

# MATLAB EXPO

POLSKA 2024



Oprogramowanie  
Naukowo-Techniczne  
sp. z o.o.

# MATLAB EXPO

Warsaw, 04.06.2024 r.

## FPGA-based and embedded hardware support from MATLAB environment

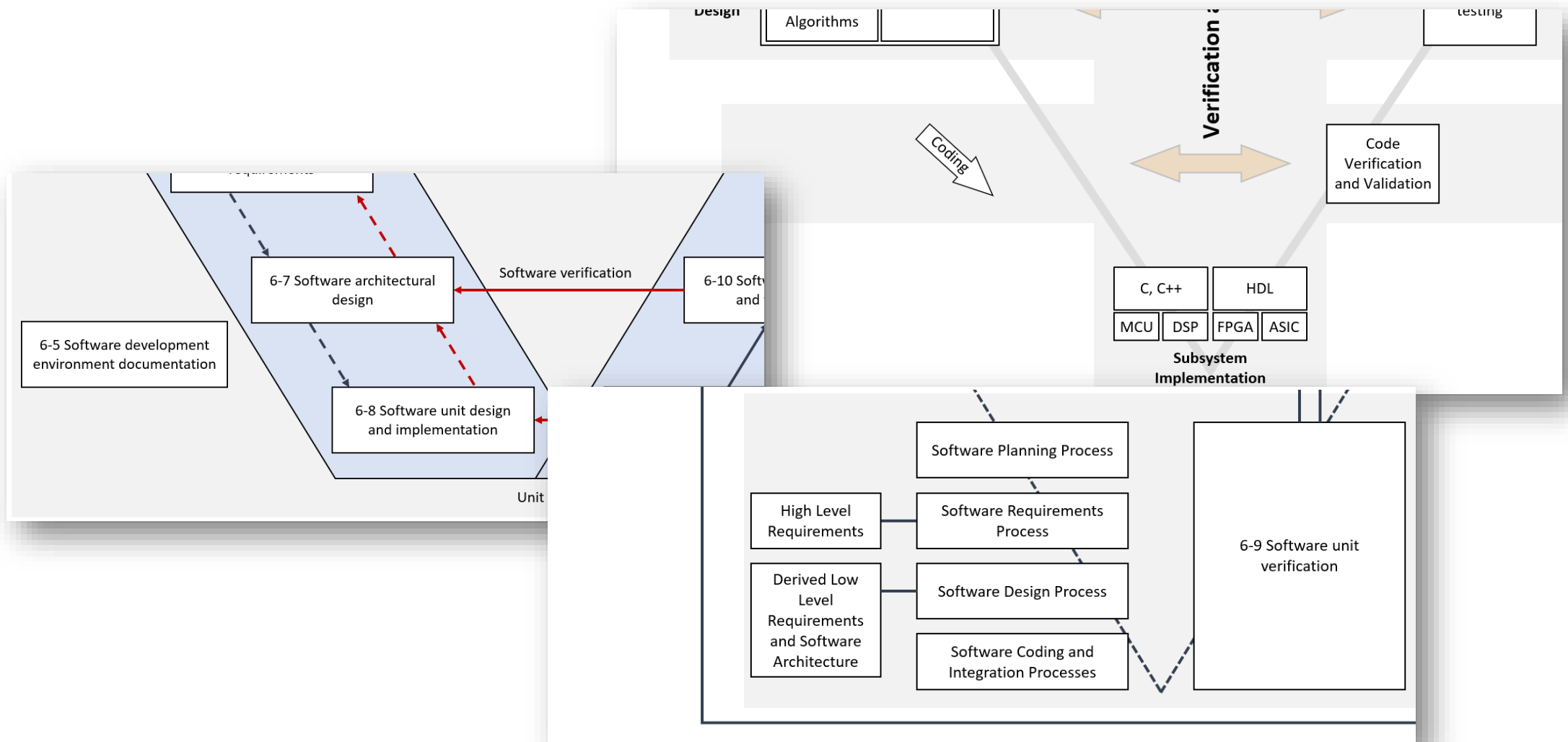
Konrad Kolski, ONT



# Embedded Hardware

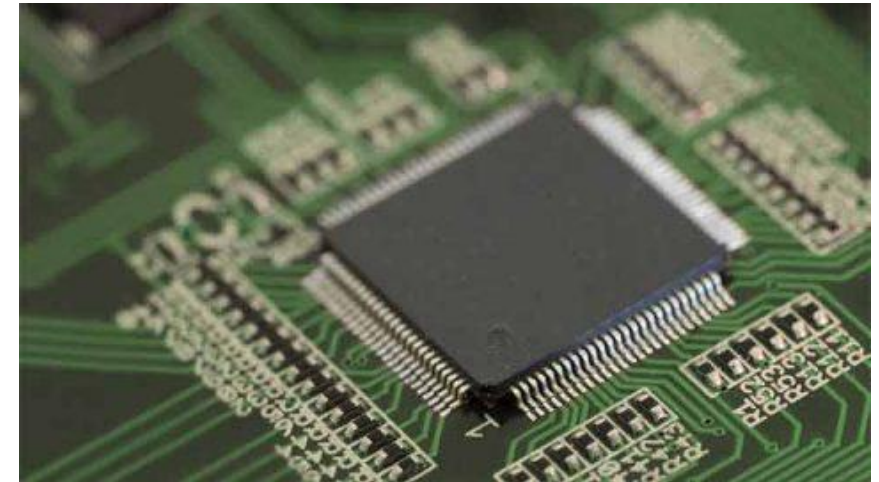


# Embedded Code in software workflows



## Introduction – Embedded Coder

- Generate ANSI/ISO C, C++ and encapsulated C++ production code
  - optimized for specific procesor architecture
  - allows extensive customisation (data types, variables etc.
- Optimize code for your goals
  - efficency , traceability, debugging and safety considerations
  - control format of each generated file
- Provides verification mechanisms
  - generates main program, multirate/multitask execution
  - SIL/PIL testing with support for coverage analysis
- Code is NOT a „Black box”
  - comes with code report and full documentation/comments
  - editable, customisable, full traceability to source

