

Polyspace Hands-on Workshop

Case Study



龚小平

Kevin.GONG@mathworks.cn



目标

- Understanding Polyspace workflow
- Learn how to use Polyspace
 - Checking MISRA C rules violations
 - Checking defects and runtime errors

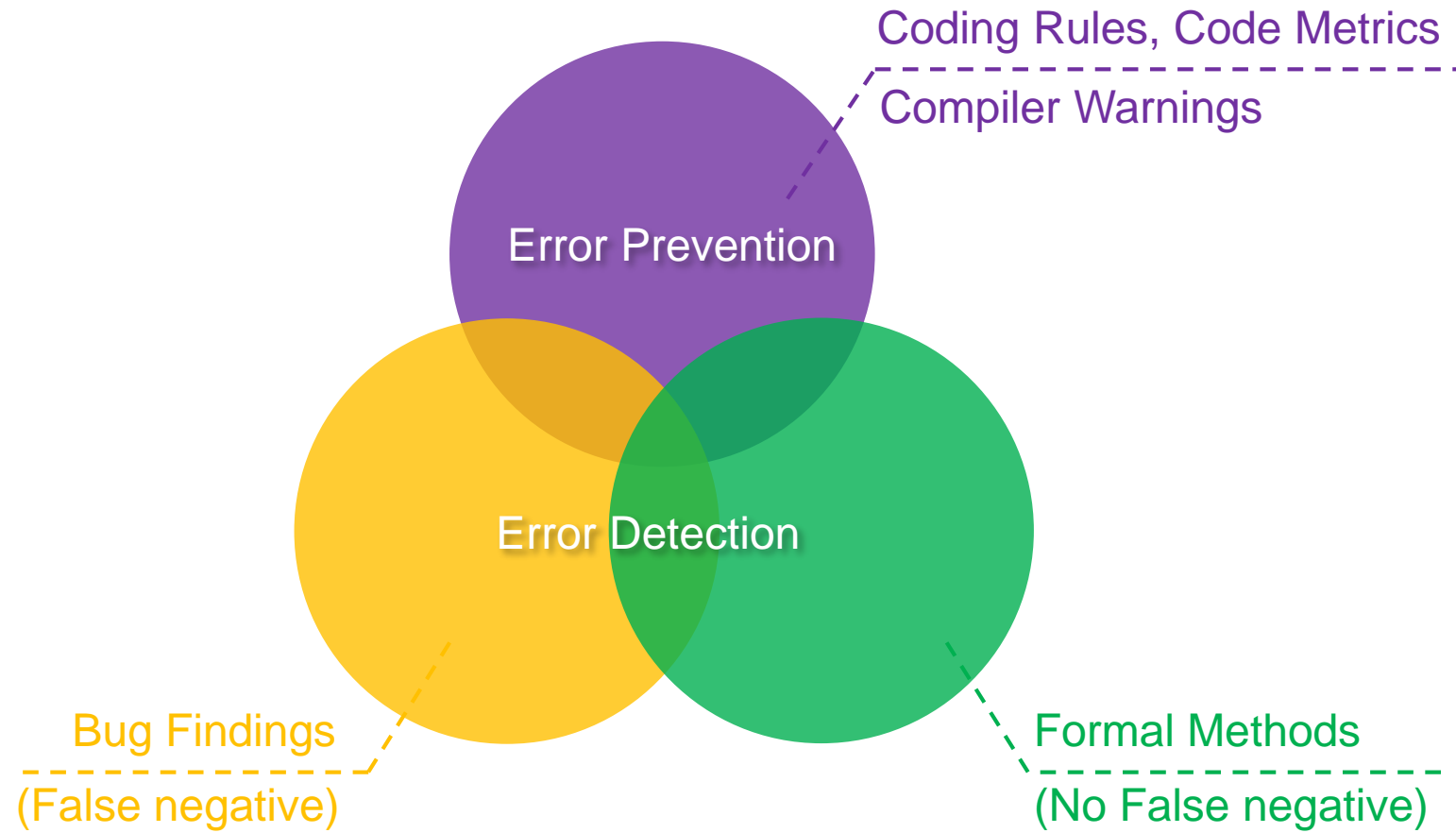
内容

- Polyspace Overview
 - Category of Static analysis
 - Use case of Polyspace products
- Polyspace Hands-on workshop
 - Description of the logic
 - Exercise 1 / 2 / 3 / 4 (Review MISRA, Defect, Code Metrics, Runtime Error)
 - Conclusion
- Q&A

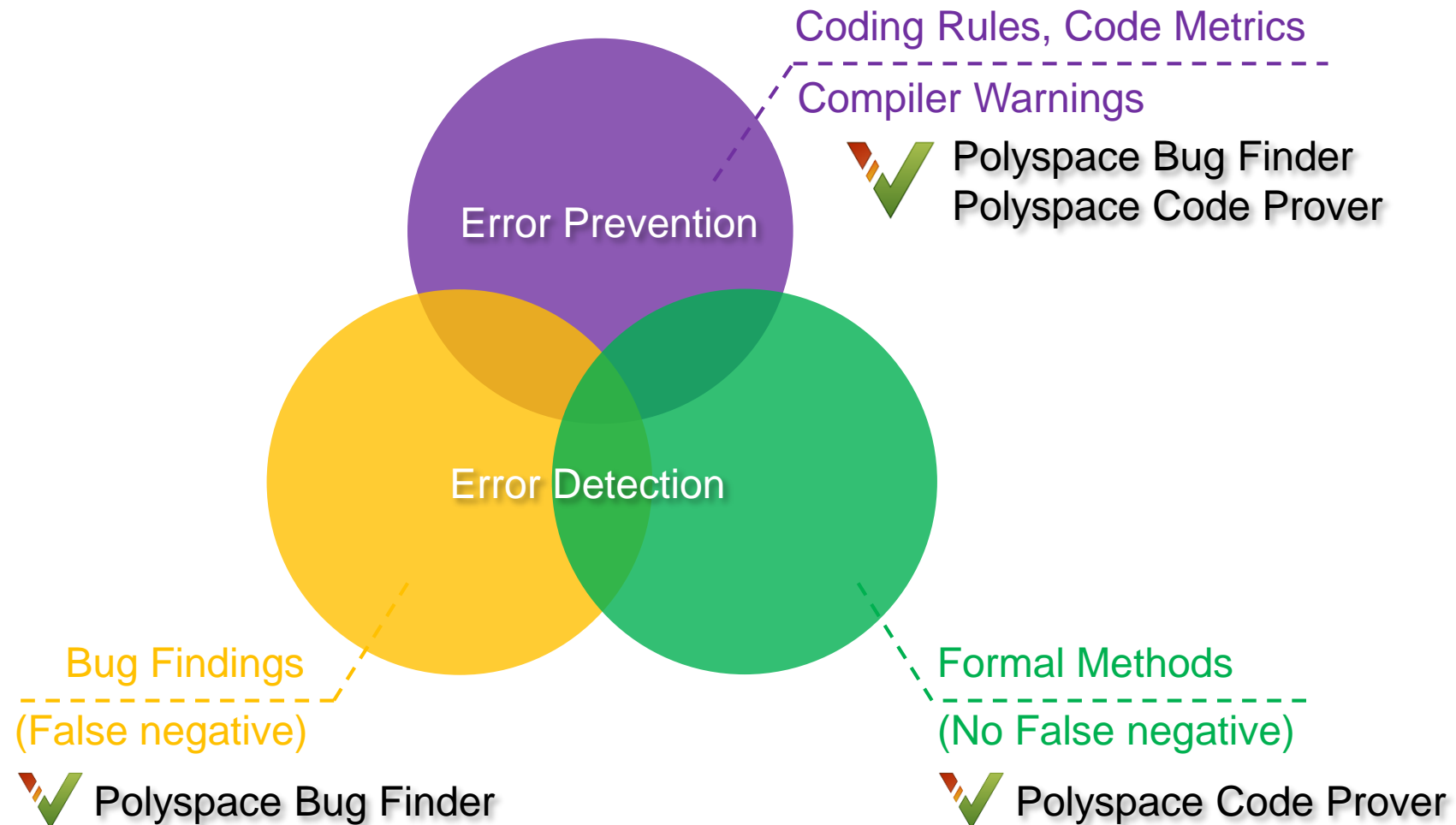
内容

- **Polyspace Overview**
 - **Category of Static analysis**
 - **Use case of Polyspace products**
- Polyspace Hands-on workshop
 - Description of the logic
 - Exercise 1 / 2 / 3 / 4 (Review MISRA, Defect, Code Metrics, Runtime Error)
 - Conclusion
- Q&A

