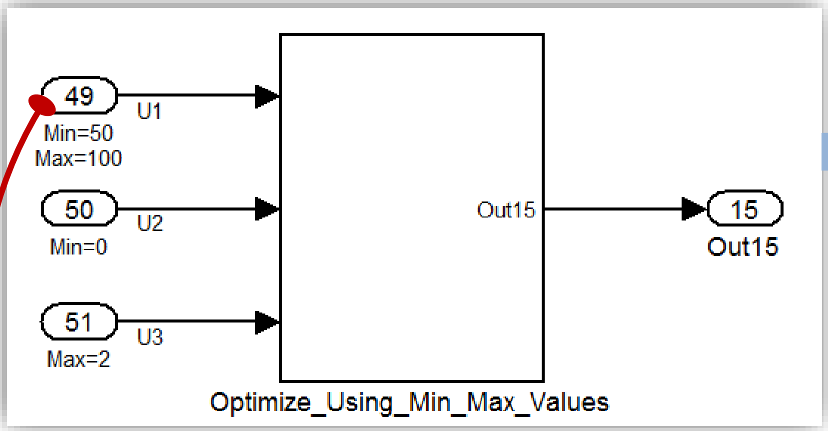

```
void Subsystem(void)
{   ...
    for (i = 0; i < 100; i++) {
        Reuse[i] = 2.0F * Reuse[i] * 20.0F;
    }
    fcn3(&Reuse[0]);
}
```