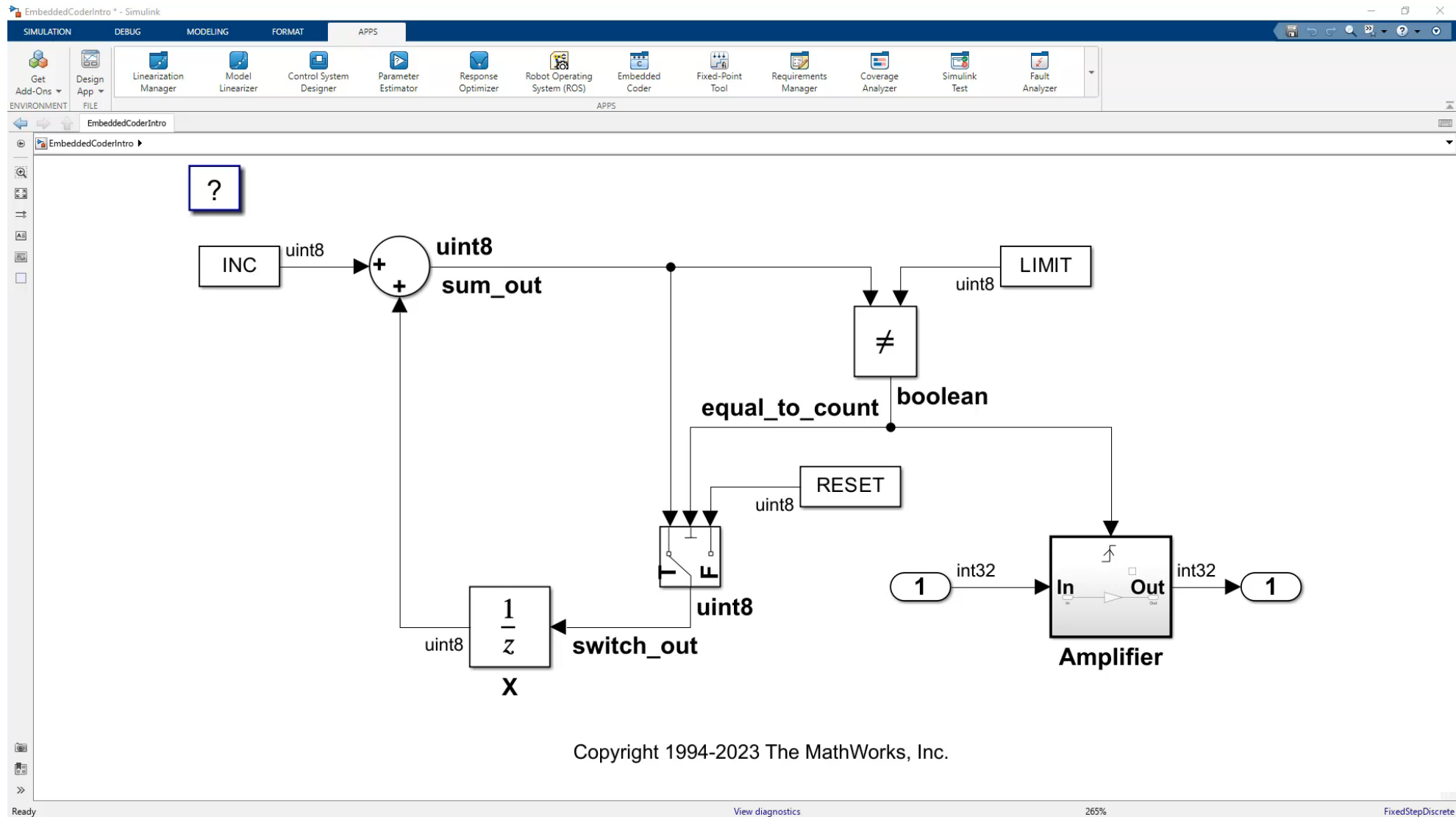


## Code Generation – Target Hardware

Vendor optimized C/C++ production code

- Code generation presets for most popular Embedded Hardware vendors
- Applies vendor derived configurations to code generation settings
- Supports custom device definitions
- Allows to generate production code for external IDE integration

The screenshot shows the 'Code Generation Target Configuration' dialog box. The 'Hardware board' is set to 'None'. The 'Code Generation system target file' is 'ert.tlc'. The 'Device vendor' is 'ARM Compatible' and the 'Device type' is 'ARM 10'. A dropdown menu for 'Device type' is open, showing options: ARM 10 (selected), ARM 11, ARM 64-bit (LLP64), ARM 64-bit (LP64), ARM 7, ARM 8, ARM 9, ARM Cortex-A, ARM Cortex-M, and ARM