Optymalizacja deploymentu funkcji ze środowiska MATLAB na platformy Cortex-M4

Dr hab. inż. Adam Jabłoński, prof. AGH

Akademia Górniczo-Hutnicza im. Stanisława Staszica w Krakowie

AGH University of Krakow

MATLAB EXPO, Warszawa,04.06.2024 r.

Opis problemu

my_signal_processing.m

```
function out = my_signal_processing(x)

% my code goes here

out = fft(x);

% my code goes here

end
```

Urządzenia wbudowane

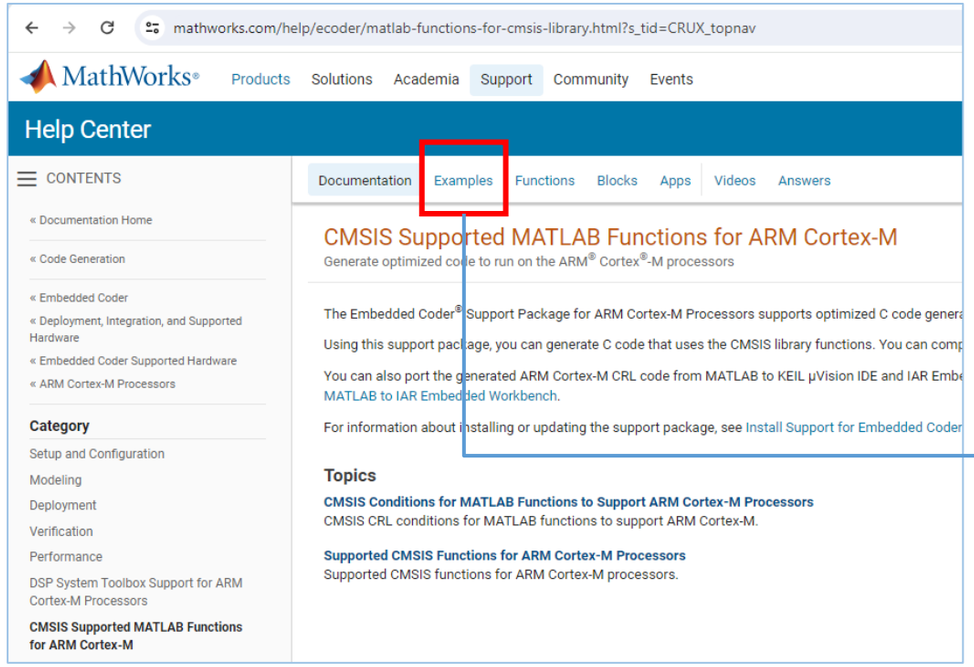
ARM Cortex znajdują zastosowanie w szerokiej gamie urządzeń elektronicznych, takich jak smartfony, tablety, smartwatche, nowoczesny sprzęt AGD i samochody.

Signal Processing

Output

Dzięki narzędziu MATLAB CRL oraz dostosowaniu *m-kodu* do CMSIS DSP, możliwe jest automatyczny deploymnet algorytmów

Motywacja tematu prezentacji



Do dziś (27.05.2024 r.) w zakładce „Examples” MATLAB (2024a) strona *MathWorks* wyświetla: „*No results*”.

Rozwiązanie problemu

Krok 1: analiza „*MATLAB conditions*” dla używanych funkcji

Krok 2: instalacja komponentów (EC, HSP, CMSIS, compiler)

Krok 3: analiza bibliotek producenta (weryfikacja krzyżowa)

Krok 4: przygotowanie skryptu demo

Krok 5: obsługa procedury Wizarda

Krok 6: weryfikacja via Coder Report Viewer (CRV)

Krok 7: pakowanie zip

Rozwiązanie problemu

Krok 1: analiza „*MATLAB conditions*”

Krok 2: instalacja komponentów

Krok 3: analiza bibliotek producenta

Krok 4: przygotowanie skryptu demo

Krok 5: obsługa procedury Wizarda

Krok 6: weryfikacja via CRV

Krok 7: pakowanie zip

CMSIS Conditions for MATLAB Functions to Support ARM Cortex-M Processors

R2024a

Each MATLAB® function that can be used with the Support Package for ARM® Cortex®-M processors requires specific conditions to allow code replacement by using the CMSIS Library. You use this code replacement when code.

If you do not meet the specific requirements, then the generated C code runs on the ARM Cortex-M processors. However, this generated code does not use CMSIS library support.

The CMSIS library supports these MATLAB functions only when you set specific properties, as indicated in this table:

MATLAB Function	Supported Function Signatures	Input Requirements	Parameter Requirements
`fft`	Y = fft(X) Y = fft(X, n) Y = fft(X, n, dim)	• Real or Complex Values • Multichannel Input (Multiple columns) • `single` data type	**Transform Length `n`:** • Real Input: `n` must be in the range `[9 2048] U {4096}-{16}`. • Complex Input: `n` must be in the range `[5 2048] U {4096}-{8}`. **Dimension to operate along `dim`:** `dim` must be a compile time constant.

Rozwiązanie problemu

Krok 1: analiza „MATLAB conditions”

Krok 2: instalacja komponentów

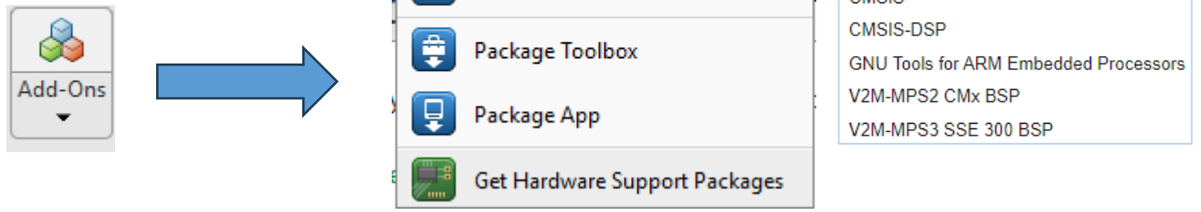
Krok 3: analiza bibliotek producenta

Krok 4: przygotowanie skryptu demo

Krok 5: obsługa procedury Wizarda

Krok 6: weryfikacja via CRV

Krok 7: pakowanie zip

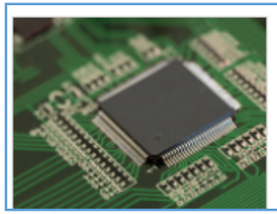


1) Arm_Compiler_Support_Package

2) Embedded Coder

3) MATLAB Support for MinGW-w64

4) CMSIS **CMSIS-DSP** Version 1.15.0
CMSIS DSP Software Library

5) 
Embedded Coder Support Package for ARM Cortex-M Processors by MathWorks Embedded Coder Team STAFF

Generate code optimized for **Cortex-M** processors.