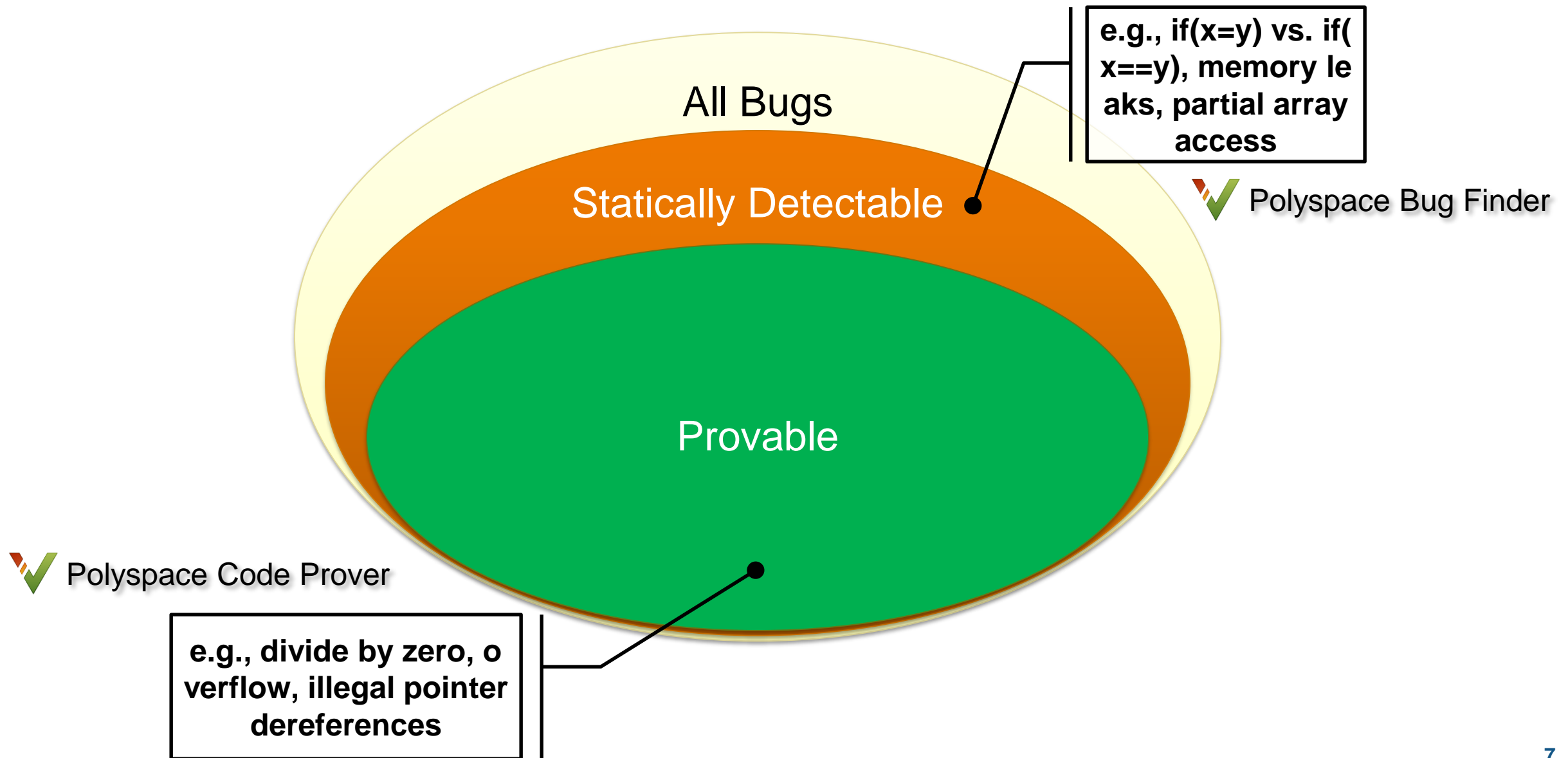
静态分析功能分类



Polyspace产品对应

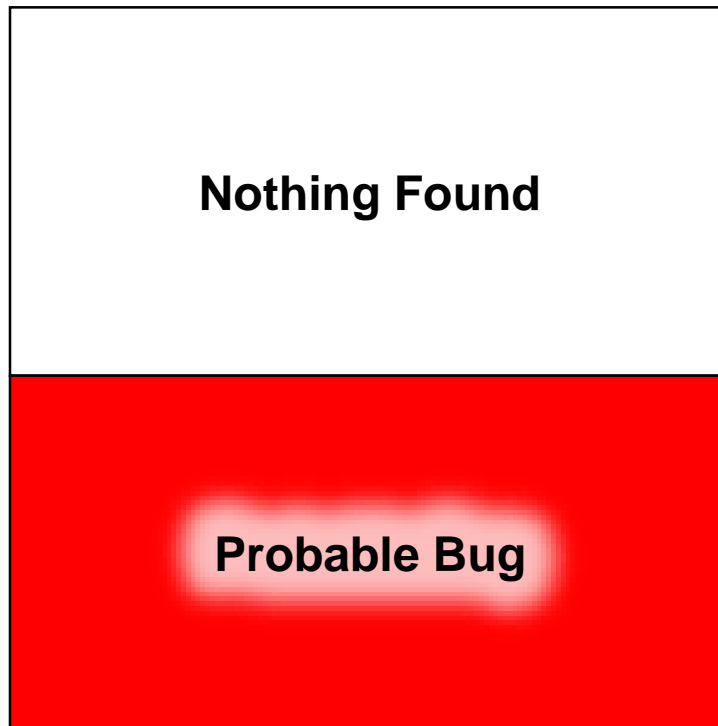


理解差异



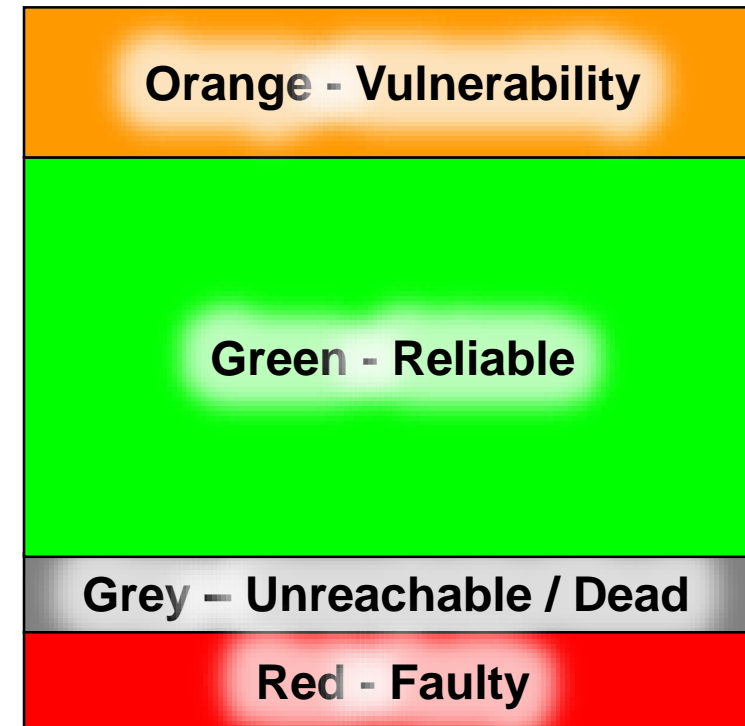
功能对比

Bug Finder



VS.