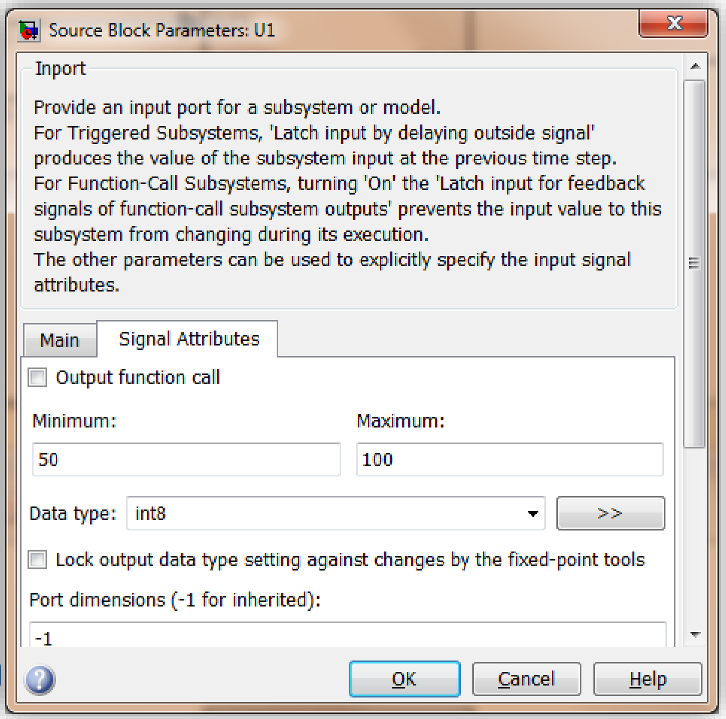
Key Feature: Reusable Storage Classes

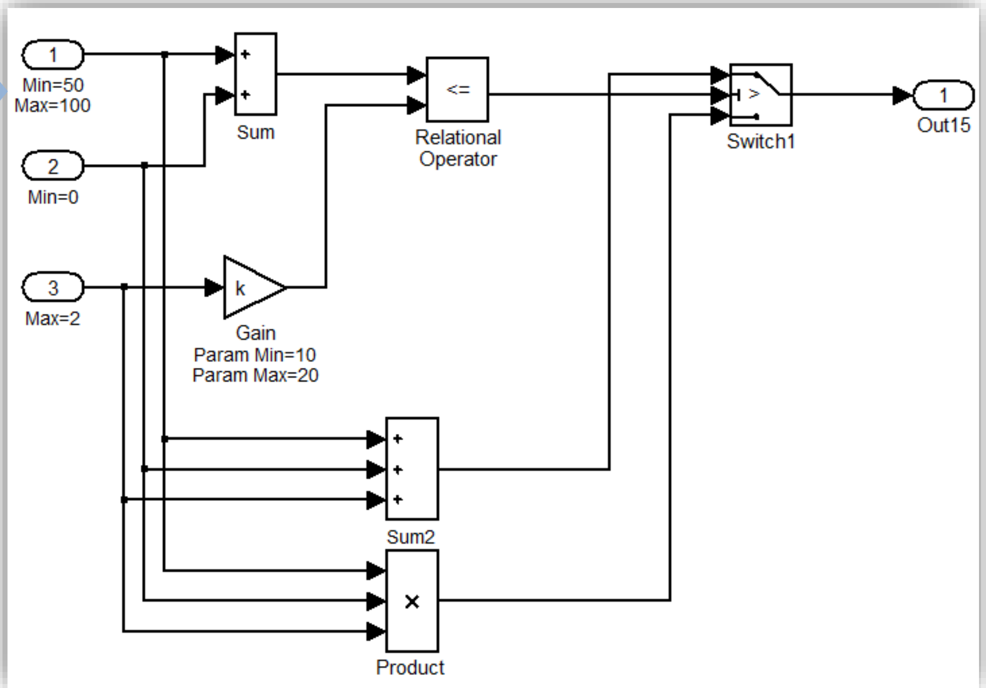
6. Optimize Using Min & Max Values

- These minimum and maximum values usually represent environmental limits, such as temperature, or mechanical and electrical limits, such as output ranges of sensors.
- Software uses the minimum and maximum values to derive range information for downstream signals in the model.
- This derived range information is used to determine if it is possible to streamline the generated code by, for example:
  - Reducing expressions to constants
  - Removing dead branches of conditional statements
  - Eliminating unnecessary mathematical operations
- This optimization results in:
  - Reduced ROM and RAM consumption
  - Improved execution speed