Embedded Coder® Support Package for ARM® **Cortex®-M** Processors lets you generate optimized code for math operations using the CMSIS library. Use this generated code for ARM **Cortex-M** processors. For

Rozwiązanie problemu

Krok 1: analiza „MATLAB conditions”

Krok 2: instalacja komponentów

Krok 3: analiza bibliotek producenta

Krok 4: przygotowanie skryptu demo

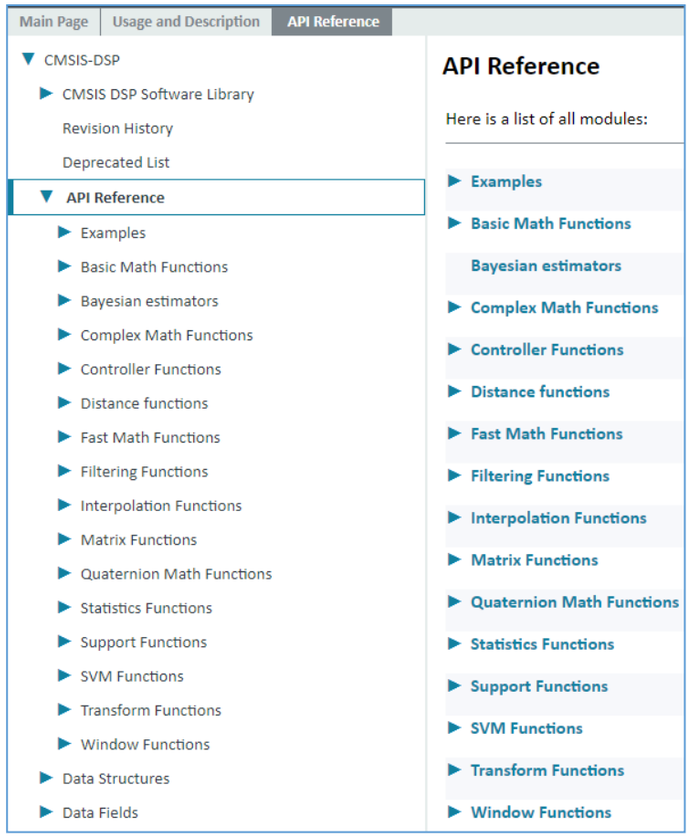
Krok 5: obsługa procedury Wizarda

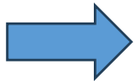
Krok 6: weryfikacja via CRV

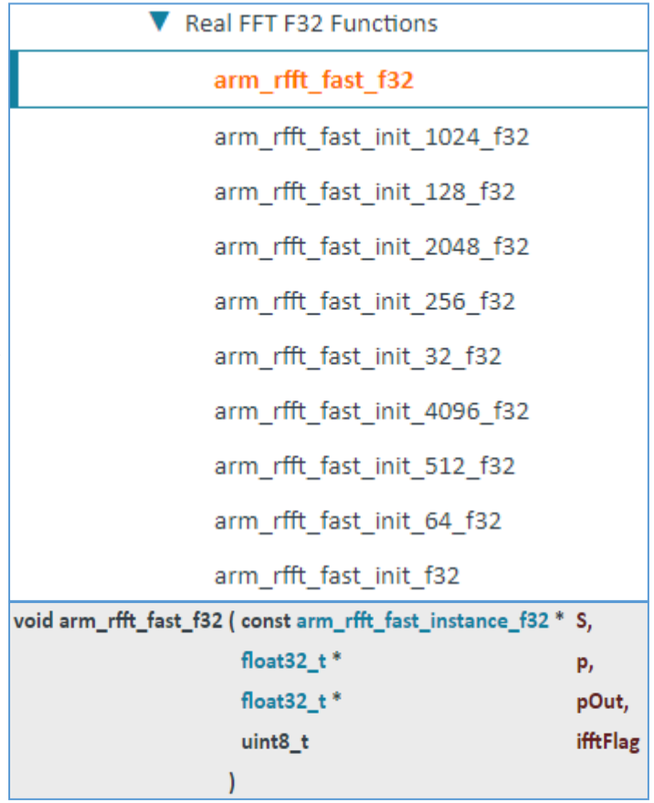
Krok 7: pakowanie zip

CMSIS **CMSIS-DSP** Version 1.15.0
CMSIS DSP Software Library

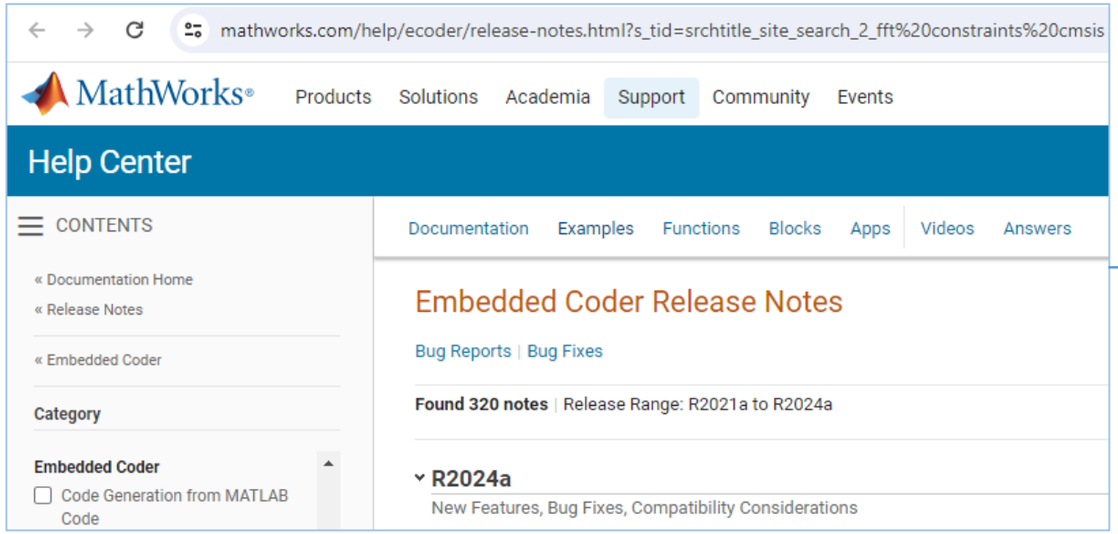
C Programming

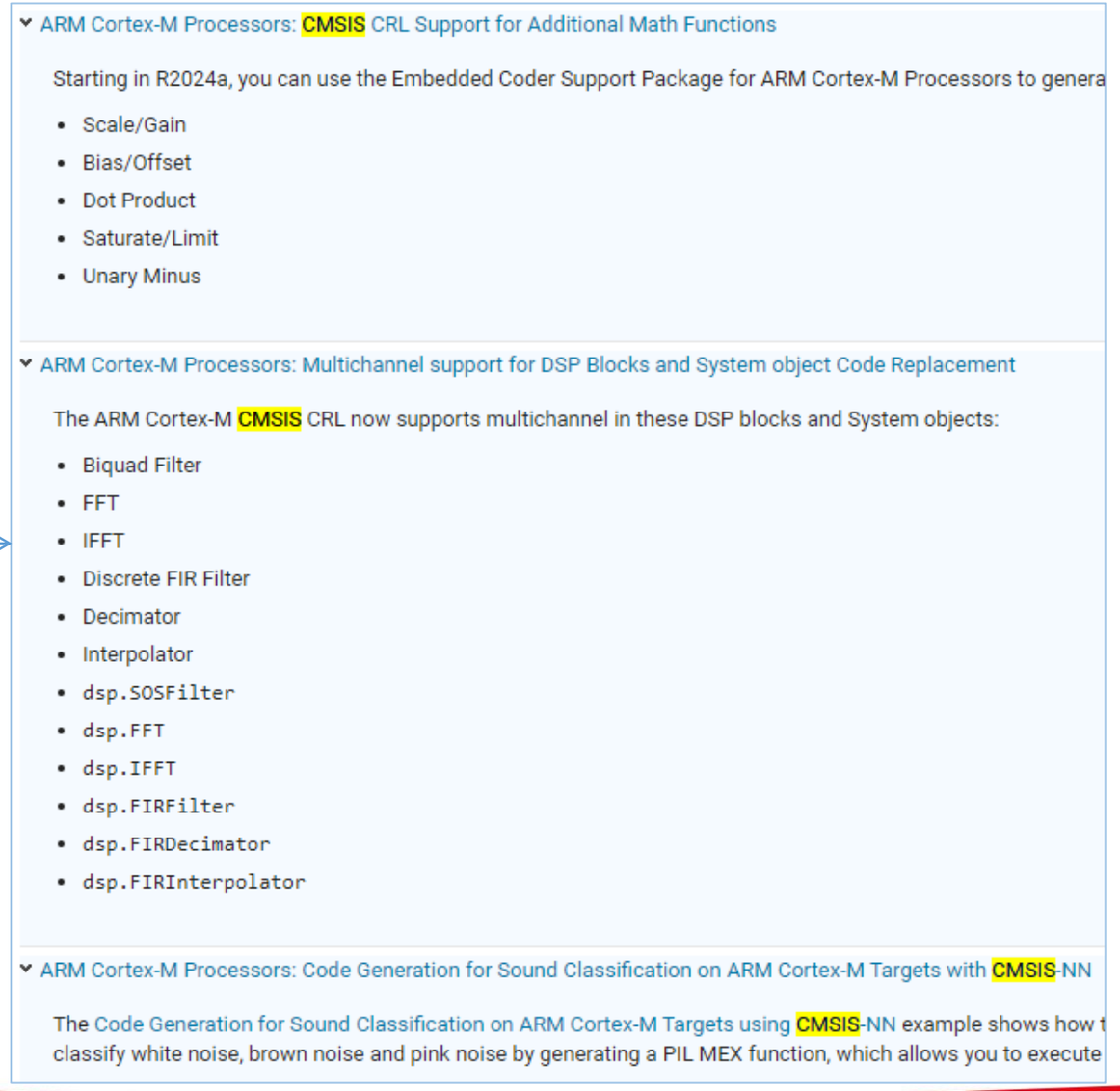






*Krok 3: analiza bibliotek producenta - **BTW**...*





Z każdą nową wersja środowiska MATLAB, zwiększa się ilość wspieranych funkcji interfejsu CMSIS w ramach CRL.

Rozwiązanie problemu

Krok 1: analiza „MATLAB conditions”

Krok 2: instalacja komponentów

Krok 3: analiza bibliotek producenta

Krok 4: przygotowanie skryptu demo

Krok 5: obsługa procedury Wizarda

Krok 6: weryfikacja via CRV

Krok 7: pakowanie zip

my_signal_processing.m

```
function out = my_signal_processing(x)

% my code goes here

out = fft(x);

% my code goes here

end
```

calling_script.m

```
1   clear; clc; close all;
2
3   x   = rand(1,1e3);
4   out = my_signal_processing(x);
5
6   disp(out)
```

Skrypt demo umożliwia automatyczną sugestię typów i rozmiarów parametrów wejściowych oraz sprawdzenie wykonywalności funkcji.

Rozwiązanie problemu

Krok 1: analiza „MATLAB conditions”