Cortex-R. Below this, the 'Device details' section shows data types and their sizes: char (8), short (16), int (32), long (32), long long (64), float (32), double (64), native (32), pointer (32), size\_t (32), ptrdiff\_t (32). The 'Byte ordering' is 'Little Endian' and 'Signed integer division rounds to' is 'Zero'. There are checkboxes for 'Shift right on a signed integer as arithmetic shift' (checked) and 'Support long long' (unchecked). At the bottom, a 'Custom Processor' dropdown is open, listing various vendors: AMD, ARM Compatible, Altera, Analog Devices, Atmel, Freescale, Infineon, Intel, Microchip, NXP, RISC-V, Renesas, STMicroelectronics, Texas Instruments, ASIC/FPGA, and Custom Processor (selected).

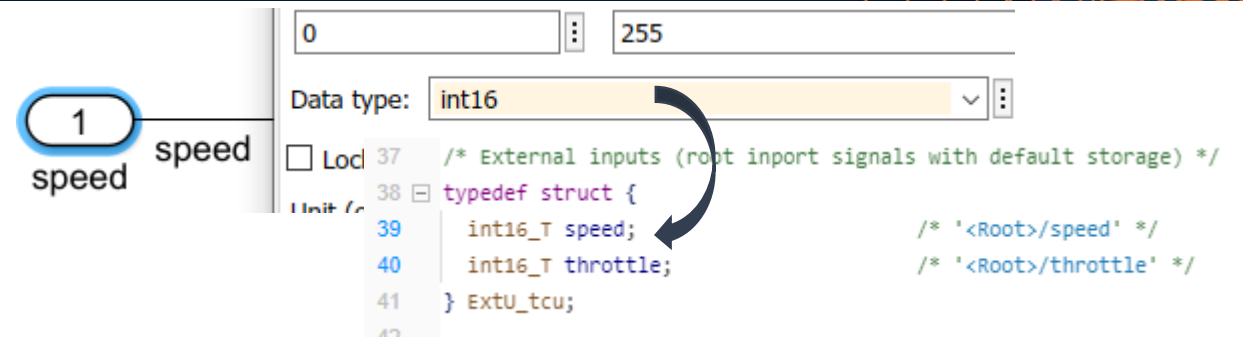


Copyright 1994-2023 The MathWorks, Inc.