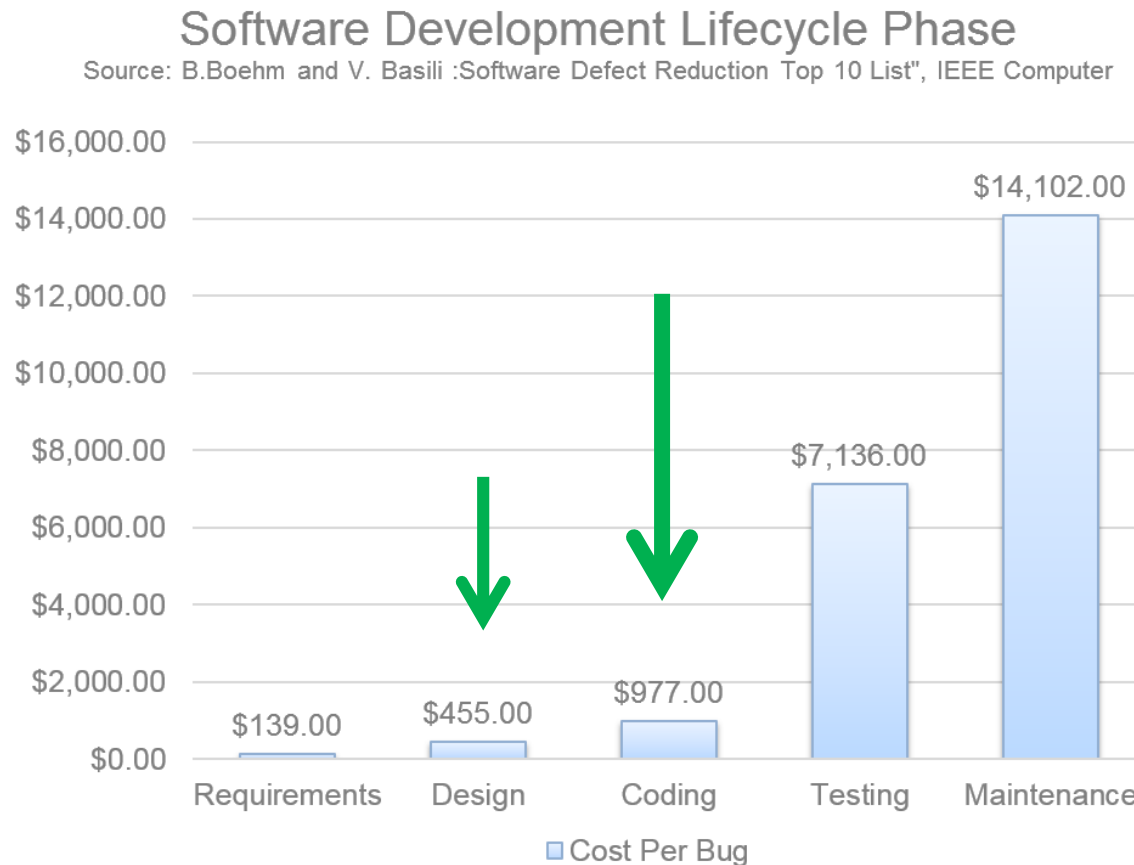
Code Prover



Purple - coding rule violations

早期验证

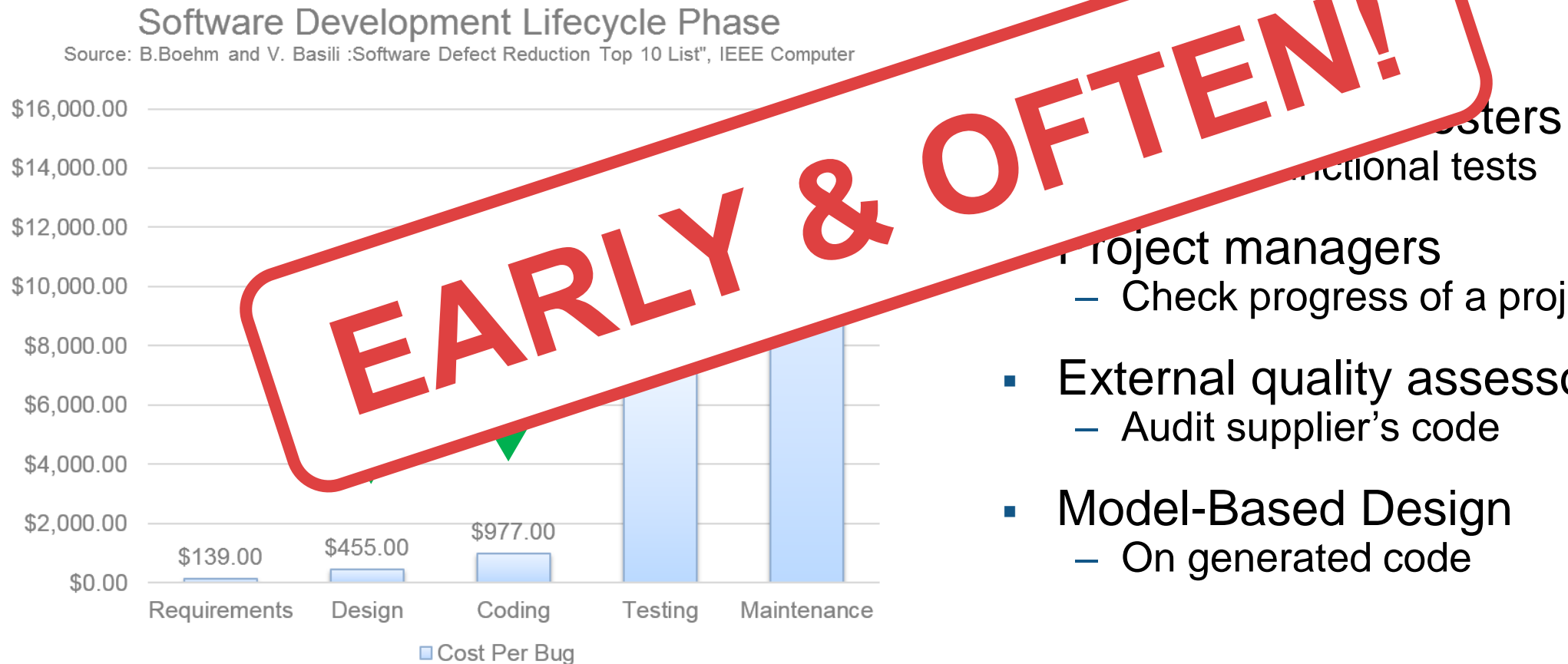
Use Verification tools early in the lifecycle:



- Developers and testers
 - Prior to functional tests
- Project managers
 - Check progress of a project
- External quality assessor
 - Audit supplier's code
- Model-Based Design
 - On generated code


早期验证

Use Verification tools early in the lifecycle:





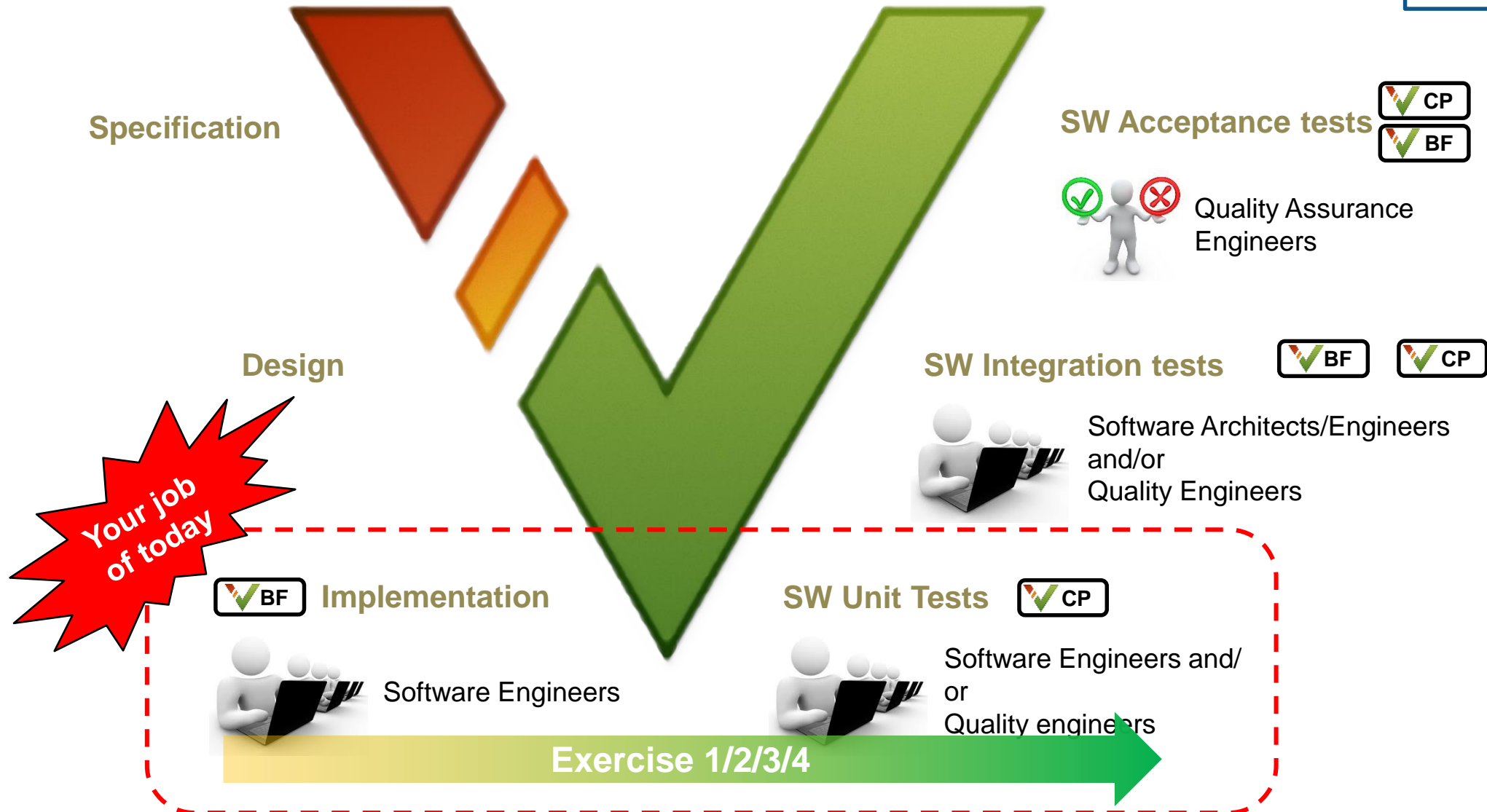
- Project managers
 - Check progress of a project
- External quality assessor
 - Audit supplier's code
- Model-Based Design
 - On generated code

使用角色

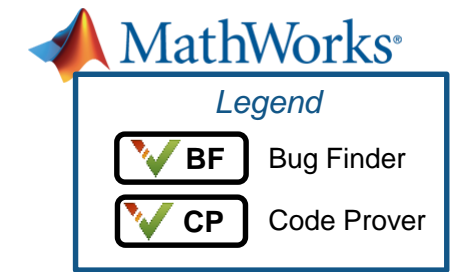
 **MathWorks®**

Legend

	Bug Finder
	Code Prover



参考流程



Specification

SW Acceptance tests



- Measure SW quality
- Quality report generation

Design

SW Integration tests



- Find integration bugs
 - Declaration mismatches
 - Data race on shared variables
 - Global variables usage

 Implementation

- Find local bugs
- Find MISRA violations
- Find “untestable” functions
- Perform Code Reviews

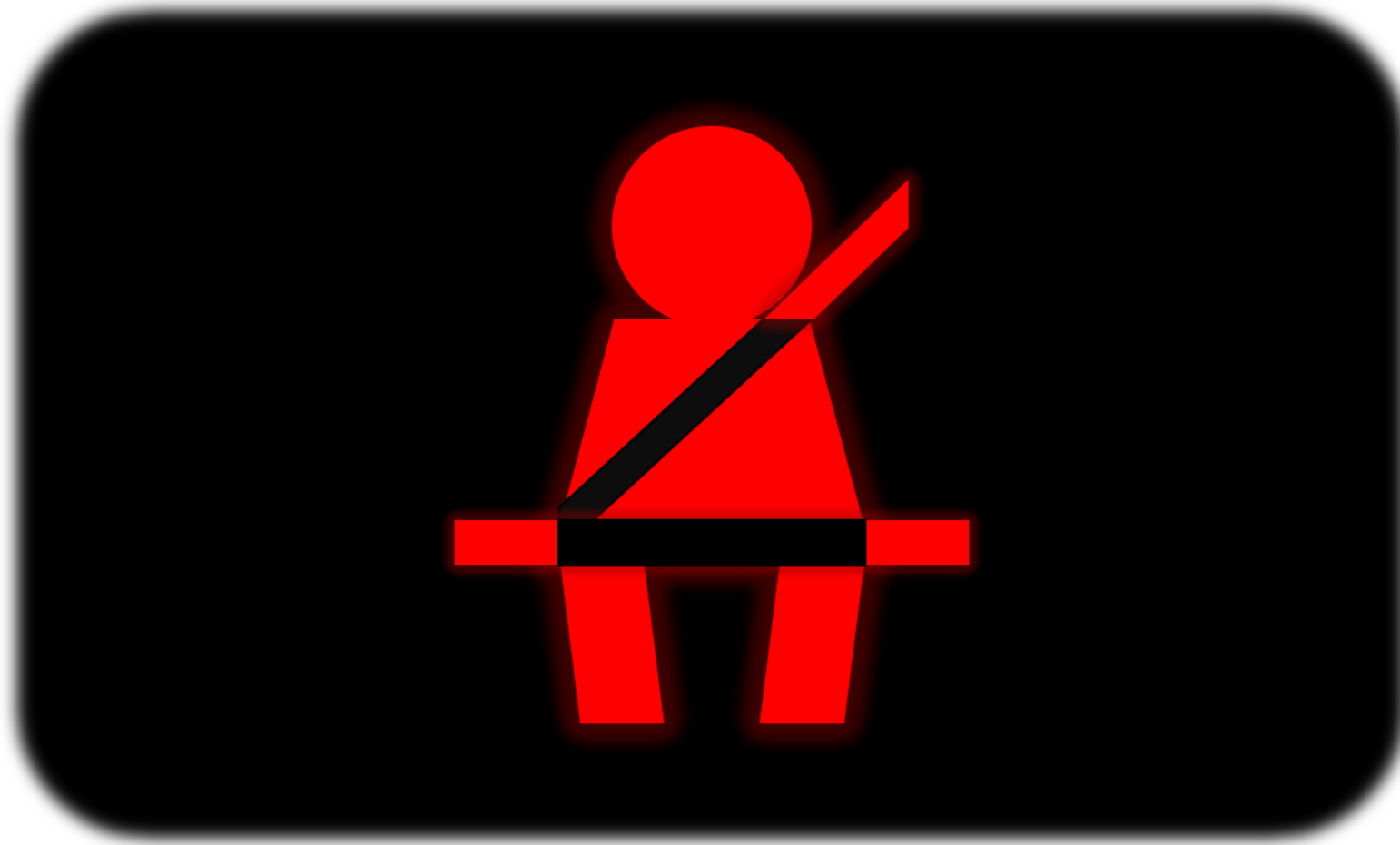
SW Unit Tests 

- Quality gate
 - Find runtime errors / unused code
 - Prove absence of runtime errors on modules
 - Justify MISRA violations