Krok 2: instalacja komponentów

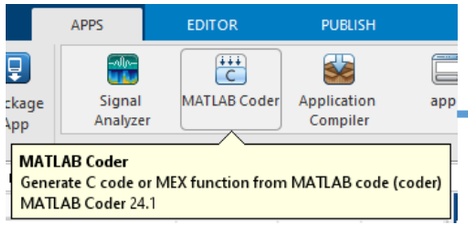
Krok 3: analiza bibliotek producenta

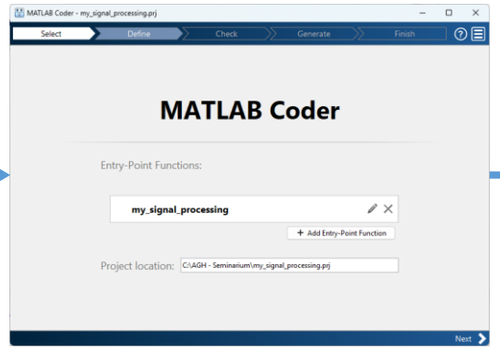
Krok 4: przygotowanie skryptu demo

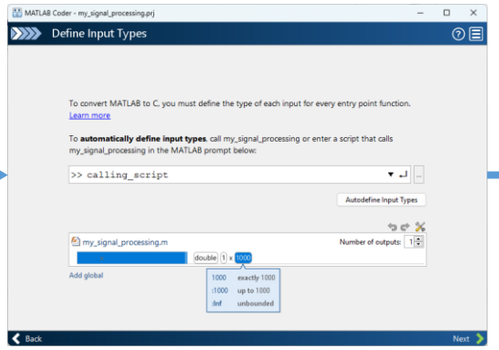
Krok 5: obsługa procedury Wizarda

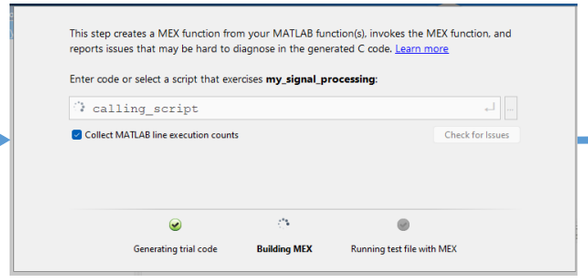
Krok 6: weryfikacja via CRV

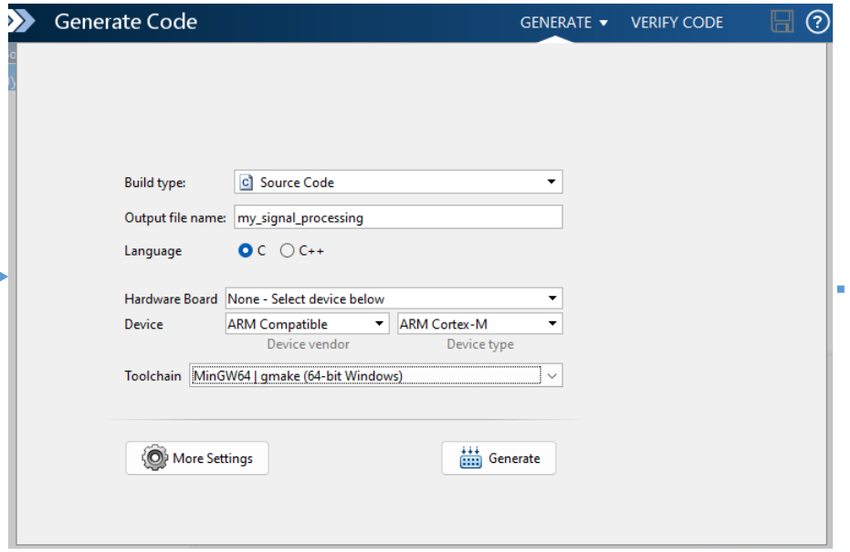
Krok 7: pakowanie zip











Rozwiązanie problemu

Krok 1: analiza „MATLAB conditions”

Krok 2: instalacja komponentów

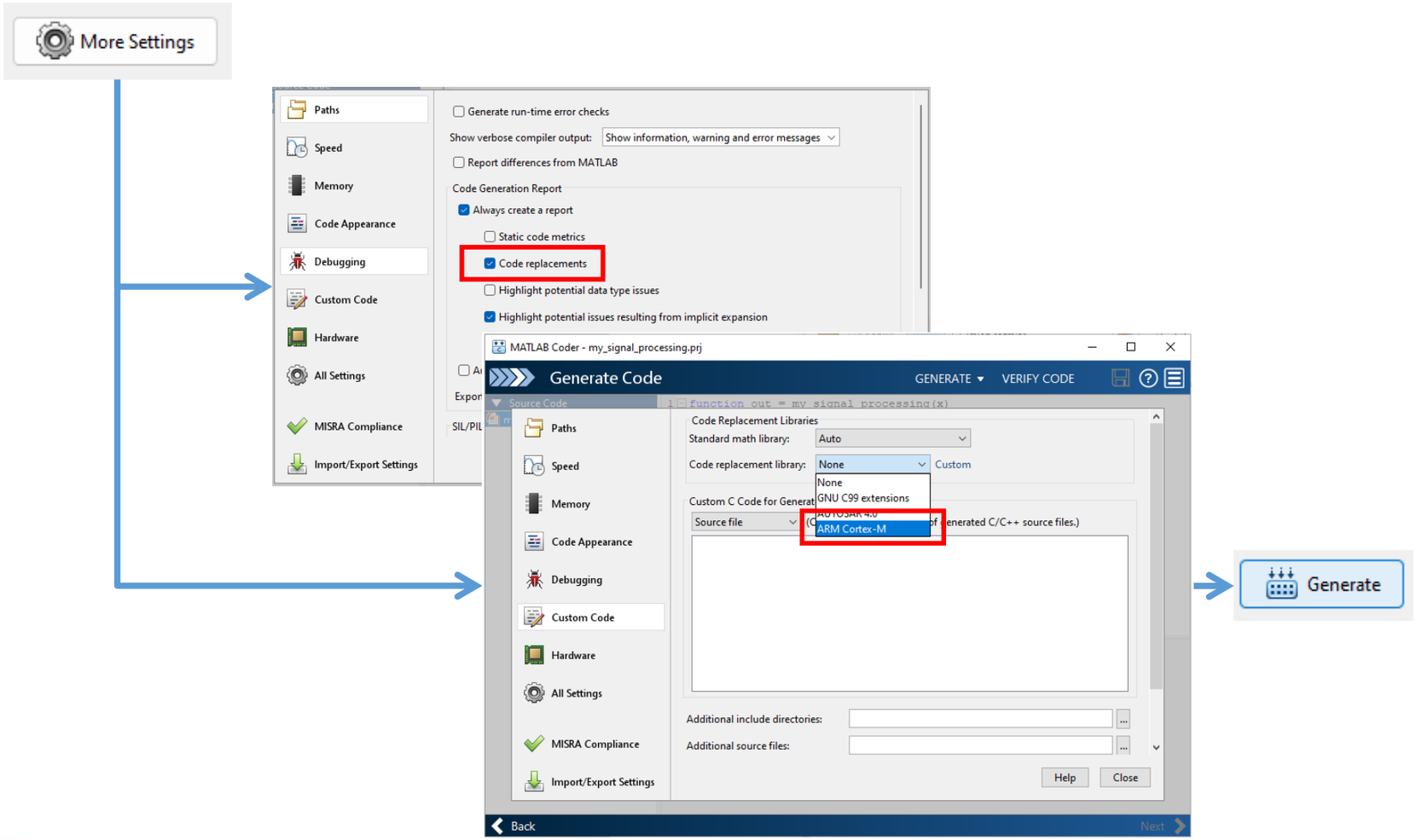
Krok 3: analiza bibliotek producenta

Krok 4: przygotowanie skryptu demo

Krok 5: obsługa procedury Wizarda

Krok 6: weryfikacja via CRV

Krok 7: pakowanie zip



Rozwiązanie problemu

Krok 1: analiza „MATLAB conditions”