Configure Model







6. Optimize Using Min & Max Values

Code generation

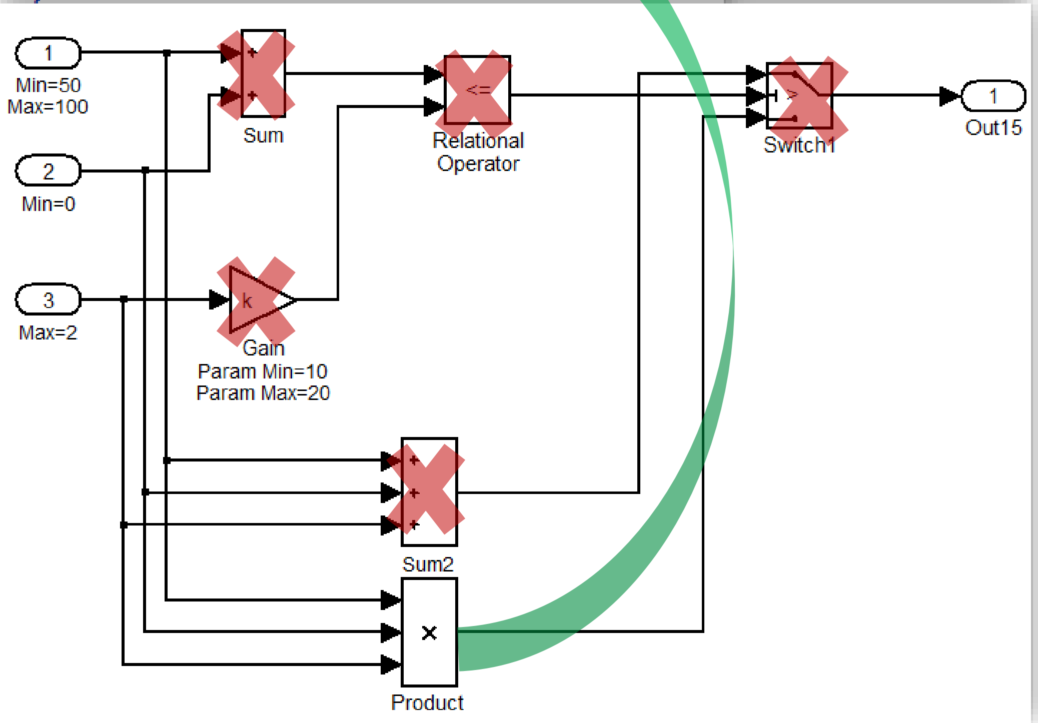
☐ Optimize using the specified minimum and maximum values

```
rtb_Sum = U1 + U2;

/* Gain: '<S8>/Gain' incorporates:
 *  Inport: '<Root>/U3'
 */
rtb_Sum2 = Code_Optimization_P.Gain_Gain * U3;

/* RelationalOperator: '<S8>/Relational Operator' */
rtb_RelationalOperator = (rtb_Sum <= rtb_Sum2);

/* Switch: '<S8>/Switch1' */
if (rtb_RelationalOperator) {
  /* Sum: '<S8>/Sum2' incorporates:
   *  Inport: '<Root>/U1'
   *  Inport: '<Root>/U2'
   *  Inport: '<Root>/U3'
   */
  rtb_Sum2 = (U1 + U2) + U3;
} else {
  /* Product: '<S8>/Product' incorporates:
   *  Inport: '<Root>/U1'
   *  Inport: '<Root>/U2'
   *  Inport: '<Root>/U3'
   */
  rtb_Sum2 = U1 * U2 * U3;
}
```