内容

- Polyspace Overview
 - Category of Static analysis
 - Use case of Polyspace products
- **Polyspace Hands-on workshop**
 - **Description of the logic**
 - Exercise 1 / 2 / 3 / 4 (Review MISRA, Defect, Code Metrics, Runtime Error)
 - Conclusion
- Q&A

案例 - 安全带提示功能



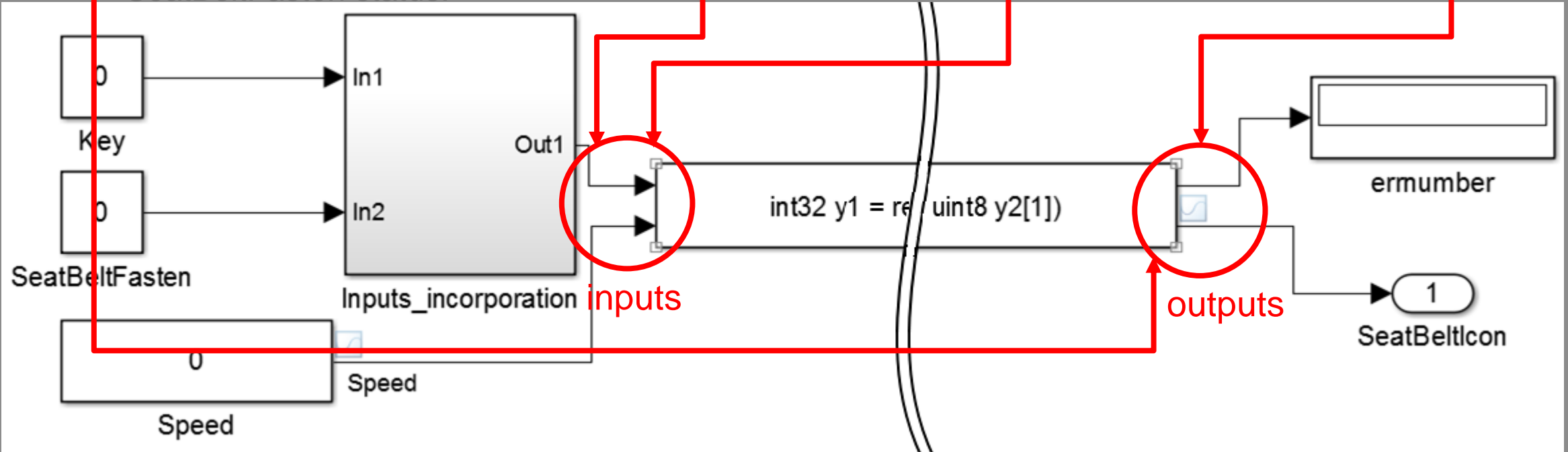
Seatbelt Reminder

安全带提示逻辑 – 软件架构

```
signed int reminder_control (signed short inputs, float *input Curr_speed, unsigned char * output)
```

Inputs description

- First parameter is input to represent Key and SeatBeltFasten. It incorporates Key and SeatBeltFasten status.



- Return value of the function means error status. If there is error, it returns non-zero.

安全带提示逻辑 – 输入输出

```
signed int reminder_control (signed short inputs, float *input Curr speed, unsigned char * output)
```

Inputs description

- First parameter is input to represent Key and SeatBeltFasten. It incorporates Key and SeatBeltFasten status.
- Second parameter is input to represent current Speed. This should be processed by low-pass filter to prevent the signal from splashing in a sudden period.
- First parameter has information for KEY and SeatBeltFasten status.

Input (First parameter)	KEY	SeatBeltFasten
0 (00000000 00000000 ₍₂₎)	OFF	Unfastened
256 (00000001 00000000 ₍₂₎)	ON	Unfastened
512 (00000010 00000000 ₍₂₎)	CRANK	Unfastened

Input (First parameter)	KEY	SeatBeltFasten
0 (00000000 00000001 ₍₂₎)	OFF	Fastened
257 (00000001 00000001 ₍₂₎)	ON	Fastened
513 (00000010 00000001 ₍₂₎)	CRANK	Fastened

Outputs description

- Third parameter is output to represent Seatbelt reminder icon.
- Return value of the function means error status. If there is error, it returns non-zero.

安全带提示逻辑 – 功能描述

■ Functional description

- Sample time : 0.001 s (1000 Hz)
- It should have low-pass filter to remove noise for the Speed value.
- At **Key Off**, the **Seatbelt reminder icon** which is showed in the car dashboard must be switched **off**.
- At **Key On**, the Seatbelt reminder icon status is related to the two inputs Speed and SeatBeltFasten. If the **seat belt is fastened**, **SeatBeltIcon is always off**. If the **seat belt is unfastened**, we consider **two different cases (Low speed and High speed)**.
- At **Key Crank**, the **Seatbelt reminder icon** must be switched **on**.

Speed	KEY	SeatBeltFasten	Seatbelt Reminder icon
< 15 km/h (Low speed)	ON	Unfastened	Always ON
>= 15km/h (High speed)	ON	Unfastened	Flickering every 2 sec.
-	ON	Fastened	Always OFF
-	CRANK	Unfastened / Fastened	Always ON
-	OFF	Unfastened / Fastened	Always OFF

内容

- Polyspace Overview
 - Category of Static analysis
 - Use case of Polyspace products
- **Polyspace Hands-on workshop**
 - Description of the logic
 - **Exercise 1 / 2 / 3 / 4 (Review MISRA, Defect, Code Metrics, Runtime Error)**
 - Conclusion
- Q&A

Exercises概览

- **Exercise 1 : Create Polyspace project and Run it**

1. Simulate the logic of the code and understand problems of it
2. Create Bug Finder project first and Run it

 Bug Finder

- **Exercise 2 : Review MISRA violations and defects**

1. Enable all rules of MISRA C:2012 and most of defects
2. Review all of them and try to remove most of violations and defects
3. Run analysis for the modified code repeatedly

 Bug Finder

Exercises概览

■ Exercise 3 : Review Code Metrics

1. Enable option to calculate Code Metrics
2. Review the result and modify source code to comply with HIS
3. Run analysis for the modified code repeatedly

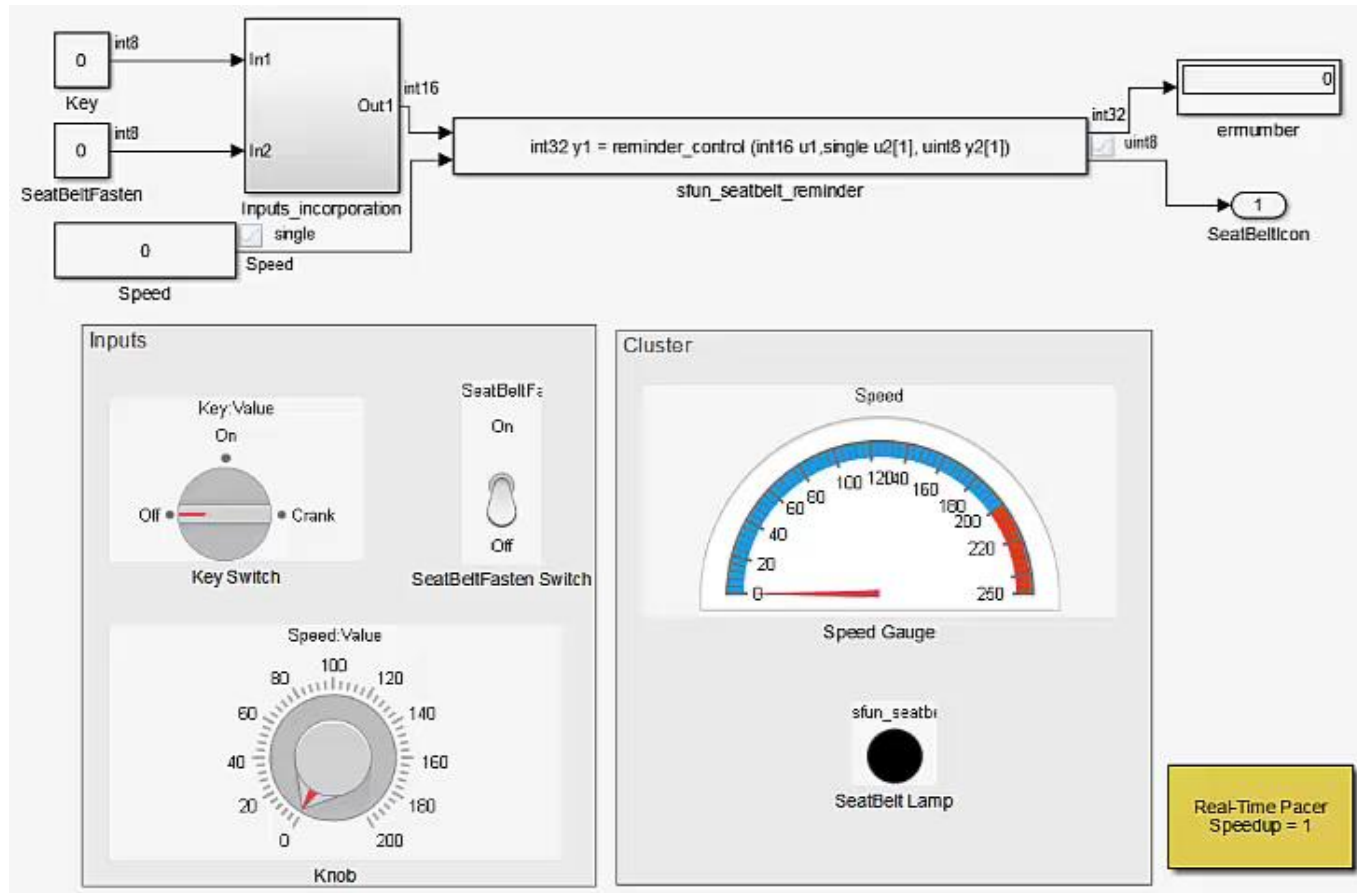
 Bug Finder

■ Exercise 4 : Review Code Prover result

1. Import Bug Finder project into Code Prover and Run it
2. Run verification with Code Prover
3. Review all MISRA violations and RTEs
4. Do anything to make ALL GREEN! (Please don't remove all source code.)