Krok 2: instalacja komponentów

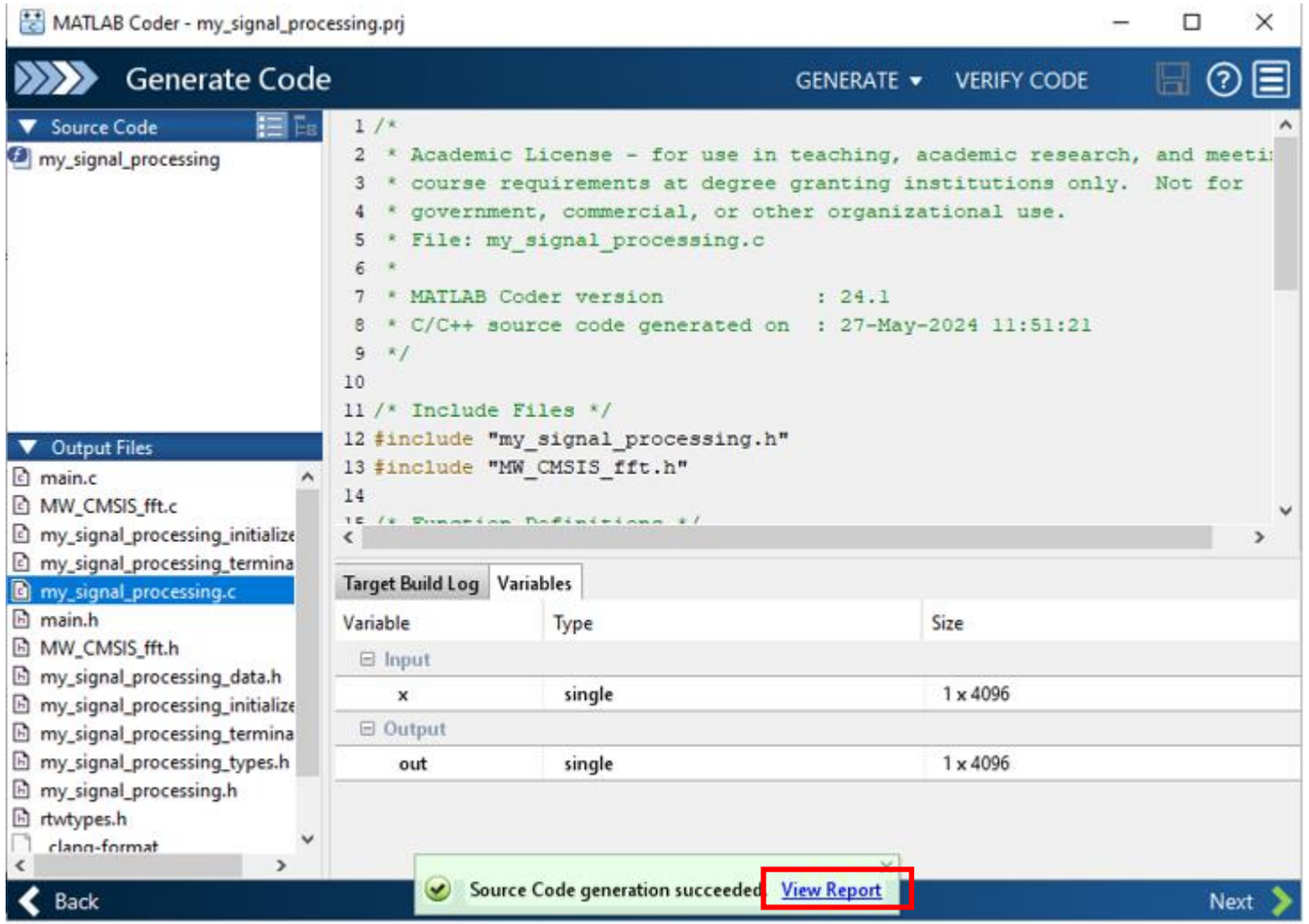
Krok 3: analiza bibliotek producenta

Krok 4: przygotowanie skryptu demo

Krok 5: obsługa procedury Wizarda

Krok 6: weryfikacja via CRV

Krok 7: pakowanie zip



Rozwiązanie problemu

Krok 1: analiza „MATLAB conditions”