## Code Customization

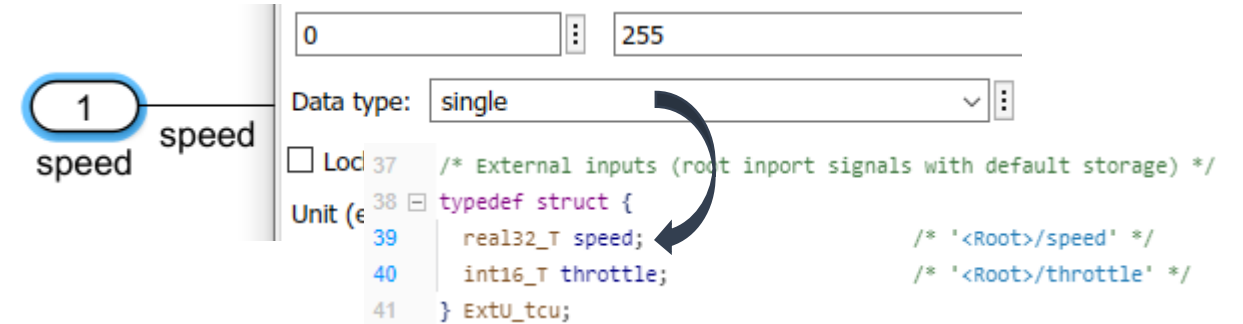
- Control Data Types of signals, ports and parameters and their appearance in source code
  - Model Simulates on same data types that code will run on
- Supports float and integer datatypes
  - (u)int8/16/32/64, half, single, double, boolean, enum, string
- Fixed-point data type support and code generation (with Fixed-Point Designer)



0 | 255  
 Data type: int16  
 speed  
 speed  

```

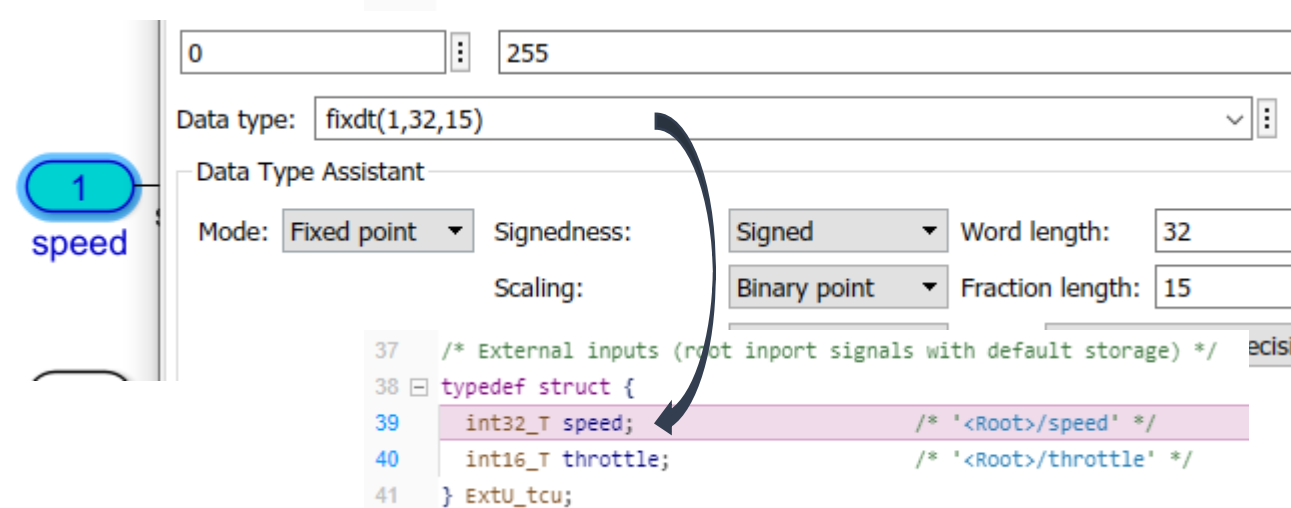
37 /* External inputs (root inport signals with default storage) */
38 typedef struct {
39     int16_T speed; /* '<Root>/speed' */
40     int16_T throttle; /* '<Root>/throttle' */
41 } ExtU_tcu;
    
```



0 | 255  
 Data type: single  
 speed  
 speed  

```

37 /* External inputs (root inport signals with default storage) */
38 typedef struct {
39     real32_T speed; /* '<Root>/speed' */
40     int16_T throttle; /* '<Root>/throttle' */
41 } ExtU_tcu;
    
```