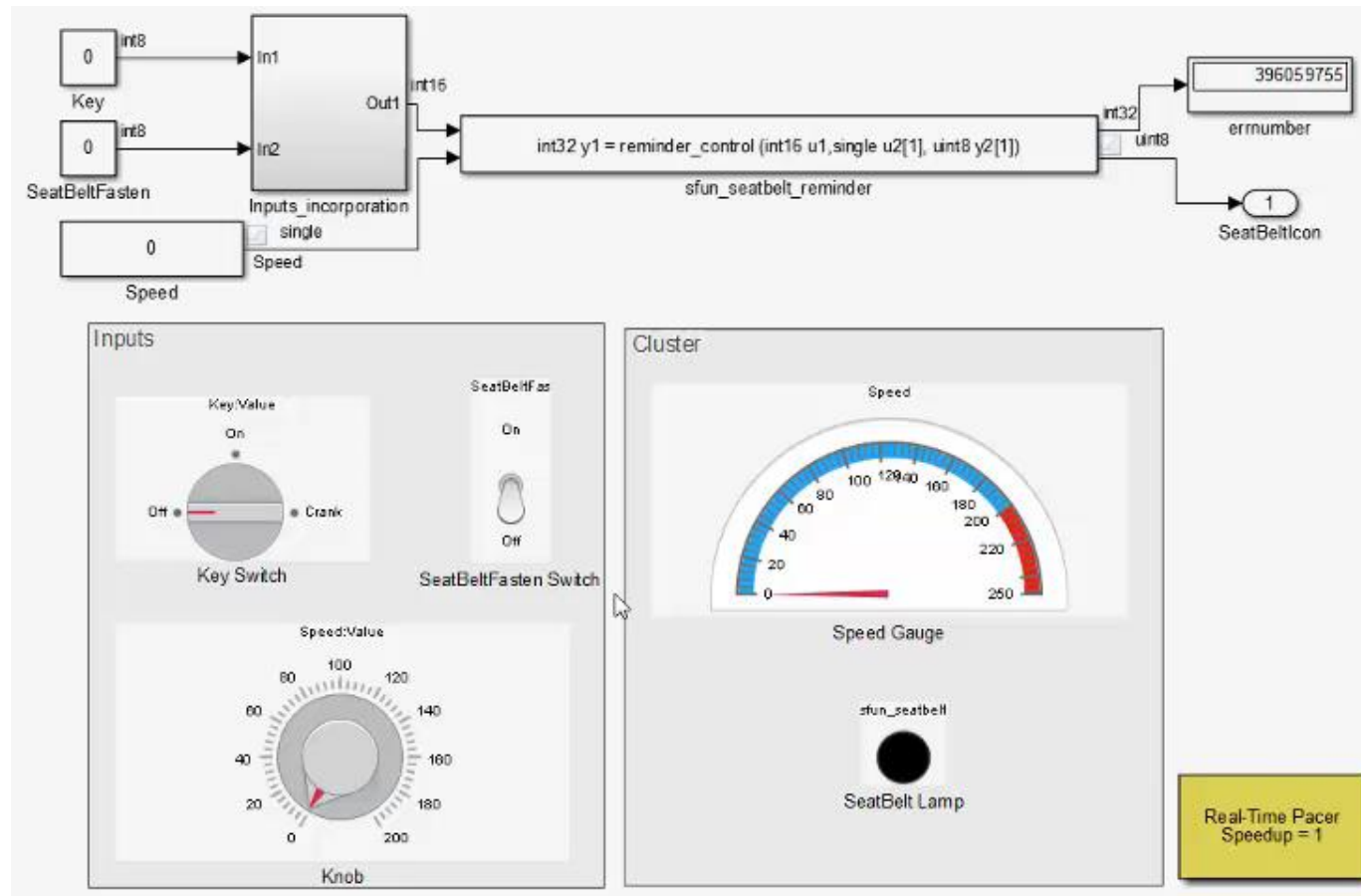
 Code Prover

系统需求理解 – 期望行为



Speed	KEY	SeatBeltFasten	Seatbelt Reminder icon
< 15 km/h (Low speed)	ON	Unfastened	Always ON
>= 15km/h (High speed)	ON	Unfastened	Flickering every 2 sec.
-	ON	Fastened	Always OFF
-	CRA NK	Unfastened / Fastened	Always ON
-	OFF	Unfastened / Fastened	Always OFF

系统需求理解 – 问题发现



Speed	KEY	SeatBeltFasten	Seatbelt Reminder icon
< 15 km/h (Low speed)	ON	Unfastened	Always ON
>= 15km/h (High speed)	ON	Unfastened	Flickering every 2 sec.
-	ON	Fastened	Always OFF
-	CRA NK	Unfastened / Fastened	Always ON
-	OFF	Unfastened / Fastened	Always OFF

- Errnumber when low speed
- Rapid flash when high speed
- ...

代码审查...

```
signed int reminder_control (signed short inputs, float *input_Curr_speed, unsigned char * output)
{
    int errnumber;
    unsigned char input_SeatBeltFasten, input_KEY;
    static int timer_cnt;
    static unsigned char output_SeatBeltIcon = SB_ICON_OFF;
    signed int gFiltered_speed = 0;

    if (((unsigned short)inputs&0x00FF) == 1) {
        input_SeatBeltFasten = 1;
    } else {
        input_SeatBeltFasten = 0;
    }

    switch (((unsigned short)inputs&0xFF00)>>8) {
    case 0:
        input_KEY = 0;
        break;
    case 1:
        input_KEY = 1;
        break;
    case 2:
        input_KEY = 2;
        break;
    }

    errnumber = lowPassFilter_SpeedSignal(&gFiltered_speed, input_Curr_speed);
}
```

```
if (errnumber == 0) {
    if (KEY_OFF == input_KEY) {
        output_SeatBeltIcon = SB_ICON_OFF;
    } else if (KEY_ON == input_KEY) {
        if (SB_UNFASTENED == input_SeatBeltFasten) {
            if (gFiltered_speed < 15) {
                output_SeatBeltIcon = SB_ICON_ON;
            } else {
                if (timer_cnt > COUNT_FOR_BLINK)
                {
                    if (output_SeatBeltIcon == SB_ICON_OFF) {
                        output_SeatBeltIcon = SB_ICON_ON;
                    } else {
                        output_SeatBeltIcon = SB_ICON_OFF;
                    }
                }
                timer_cnt++;
            }
        } else {
            output_SeatBeltIcon = SB_ICON_OFF;
        }
    } else {
        output_SeatBeltIcon = SB_ICON_ON;
    }
}

*output = output_SeatBeltIcon;

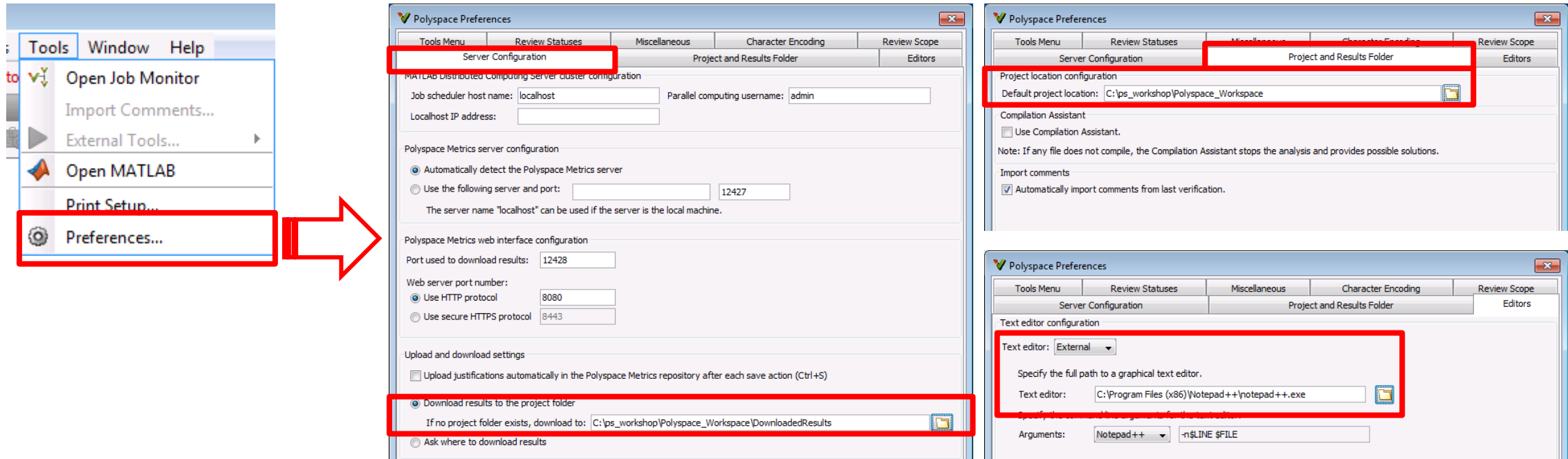
return errnumber;
}
```