Krok 2: instalacja komponentów

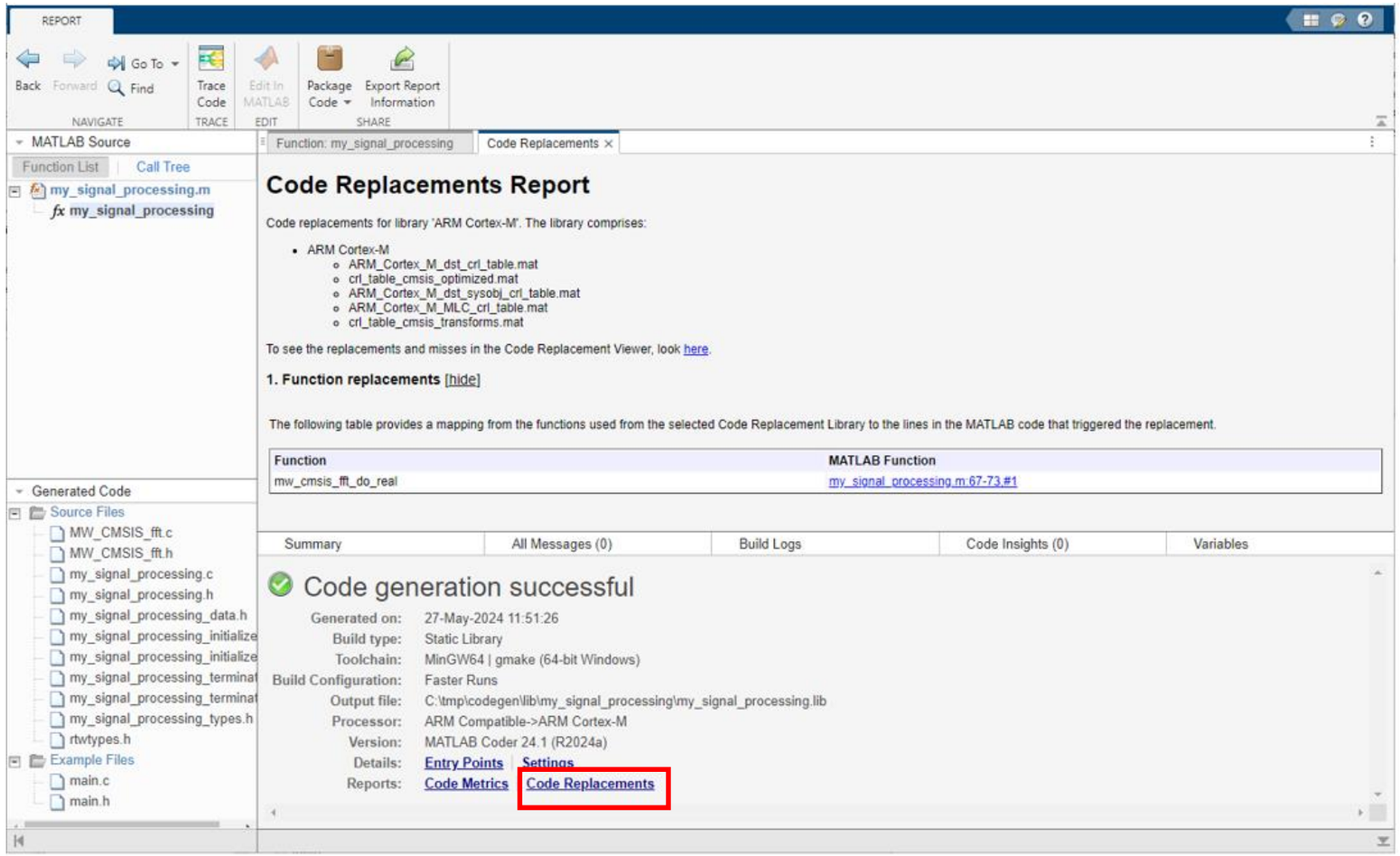
Krok 3: analiza bibliotek producenta

Krok 4: przygotowanie skryptu demo

Krok 5: obsługa procedury Wizarda

Krok 6: weryfikacja via CRV

Krok 7: pakowanie zip



Rozwiązanie problemu

Krok 1: analiza „MATLAB conditions”

Krok 2: instalacja komponentów

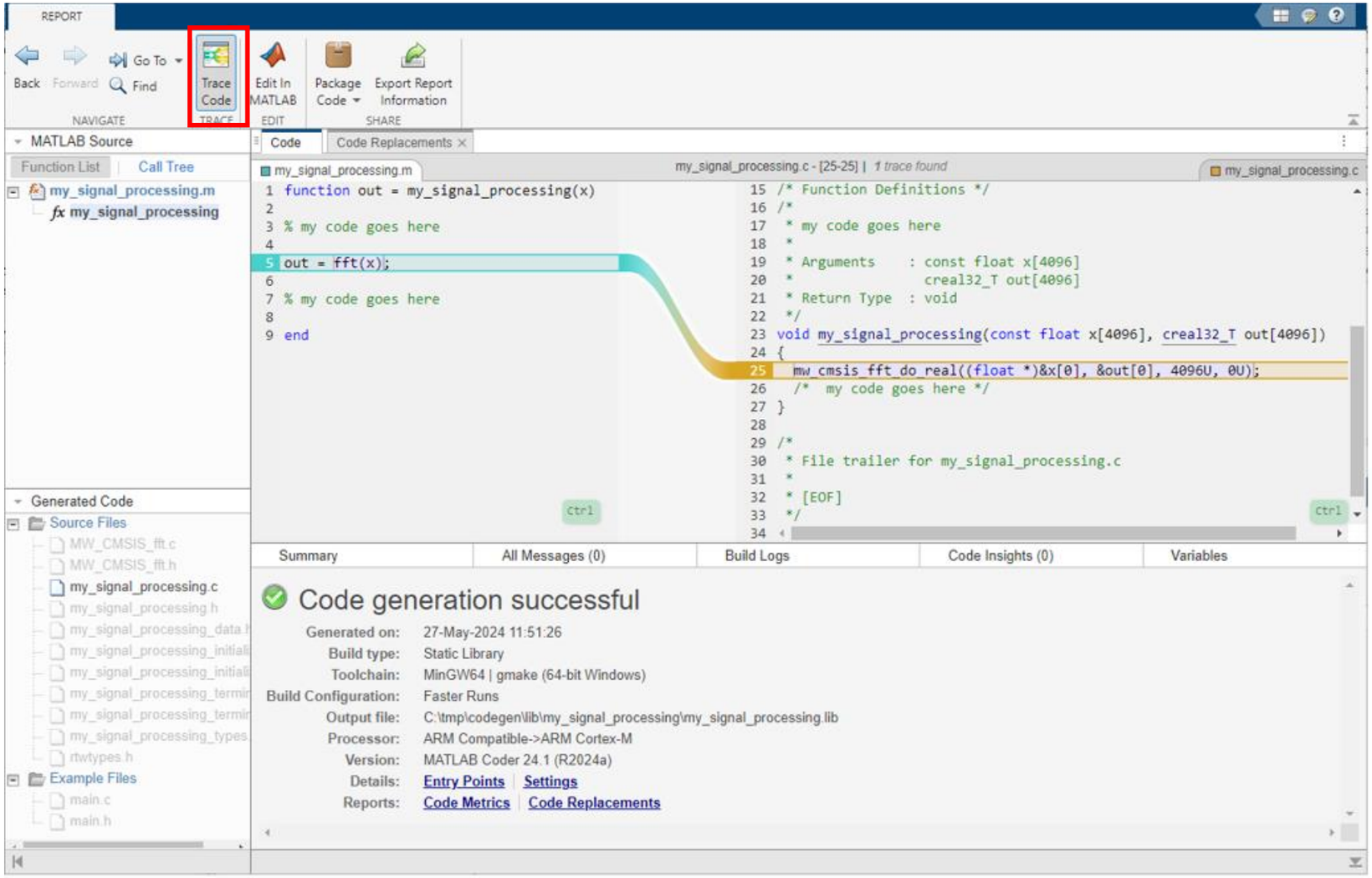
Krok 3: analiza bibliotek producenta

Krok 4: przygotowanie skryptu demo

Krok 5: obsługa procedury Wizarda

Krok 6: weryfikacja via CRV

Krok 7: pakowanie zip



*Krok 6: weryfikacja via CRV: **Porównanie** NOK vs OK…*

```
12 #include "my_signal_processing.h"
13 #include "FFTImplementationCallback.h"
14 #include "my_signal_processing_data.h"
```

```
12 #include "my_signal_processing.h"
13 #include "MW_CMSIS_fft.h"
```

MW_CMSIS_fft.c

MW_CMSIS_fft.h

Code Replacements Report

Code replacements for library 'ARM Cortex-M'. The library comprises:

- ARM Cortex-M
  - ARM_Cortex_M_dst_crl_table.mat
  - crl_table_cmsis_optimized.mat
  - ARM_Cortex_M_dst_sysobj_crl_table.mat
  - ARM_Cortex_M_MLC_crl_table.mat
  - crl_table_cmsis_transforms.mat

To see the replacements and misses in the Code Replacement Viewer, look here.