0 | 255  
 Data type: fixdt(1,32,15)  
 speed  
 speed  
 Data Type Assistant  
 Mode: Fixed point Signedness: Signed Word length: 32  
 Scaling: Binary point Fraction length: 15  

```

37 /* External inputs (root inport signals with default storage) */
38 typedef struct {
39     int32_T speed; /* '<Root>/speed' */
40     int16_T throttle; /* '<Root>/throttle' */
41 } ExtU_tcu;
    
```

## Code Customization

- Precise control over data representation in source code
- Supports standard ISO/ANSI C/C++ data constructs i.e.
  - structures, pointers, global/local variables, bitfields
  - const, define, volatile and function accessed variables

Source	Storage Class	...
throttle	Auto	
speed	From signal object: ExportedGlobal	

global signals

Exported global signals are block signals with an external storage class designation. Code generation will declare these signals and export their symbols.

```

83
84
85
86
87 extern int32_T speed; /* '<Root>/speed' */
88
  
```

Source	Storage Class	...
throttle	Auto	
speed	From signal object: ImportedExternPointer	

Imported (extern) pointer block signals

```

71
72 extern int32_T *speed; /* '<Root>/speed' */
73
  
```

Source	Storage Class	...
throttle	Auto	
speed	From signal object: Struct	

Definition for custom storage class: Struct

```

68 typedef struct speed_struct_tag {
69     int32_T speed; /* '<Root>/speed' */
70 } speed_struct_type;
  
```

# Code Generation – Target Hardware

## Custom Hardware Target

▼ Code Generation

Optimization

Report

Comments

Identifiers

Custom Code

Interface

Code Style

Verification

Templates

Code Place

Data Type I

+

Drivers/External libraries

**h**

**c**

Code templates

Source file template:

Header file template:

Data templates

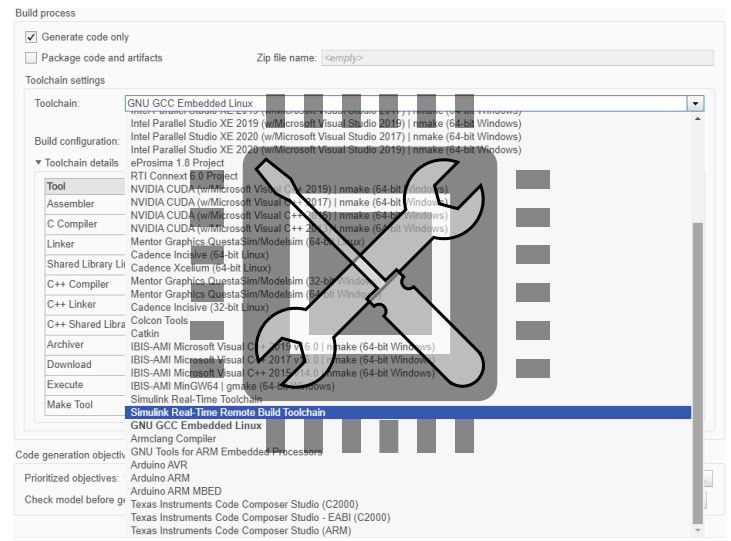
Source file template:

Header file template:

Code generation setting presets and custom parameters

Source File templates

3rd party tools (compiler, linker, programmer etc.)