Exercise 1

Create project and Run it

- Create project manually
- Create project by using Configuration template

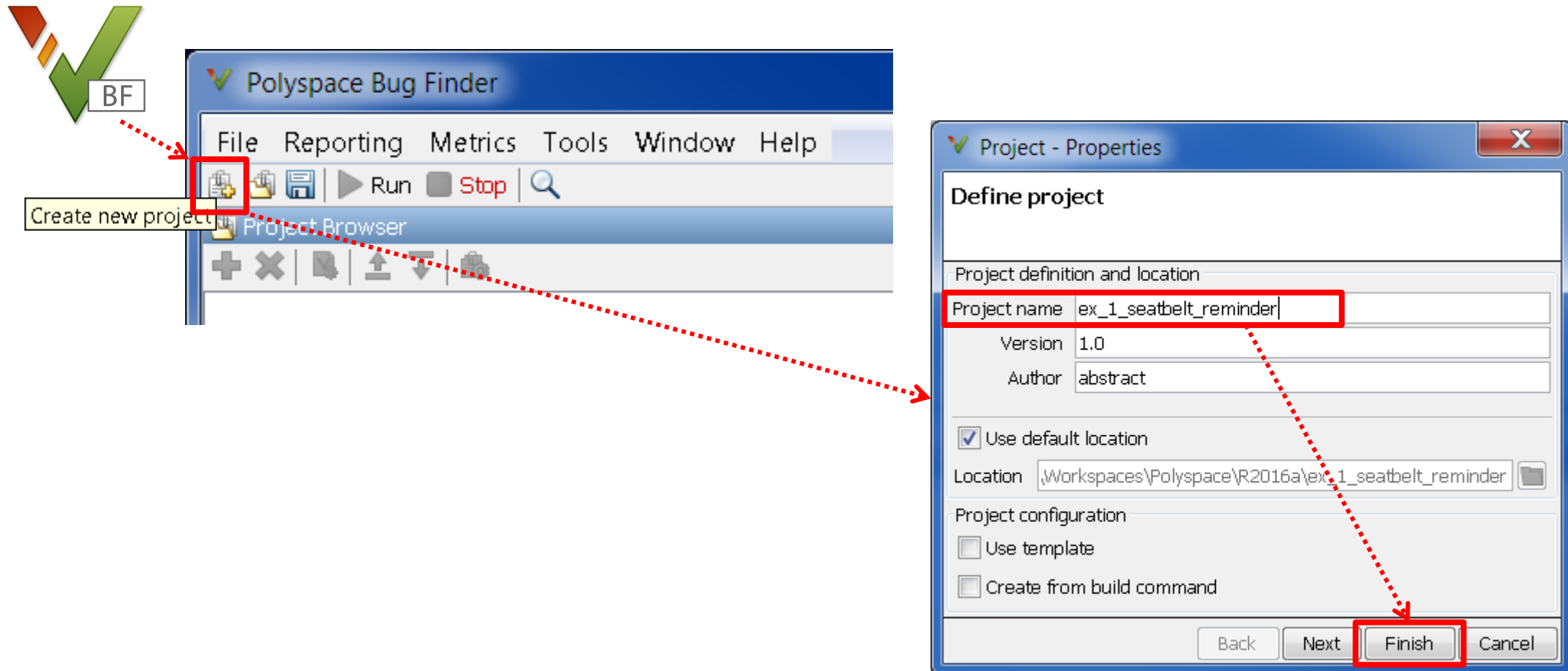
项目配置



- Default project location
 - C:\polyspace_workshop\Polyspace_Workspace
- Download results to
 - C:\polyspace_workshop\Polyspace_Workspace\DownloadResults
- Text editor configuration
 - Notepad++

Exercise 1 - 创建并运行项目

- Create Bug Finder project manually



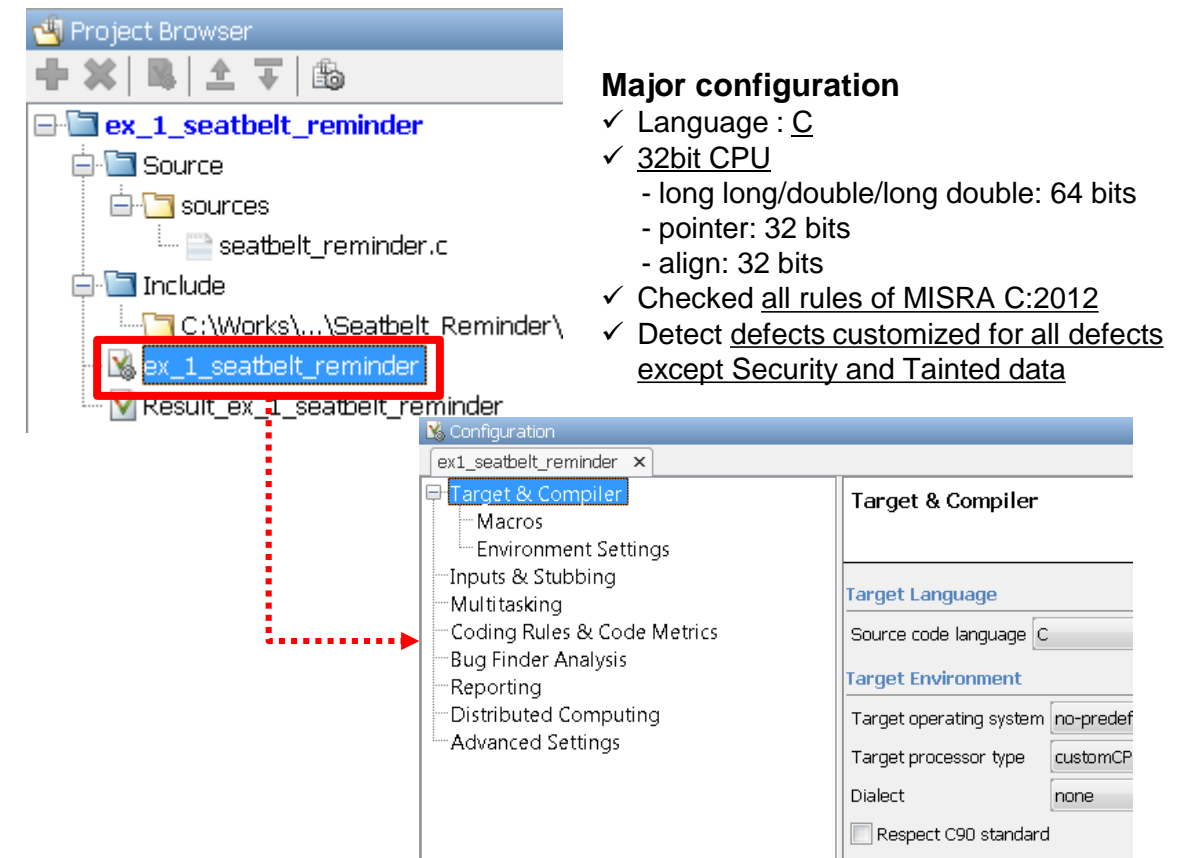
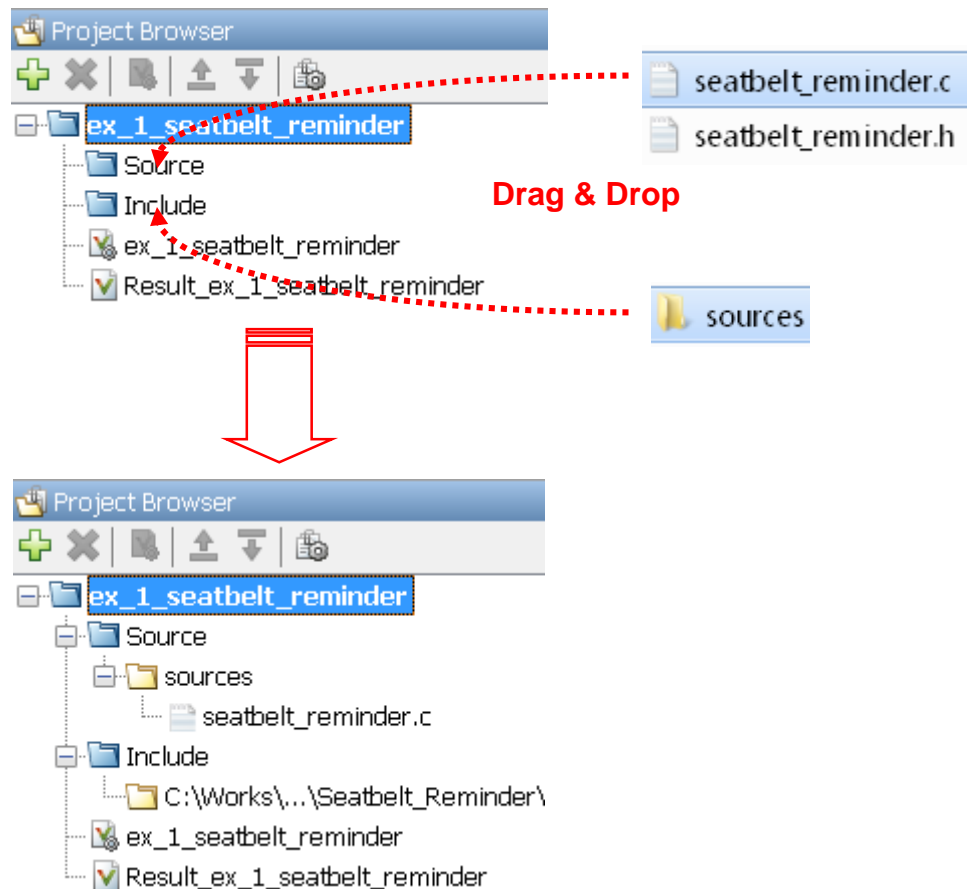
The image shows the Polyspace Bug Finder (BF) interface and the Project Properties dialog box. A green checkmark with 'BF' indicates the project is created. A red dotted arrow points from the 'Create new project' button in the Polyspace Bug Finder toolbar to the 'Project - Properties' dialog box. The dialog box is titled 'Project - Properties' and contains the following fields:

- Project name:** ex_1_seatbelt_reminder
- Version:** 1.0
- Author:** abstract
- Use default location:** ☒
- Location:** \Workspaces\Polyspace\R2016a\ex_1_seatbelt_reminder
- Project configuration:**
 - ☐ Use template
 - ☐ Create from build command

The 'Finish' button is highlighted with a red box, and a red dotted arrow points to it from the 'Project name' field.