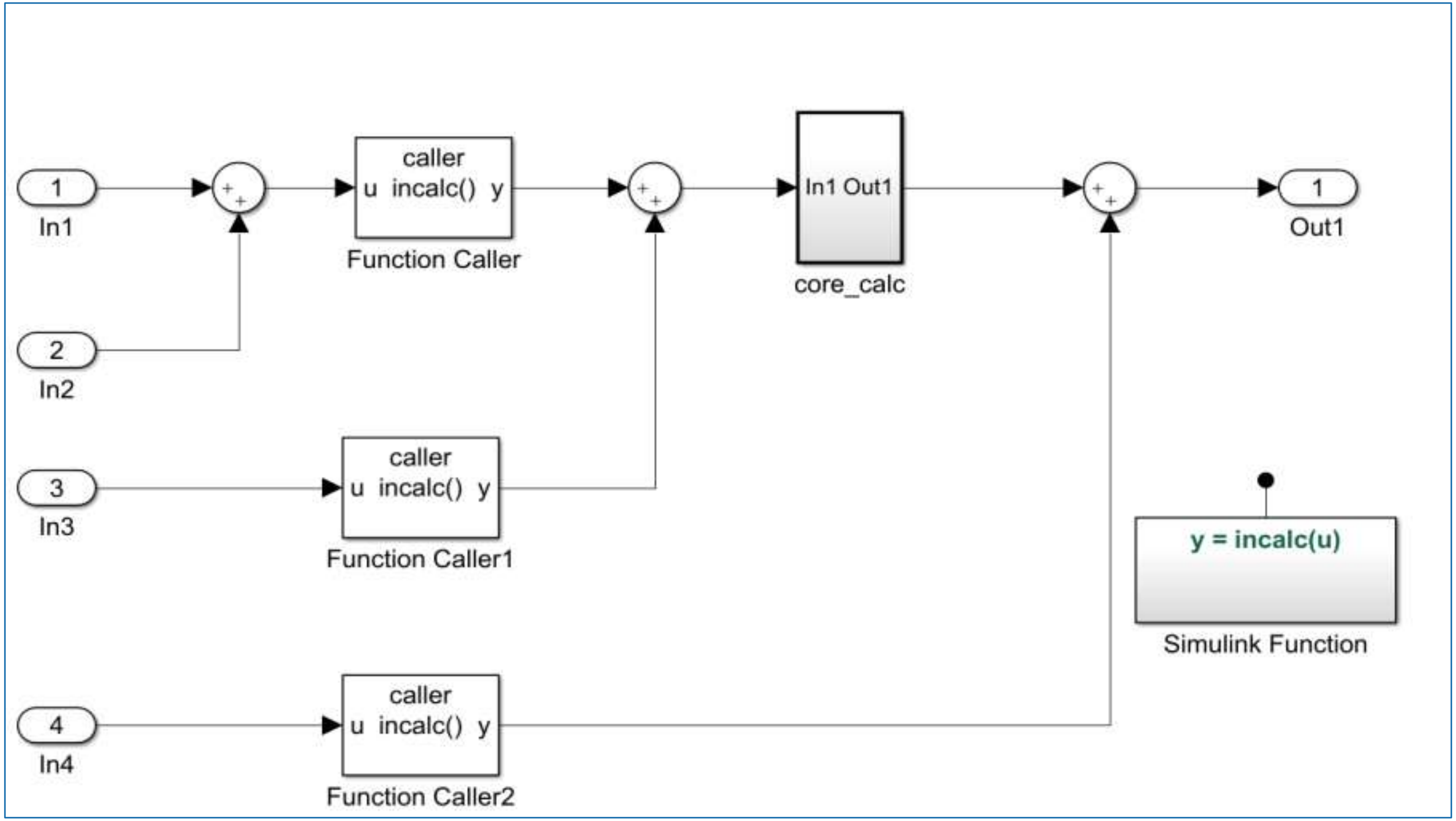
Code generation

☑ Optimize using the specified minimum and maximum values

```
/* Product: '<S8>/Product' incorporates:
 *  Inport: '<Root>/U1'
 *  Inport: '<Root>/U2'
 *  Inport: '<Root>/U3'
 *  Switch: '<S8>/Switch1'
 */
rtb_Sum2 = U1 * U2 * U3;

/* Outport: '<Root>/Out15' */
Code_Optimization_Y.Out15 = rtb_Sum2;
```