1. Function replacements [hide]

The following table provides a mapping from the functions used from the selected Code Replacement Library to the lines in the MATLAB code that triggered the replacement.

Function	MATLAB Function
memset	Replacements triggered in internal file: FFTImplementationCallback.m

Summary | All Messages (0) | Build Logs | Code Insights (0) | Variables

Code generation successful

Generated on:	27-May-2024 11:34:51
Build type:	Static Library
Toolchain:	MinGW64 | gmake (64-bit Windows)
Build Configuration:	Faster Runs
Output file:	C:\tmp\codegen\lib\my_signal_processing\my_signal_processing.lib
Processor:	ARM Compatible->ARM Cortex-M
Version:	MATLAB Coder 24.1 (R2024a)
Details:	Entry Points | Settings
Reports:	Code Metrics | Code Replacements

x = rand(1,1e3);

vs

Code Replacements Report

Code replacements for library 'ARM Cortex-M'. The library comprises:

- ARM Cortex-M
  - ARM_Cortex_M_dst_crl_table.mat
  - crl_table_cmsis_optimized.mat
  - ARM_Cortex_M_dst_sysobj_crl_table.mat
  - ARM_Cortex_M_MLC_crl_table.mat
  - crl_table_cmsis_transforms.mat

To see the replacements and misses in the Code Replacement Viewer, look here.

1. Function replacements [hide]

The following table provides a mapping from the functions used from the selected Code Replacement Library to the lines in the MATLAB code that triggered the replacement.

Function	MATLAB Function
mw_cmsis_fft_do_real	my_signal_processing.m:67-73,#1

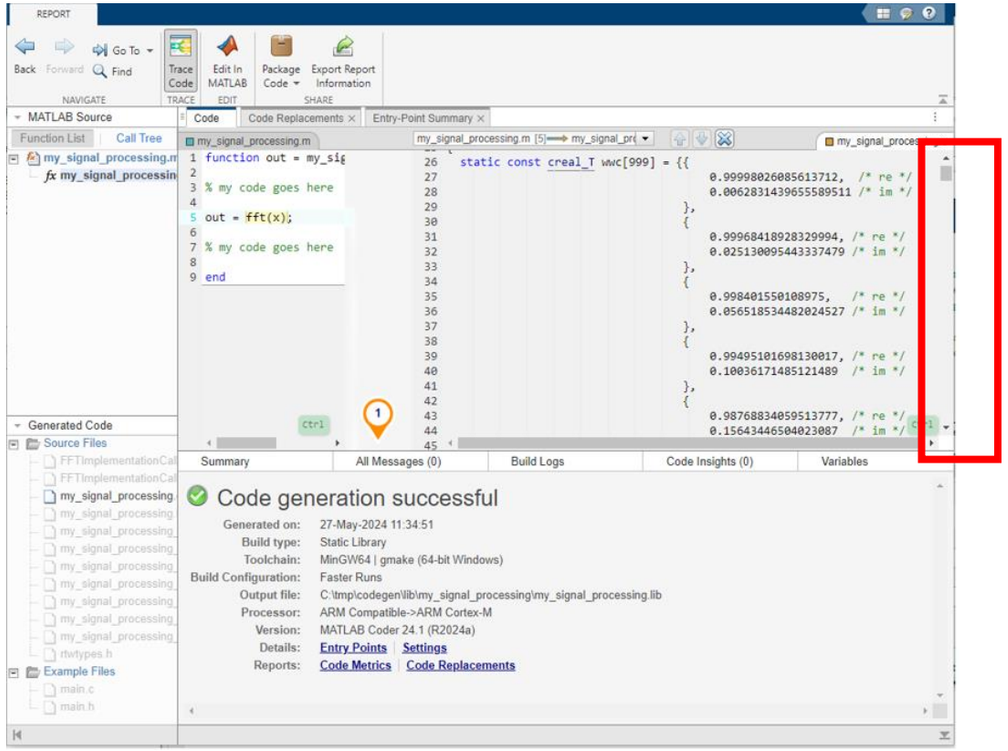
Summary | All Messages (0) | Build Logs | Code Insights (0) | Variables

Code generation successful

Generated on:	27-May-2024 11:51:26
Build type:	Static Library
Toolchain:	MinGW64 | gmake (64-bit Windows)
Build Configuration:	Faster Runs
Output file:	C:\tmp\codegen\lib\my_signal_processing\my_signal_processing.lib
Processor:	ARM Compatible->ARM Cortex-M
Version:	MATLAB Coder 24.1 (R2024a)
Details:	Entry Points | Settings
Reports:	Code Metrics | Code Replacements

x = single(rand(1,4096));

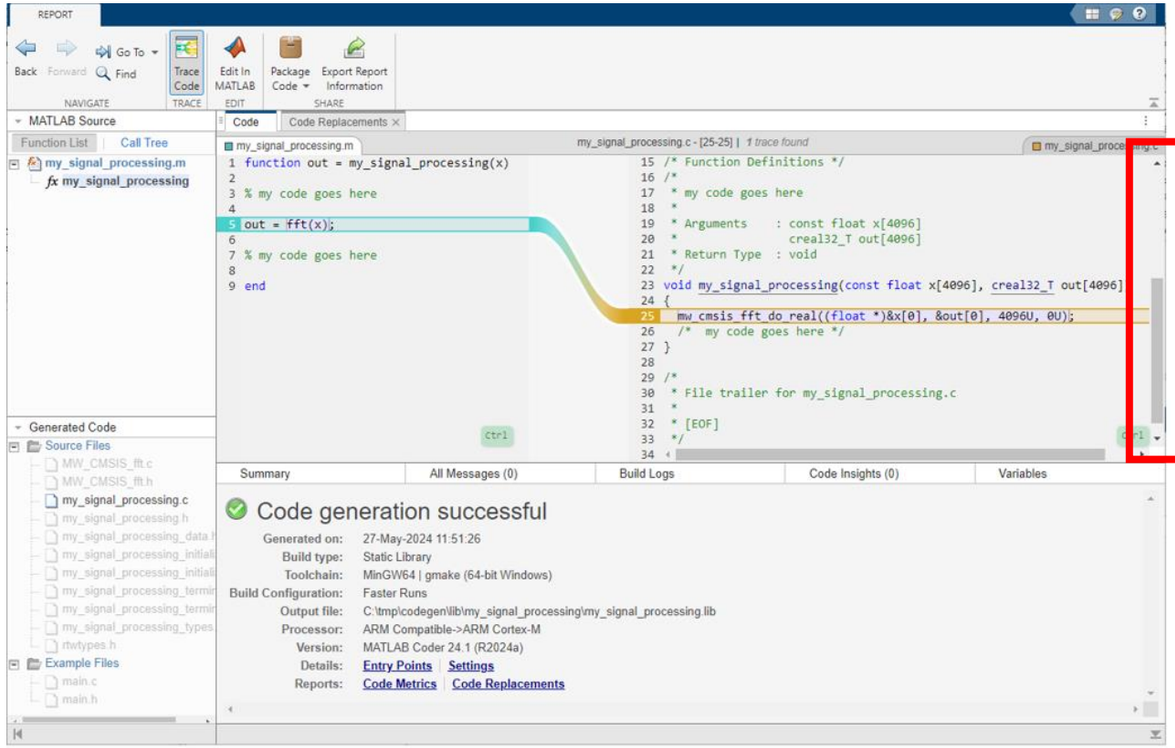
*Krok 6: weryfikacja via CRV: **Porównanie** NOK vs OK…*