Custom System Target File

System Target File Browser: tcu

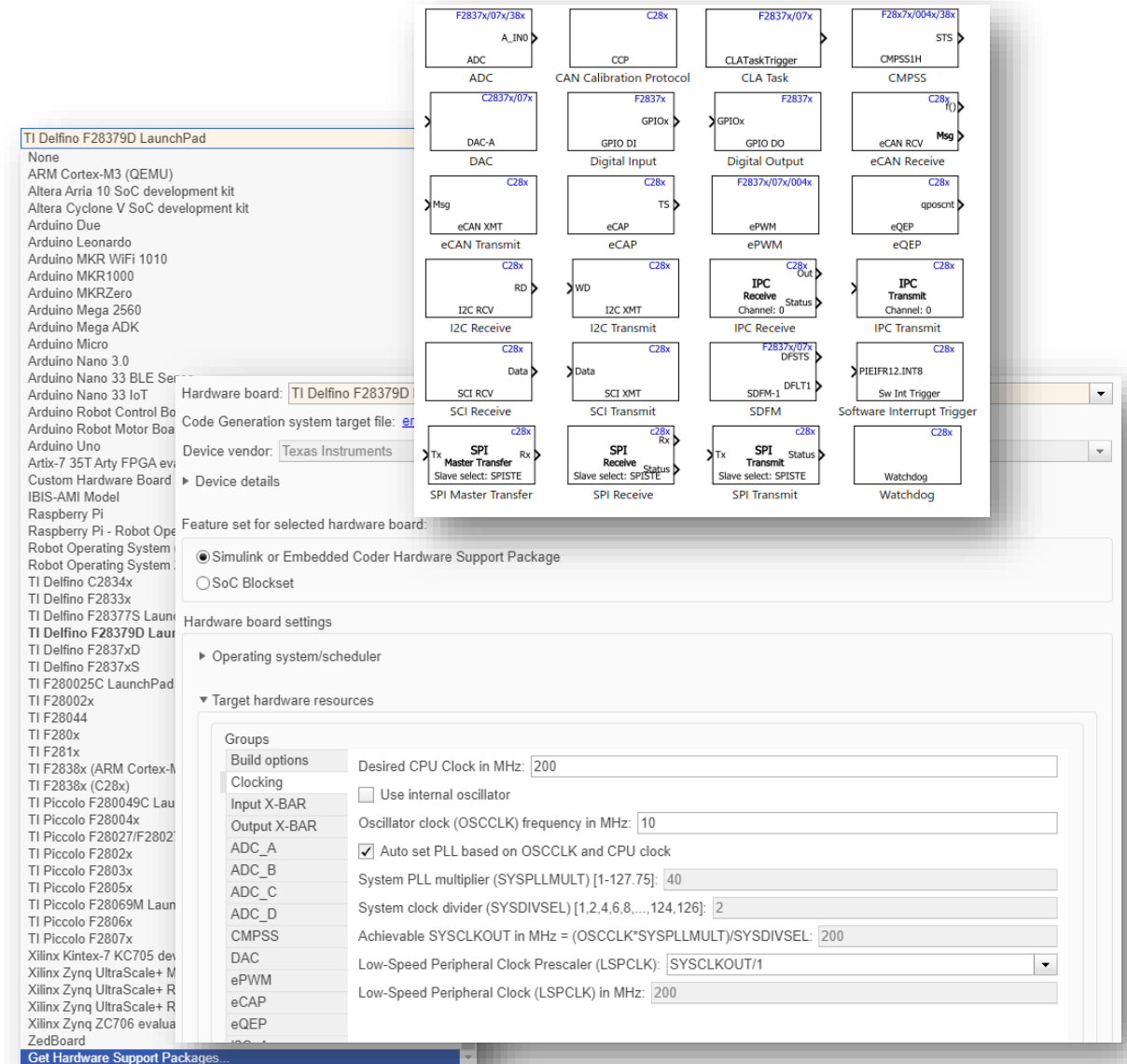
System Target File:	Description:
autosar.tlc	AUTOSAR
autosar_adaptive.tlc	AUTOSAR Adaptive
ert.tlc	Embedded Coder
ert.tlc	Create Visual C/C++ Sol
ert_shrllib.tlc	Embedded Coder (host-ba
grt.tlc	Generic Real-Time Target
grt.tlc	Create Visual C/C++ Sol
realtime.tlc	Run on Target Hardware
rsim.tlc	Rapid Simulation Target
rtwscfn.tlc	S-Function Target
sldr.tlc	Simulink Desktop Real-T
sldrtert.tlc	Simulink Desktop Real-T
slrealtime.tlc	Simulink Real-Time
systemverilog_dpi_ert.tlc	SystemVerilog DPI Comp
systemverilog_dpi_grt.tlc	SystemVerilog DPI Comp
tlmgenerator_ert.tlc	SystemC TLM Component G
tlmgenerator_grt.tlc	SystemC TLM Component G

Full Name:

# Code Generation – Target Hardware

## Hardware Support Packages

- Applies hardware specific configurations and features to code generation settings
  - hardware interrupts, timers, communication interfaces etc.
- Provides I/O drivers as MATLAB Functions/Simulink blocks