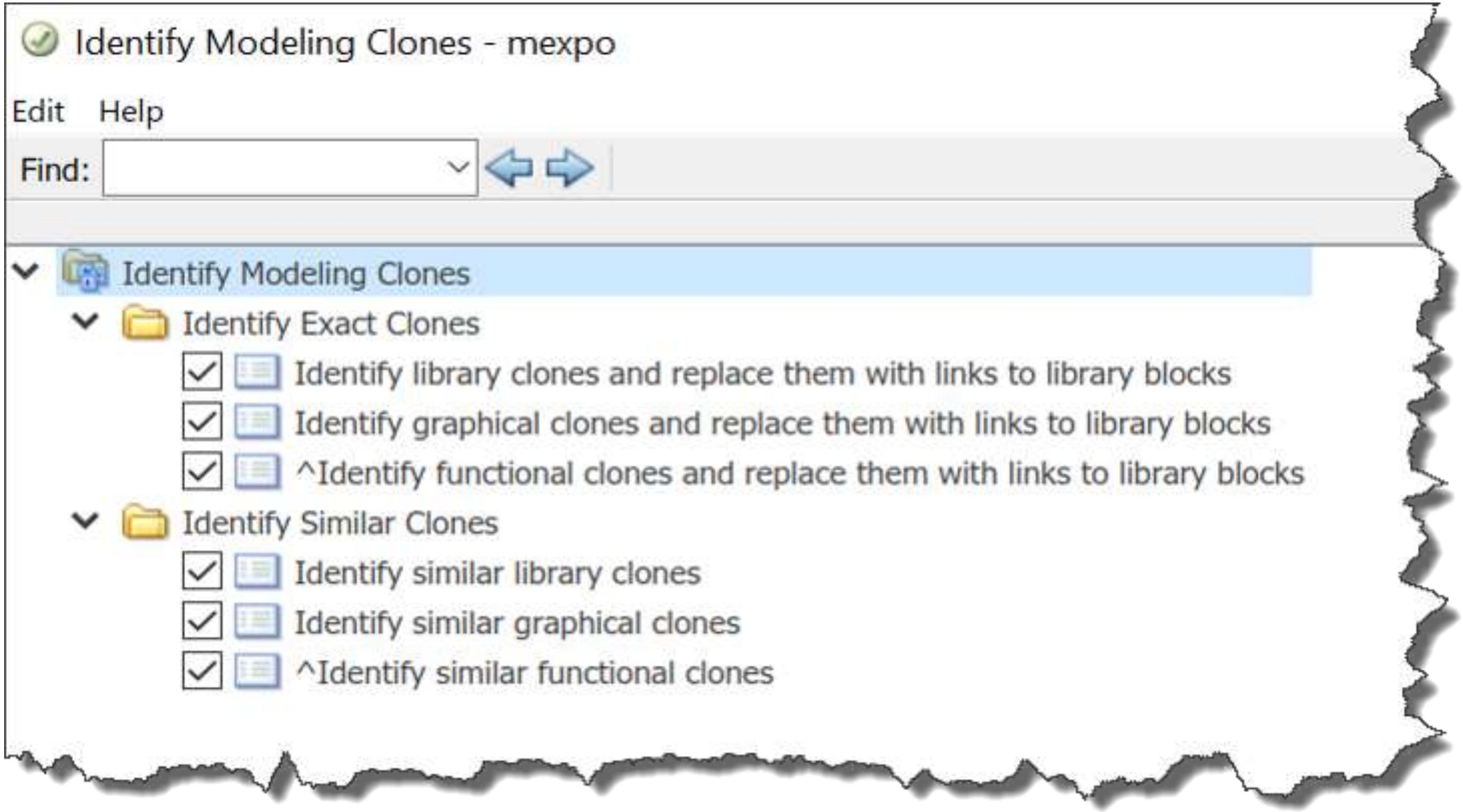


7. Reuse components