Exercise 1 -创建并运行项目

- Add source files in the project, and configure the project



Exercise 1 -创建并运行项目

Target & Compiler

Target Language

Source code language C

Target Environment

Compiler none

Target processor type ---PST Generic--- Edit

☐ Respect C90 standard

Compiler Behavior

☐ Division round down

Enum type definition defined-by-standard

Signed right shift Arithmetical

Generic target options

Enter target name sbr_target

Endianness Little endian

	8bits	16bits	32bits	64bits	
Char	<input checked="" type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input checked="" type="checkbox"/> Signed
Short	<input type="radio"/>	<input checked="" type="radio"/>	<input type="radio"/>	<input type="radio"/>	
Int	<input type="radio"/>	<input checked="" type="radio"/>	<input type="radio"/>	<input type="radio"/>	
Long	<input type="radio"/>	<input type="radio"/>	<input checked="" type="radio"/>	<input type="radio"/>	
Long long	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input checked="" type="radio"/>	
Float	<input type="radio"/>	<input type="radio"/>	<input checked="" type="radio"/>	<input type="radio"/>	
Double/Long double	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input checked="" type="radio"/>	
Pointer	<input type="radio"/>	<input type="radio"/>	<input checked="" type="radio"/>	<input type="radio"/>	
Alignment	<input type="radio"/>	<input type="radio"/>	<input checked="" type="radio"/>	<input type="radio"/>	

Save Remove Cancel

Major configuration

- ✓ Language : C
- ✓ 32bit CPU
 - long long/double/long double: 64 bits
 - pointer: 32 bits
 - align: 32 bits
- ✓ Checked all rules of MISRA C:2012
- ✓ Detect defects customized for all defects except Security and Tainted data

Bug Finder Analysis

☒ Find defects custom

☒ Defects

- ☒ Numerical
- ☒ Static memory
- ☒ Dynamic memory
- ☒ Data flow
- ☒ Resource management
- ☒ Programming
- ☒ Concurrency
- ☐ Security
- ☐ Tainted data
- ☒ Good practice

- Target & Compiler
 - Macros
 - Environment Settings
 - Inputs & Stubbing
 - Multitasking
 - Coding Rules & Code Metrics**
 - Bug Finder Analysis
- Code Prover Verification
 - Verification Assumptions


Coding Rules & Code Metrics

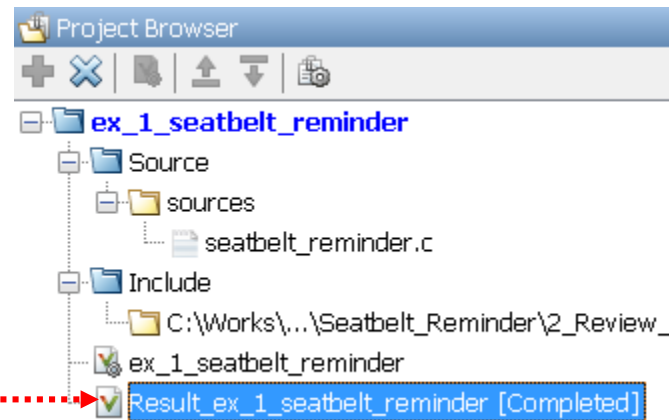
Coding Rules

<input type="checkbox"/> Check MISRA C:2004	required-rules	Edit
<input type="checkbox"/> Check MISRA AC AGC	OBL-rules	Edit
<input checked="" type="checkbox"/> Check MISRA C:2012	all	Edit

Exercise 1 -创建并运行项目

- Run the project in Bug Finder

 Run Bug Finder

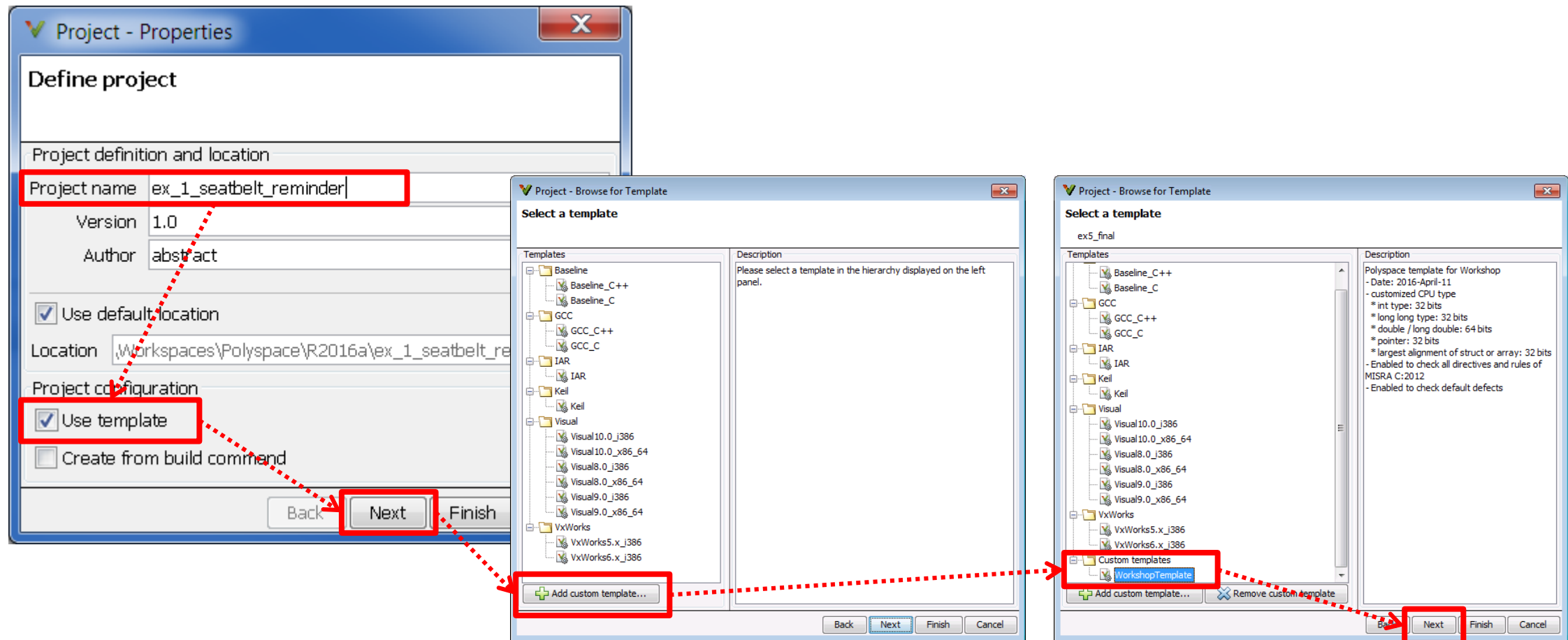


Results List		
All results		
Family	ID	Information
Defect	2	
Data flow	2	
MISRA C:2012	22	
2 Unused code	6	
4 Code design	8	
7 Literals and constants	1	
8 Declarations and definitions	3	
9 Initialization	1	
10 The essential type model	1	
14 Control statement expressions	1	
16 Switch statements	1	