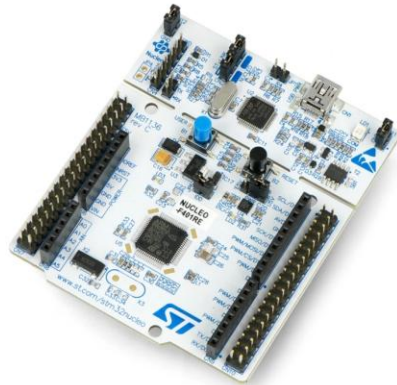


TI Delfino F28379D LaunchPad  
 None  
 ARM Cortex-M3 (QEMU)  
 Altera Arria 10 SoC development kit  
 Altera Cyclone V SoC development kit  
 Arduino Due  
 Arduino Leonardo  
 Arduino MKR WiFi 1010  
 Arduino MKR1000  
 Arduino MKRZero  
 Arduino Mega 2560  
 Arduino Mega ADK  
 Arduino Micro  
 Arduino Nano 3.0  
 Arduino Nano 33 BLE Ser  
 Arduino Nano 33 IoT  
 Arduino Robot Control Bo  
 Arduino Robot Motor Boa  
 Arduino Uno  
 Artix-7 35T Arty FPGA evi  
 Custom Hardware Board  
 IBIS-AMI Model  
 Raspberry Pi  
 Raspberry Pi - Robot Ope  
 Robot Operating System  
 Robot Operating System  
 TI Delfino C2834x  
 TI Delfino F2833x  
 TI Delfino F28377S Laun  
 TI Delfino F28379D Laun  
 TI Delfino F2837xD  
 TI Delfino F2837xS  
 TI F280025C LaunchPad  
 TI F28002x  
 TI F28044  
 TI F280x  
 TI F281x  
 TI F2838x (ARM Cortex-M  
 TI F2838x (C28x)  
 TI Piccolo F280049C Lau  
 TI Piccolo F28004x  
 TI Piccolo F28027/F2802  
 TI Piccolo F2802x  
 TI Piccolo F2803x  
 TI Piccolo F2805x  
 TI Piccolo F28069M Laun  
 TI Piccolo F2806x  
 TI Piccolo F2807x  
 Xilinx Kintex-7 KC705 dev  
 Xilinx Zynq UltraScale+ M  
 Xilinx Zynq UltraScale+ R  
 Xilinx Zynq UltraScale+ R  
 Xilinx Zynq ZC706 evalua  
 ZedBoard

Hardware board: TI Delfino F28379D  
 Code Generation system target file: [...](#)  
 Device vendor: Texas Instruments  
 Device details  
 Feature set for selected hardware board:  
 Simulink or Embedded Coder Hardware Support Package  
 SoC Blockset

Hardware board settings  
 Operating system/scheduler  
 Target hardware resources  
 Groups  
 Build options  
 Cloning  
 Input X-BAR  
 Output X-BAR  
 ADC\_A  
 ADC\_B  
 ADC\_C  
 ADC\_D  
 CMPSS  
 DAC  
 ePWM  
 eCAP  
 eQEP

Desired CPU Clock in MHz: 200  
 Use internal oscillator  
 Oscillator clock (OSCCLK) frequency in MHz: 10  
 Auto set PLL based on OSCCLK and CPU clock  
 System PLL multiplier (SYSPLLMULT) [1-127.75]: 40  
 System clock divider (SYSDIVSEL) [1,2,4,6,8,...,124,126]: 2  
 Achievable SYSCLKOUT in MHz = (OSCCLK\*SYSPLLMULT)/SYSDIVSEL: 200  
 Low-Speed Peripheral Clock Prescaler (LSPCLK): SYSCLKOUT/1  
 Low-Speed Peripheral Clock (LSPCLK) in MHz: 200



» <http://www.mathworks.com/hardware-support/>

## Benefits of automated code generation



- Quality
  - Directly related to quality of templates and tool used to generate code



- Consistency
  - Variable, function, method and class names built in the same way in all of the output code



- Productivity
  - Running code generator iteratively seamlessly updates the output code with every change



- Abstraction
  - Code is being build based on abstract model of target code which becomes source for the code generator



- Quality of templates/tool increases quality of the whole codebase
- Easy to understand, use and layer more functionality
- Ability to seamlessly update code based on changing requirements
- Code generator can be retargeted to build code for different platform while source model remains the same

# MATLAB EXPO

Warszawa | 4.06.2024

Konrad Kolski

Application Engineer, ONT

[konrad.kolski@ont.com.pl](mailto:konrad.kolski@ont.com.pl)