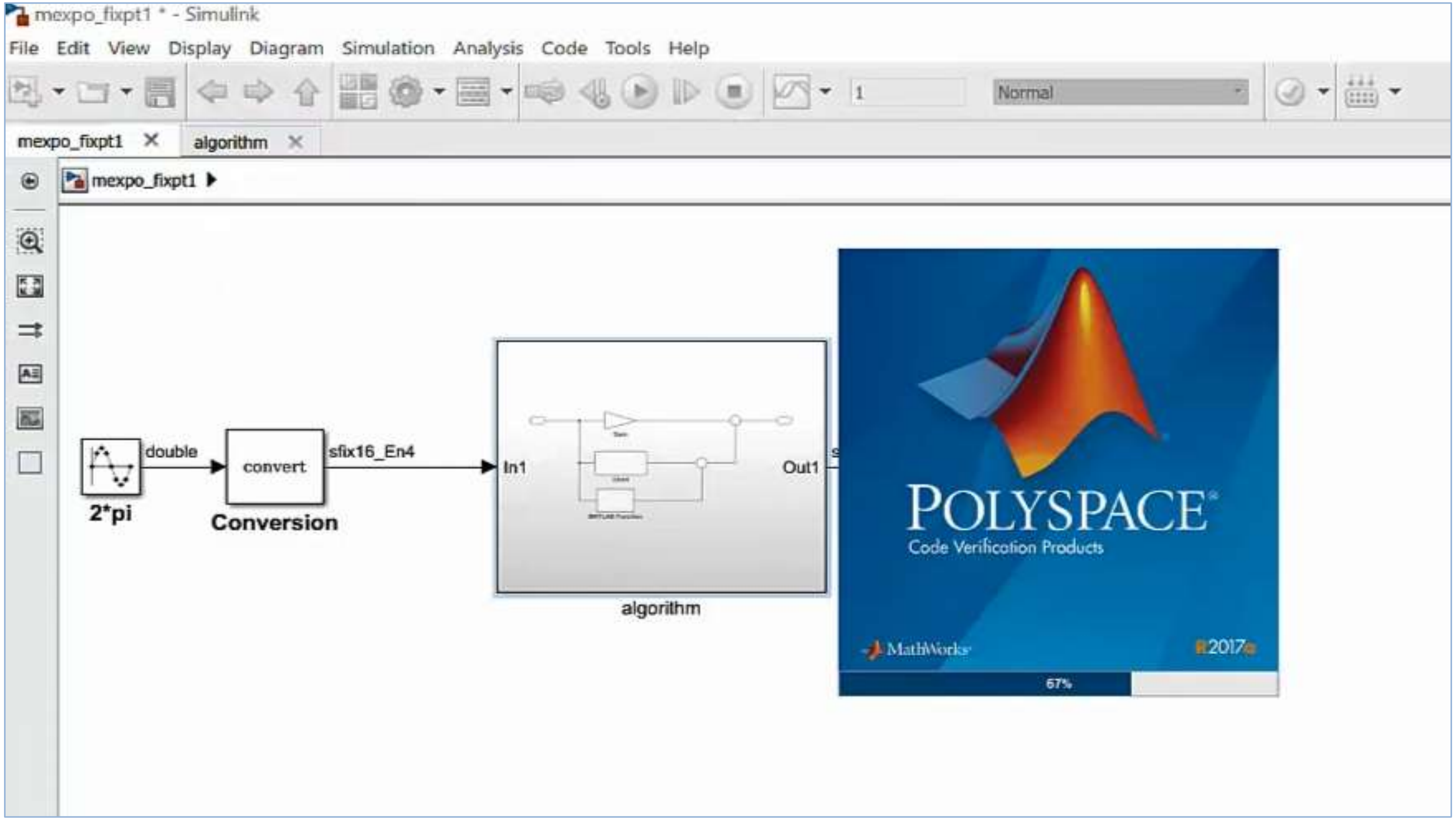
Key Features: Subsystem Reuse and Simulink Functions