5000+ linii

vs

4 linijki




```
x = rand(1,1e3);
```

```
x = single(rand(1,4096));
```

Rozwiązanie problemu

Krok 1: analiza „MATLAB conditions”

Krok 2: instalacja komponentów

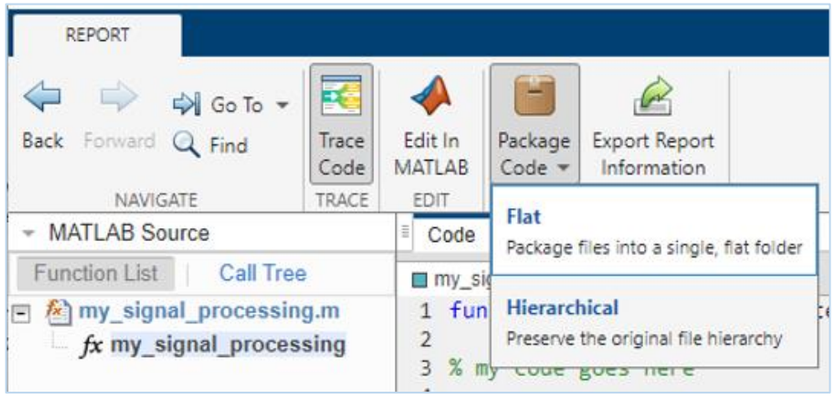
Krok 3: analiza bibliotek producenta

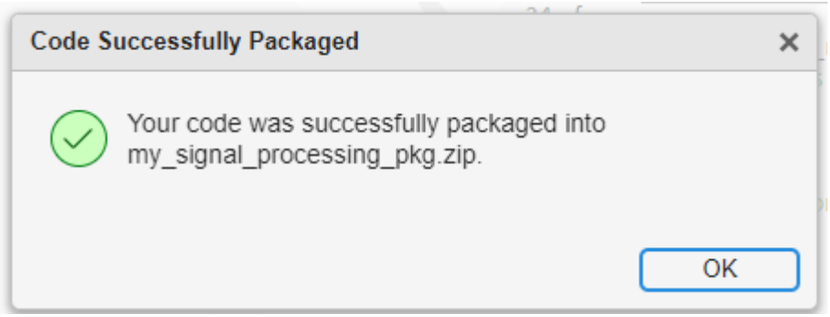
Krok 4: przygotowanie skryptu demo

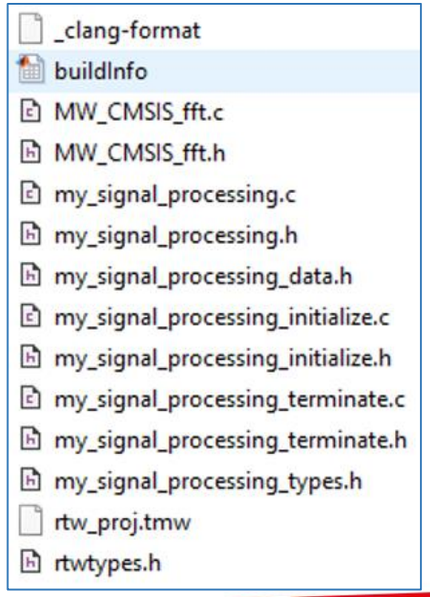
Krok 5: obsługa procedury Wizarda

Krok 6: weryfikacja via Wizard Viewer

Krok 7: pakowanie zip







Praktyczne wskazówki

@Krok 1: funkcje z *DSP System Objects* (filtry) są opisane osobno od funkcji z rodziny *”FFT”*

@Krok 2: po instalacji wtyczek, MATLAB musi być zrestartowany

@Krok 3: zaleca się uruchomienie „*crviewer*” w *Command Window*

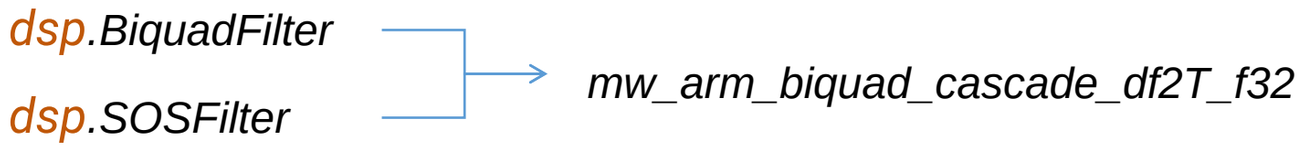
@Krok 4: zaleca się centralizację kodu MATLAB oraz statyczną konwersję precyzji

@Krok 5: konkretne projekty wymagają dodatkowej minimalizacji alokacji pamięci

@Krok 6: ręczna analiza „*Code Replacements*” jest zalecana do weryfikacji

@Krok 7: użycie interfejsu *zip* zapewnia, że dodane zostaną odpowiednie biblioteki MATLAB

@deployment filtrów: referencja [4] pokazuje rozwiązanie półautomatyczne; wzorując się na prezentacji można zautomatyzowac proces, uważając, aby cutoff był wyraźnie różny od wartości granicznych zakresu widmowego



Dziękuję za uwagę!

Bibliografia:

[1] Adam Jablonski, Condition Monitoring Algorithms in MATLAB, Springer, 2021

[2] https://www.mathworks.com/help/ecoder/matlab-functions-for-cmsis-library.html?s_tid=CRUX_lftnav

[3] https://www.arm.com/technologies/cmsis

[4] https://gaidi.ca/weblog/implementing-a-biquad-cascade-iir-filter-on-a-cortex-m4/