8. Detecting Clones in model



Key Feature: Simulink Clone Detection

8. Thrift Logic (Prove)



Key Feature: Polyspace Code Prover

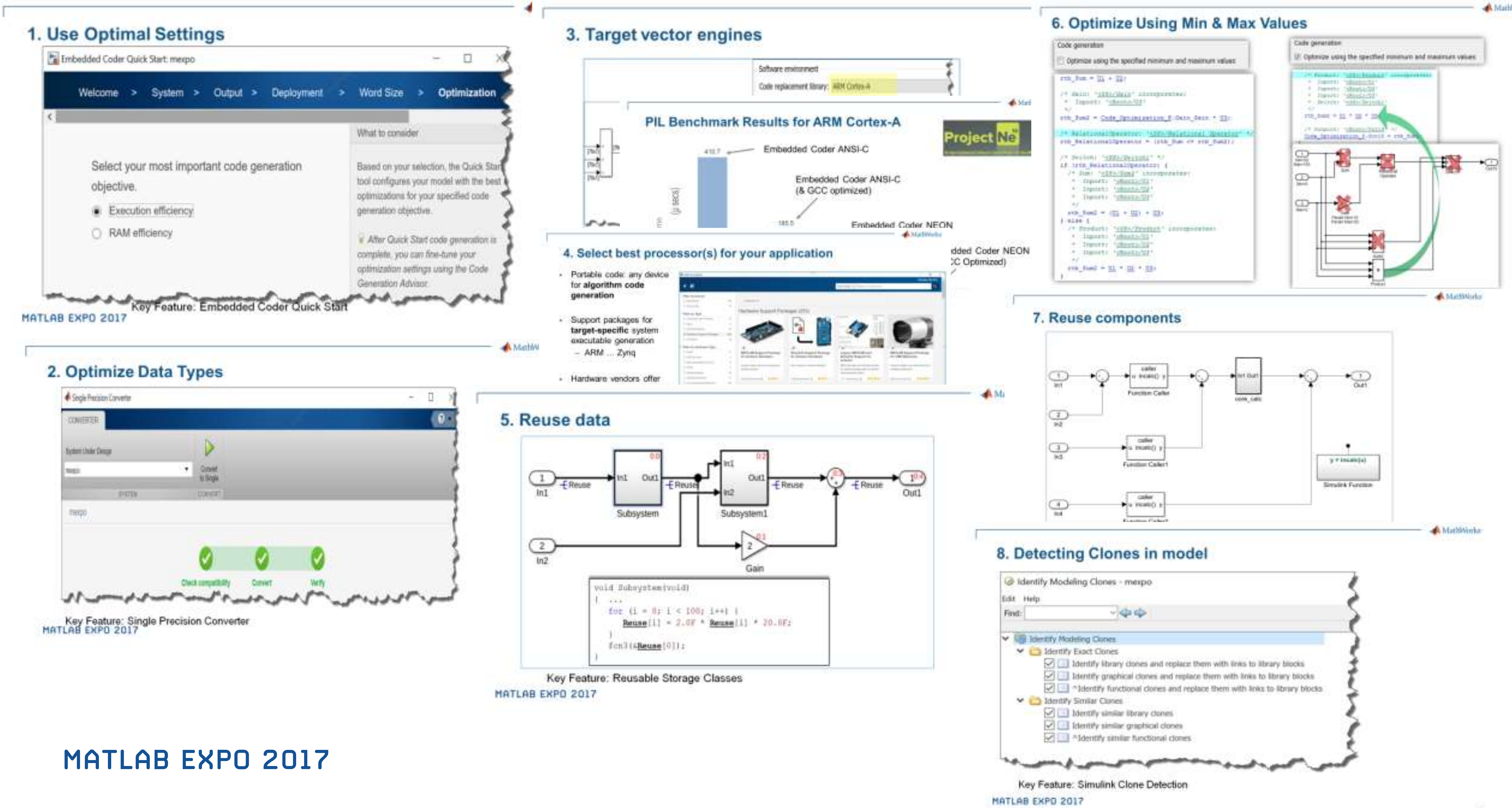
Solution Summary

Optimization Techniques:

1. Use optimal settings
2. Minimize data sizes
3. Target vector engines
4. Select best processor(s)
5. Reduce data copies
6. Reuse components
7. Thrift logic

*"The code generated with Embedded Coder required about **16% less RAM** than the handwritten code used on a previous version of the ECU; the code met all project requirements for efficiency and structure." Mario Wünsche, Daimler*

[Daimler Designs Cruise Controller for Mercedes-Benz Trucks]



MATLAB EXPO 2017

Key Takeaways

Simulink and Embedded Coder new optimizations let you:

1. Reduce costs by minimizing hardware resources
2. Create innovative products by maximizing algorithm content
3. Expand code generation use to more applications (e.g., Mitsuba Uses Embedded Coder for **NEC 78K 8-bit** microcontroller)

"When we generated code with Embedded Coder, the team we handed it off to knew it ***was gold****" Maria Radecki, BAE Systems*

BAE Systems Delivers **DO-178B Level A** Flight Software on Schedule with Model-Based Design

Additional Customer References and Production Applications

Honeywell Aerospace, USA
Certified Flight Control Processor

FLIR Systems, USA and Sweden
Thermal Imaging FPGA

Festo AG, Germany
Robotic PLC

GM, USA
Powertrain ECU

Alstom Grid, UK
HDVC Power DSP

Baker Hughes, Germany
Oil and Gas Drill Processor

www.mathworks.com/company/user_stories/

Training Services

Exploit the full potential of MathWorks products

Flexible delivery options:

- Public training available in several cities
- Onsite training with standard or customized courses
- Web-based training with live, interactive instructor-led courses

More than 48 course offerings:

- Introductory and intermediate training on MATLAB, Simulink, Stateflow, code generation, and Polyspace products
- Specialized courses in control design, signal processing, parallel computing, code generation, communications, financial analysis, and other areas

www.mathworks.in/training

Generating Optimized Code for Embedded Microcontroller Algorithms

- Testing Generated Code in Simulink
  - This one-day course provides a working introduction to designing and testing embedded applications with Simulink Coder™ and Embedded Coder. Themes of simulation speedup, parameter tuning in the deployed application, structure of embedded code, code verification, and execution profiling are explored in the context of Model-Based Design

- Embedded Coder for Production Code Generation
  - This three-day course focuses on developing models in the Simulink environment to deploy on embedded systems. The course is designed for Simulink users who intend to generate, validate, and deploy embedded code using Embedded Coder

MathWorks®

Accelerating the pace of engineering and science

Speaker Details

Email: Gaurav.Dubey@mathworks.in

LinkedIn: https://www.linkedin.com/in/gauravdubey4

Call: **080-6632-6053**

Contact MathWorks India

Products/Training Enquiry Booth

Call: 080-6632-6000

Email: info@mathworks.in

Your feedback is valued.

Please complete the feedback form provided to you.