Tip #1 – 项目模板



- Create Bug Finder project by using Configuration Template

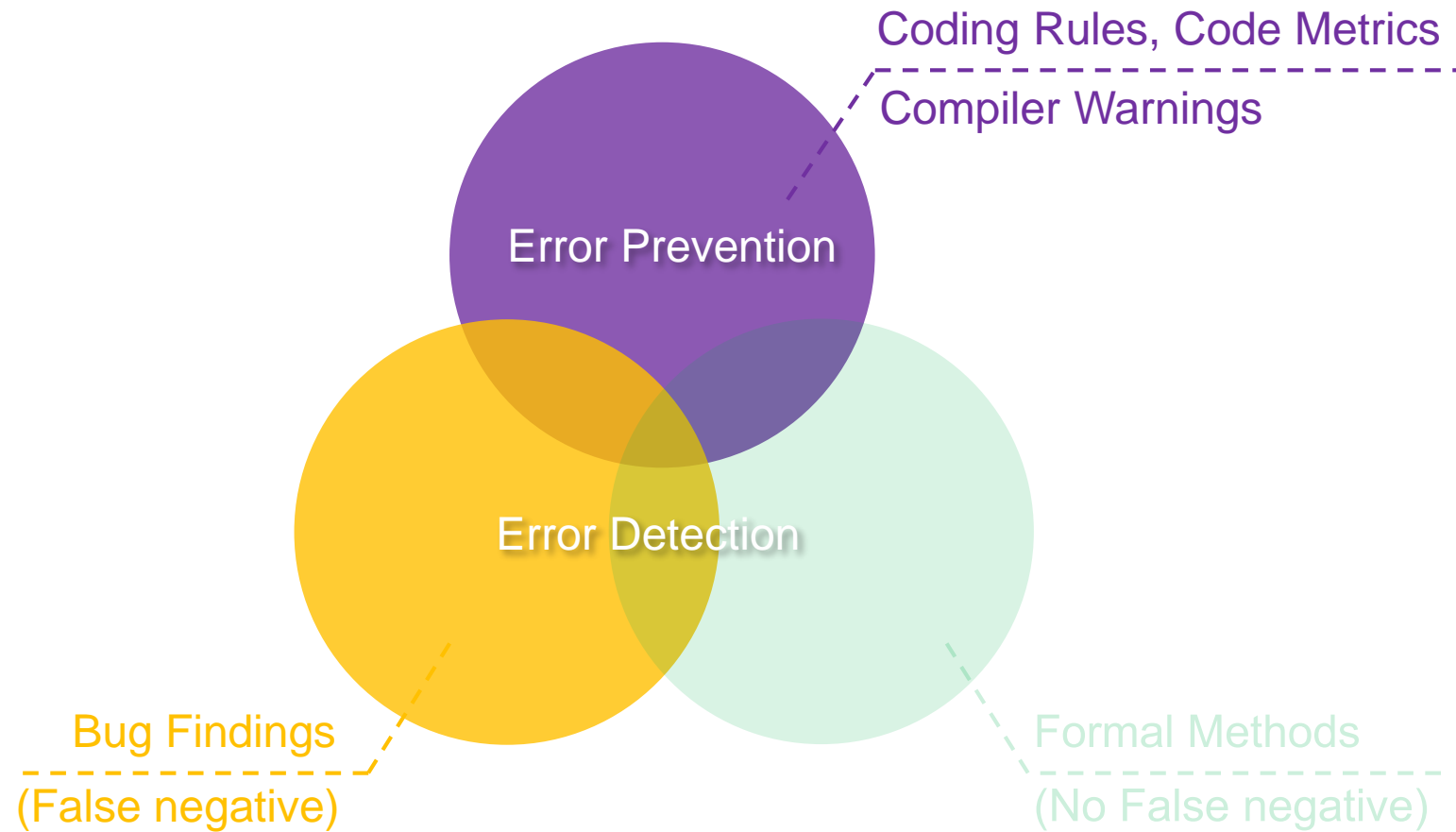


Exercise 2

Remove MISRA Violations and Defects

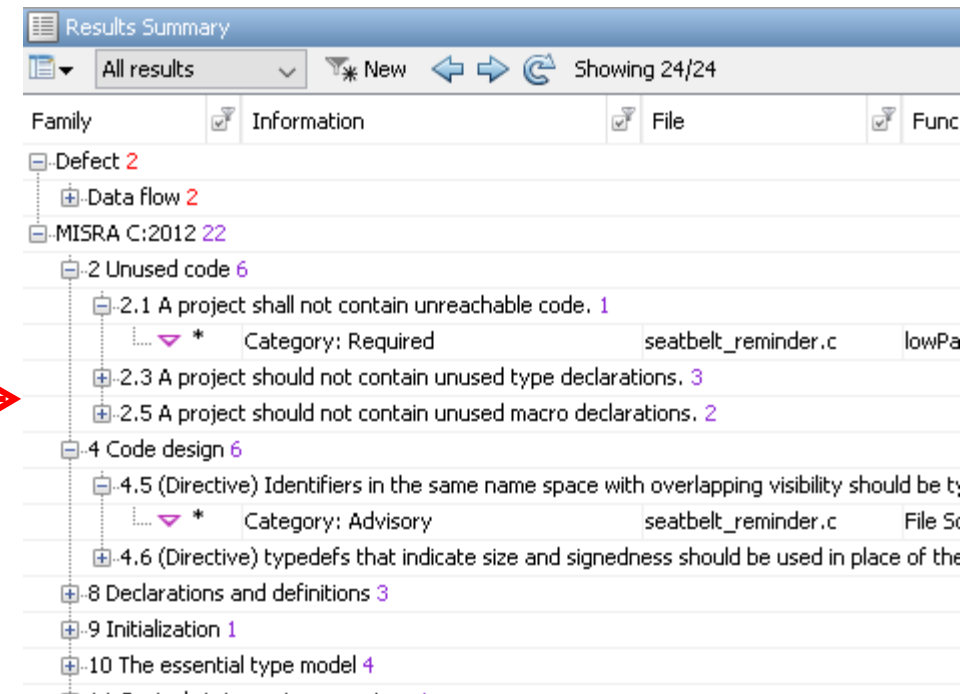
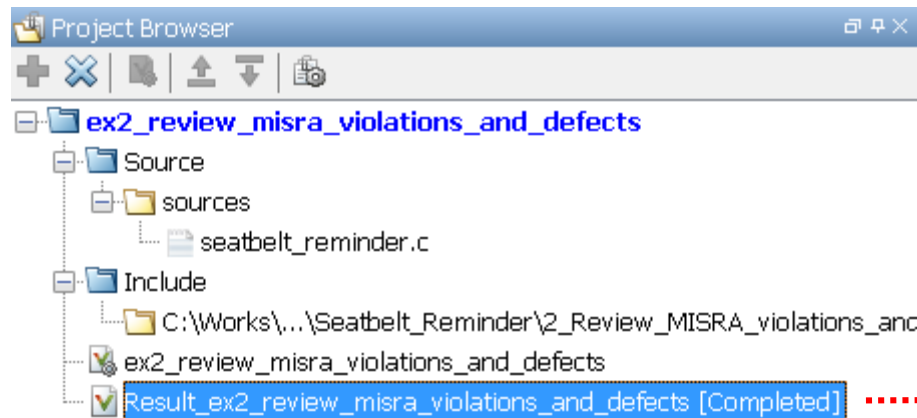
- Create new project file for this exercise
- Remove/Justify MISRA C:2012 Rule violations

Exercise 2 & 3 – 覆盖内容




Exercise 2 – MISRA检查和软件缺陷查找


- Review all MISRA violations with Bug Finder
 - Create new project with source files in *2_Review_MISRA_violations_and_Defects*
 - Review MISRA violations first




Tip #2 – 获取帮助



- Click  to see Contextual Help
 - Modify the code as you wish to remove this violation


▼ **MISRA C:2012 8.2 (Required)** 
Function types shall be in prototype form with named parameters

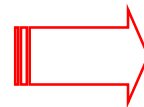
! **Dead code (Impact: Low)** 
If-condition always evaluates to true.
Dead branch from line 57 to line 59.

Target Environment

Target operating system	no-predefined-C
Target processor type	customCPU

Target processor type (-target)
Select the target machine processor type.

 More Help



MISRA C:2012 Rule 8.2

Function types shall be in prototype form with named parameters

Description

Rule Definition
Function types shall be in prototype form with named parameters

Rationale
The mismatch between expected and actual return types. This rule also requires that parameter names provided between a declaration and a function definition match.

Polyspace Specification
Polyspace also checks for the following conditions:

Examples

► Dead Code from if-condition

Dead code

Code does not execute.

Description

Dead code occurs when the defect excludes:

- Code deactivated as #if 0.
- Unreachable code or return.
- Useless if, which is always true or false.

Target processor type (-target)

Specify size of data types and endianness by using predefined target processor list

Description

Specify the target processor type. This option is available on the **Target & Compiler** node in the **Configuration** pane.

This determines the size of fundamental data types and the endianness of the target machine. You can analyze code intended for an unlisted processor type using one of the other processor types, if they share common data properties.

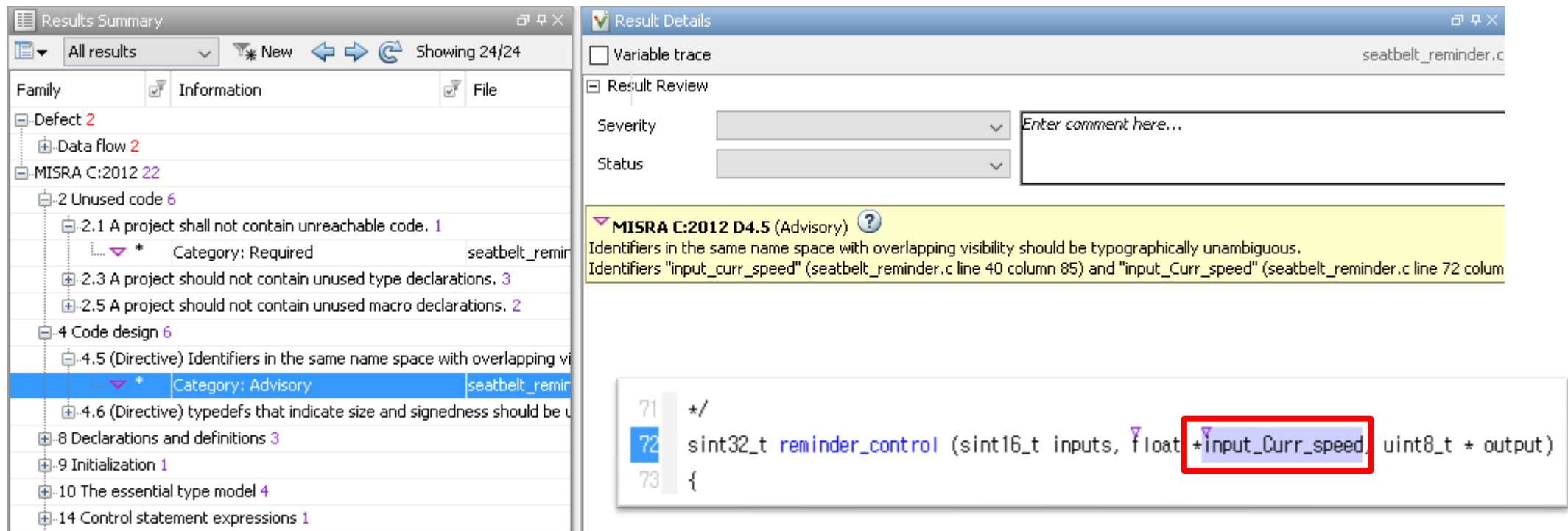
Settings:

Default: i386

The table below shows the size of each fundamental data type that Polyspace considers. For some targets, you can modify the default size by clicking the **Edit** button to the right of the **Target processor type** drop-down menu. The optional values for those targets are shown in [brackets] in the table.

Exercise 2 – MISRA检查和软件缺陷查找

- Review an easy MISRA violation and fix it
 - Check violation related to Directive 4.5 and Directive 4.6

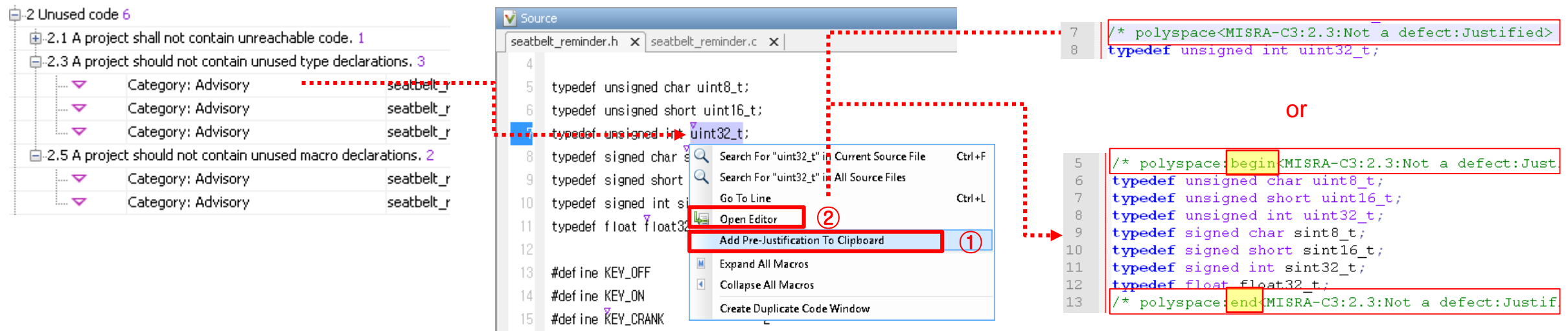


The screenshot displays the MATLAB IDE interface with two main panels. The left panel, titled 'Results Summary', shows a tree view of analysis results. Under 'MISRA C:2012 22', the '4.5 (Directive) Identifiers in the same name space with overlapping visibility' rule is highlighted. The right panel, titled 'Result Details', shows the details for this violation. It includes a 'Severity' dropdown set to 'Advisory' and a 'Status' dropdown set to 'New'. The description states: 'Identifiers in the same name space with overlapping visibility should be typographically unambiguous. Identifiers "input_curr_speed" (seatbelt_reminder.c line 40 column 85) and "input_Curr_speed" (seatbelt_reminder.c line 72 column 85)'. Below this, a code snippet from 'seatbelt_reminder.c' is shown. Line 72 is highlighted, showing the declaration: `sint32_t reminder_control (sint16_t inputs, float *input_Curr_speed, uint8_t * output)`. The identifier `*input_Curr_speed` is enclosed in a red box, indicating the violation.

- Modify the code as you wish to remove this violation

Exercise 2 – MISRA检查和软件缺陷查找

- Review some MISRA violations, and justify it by using pre-justification if necessary
 - Justify violations related to Rule 2.3 and Rule 2.5.

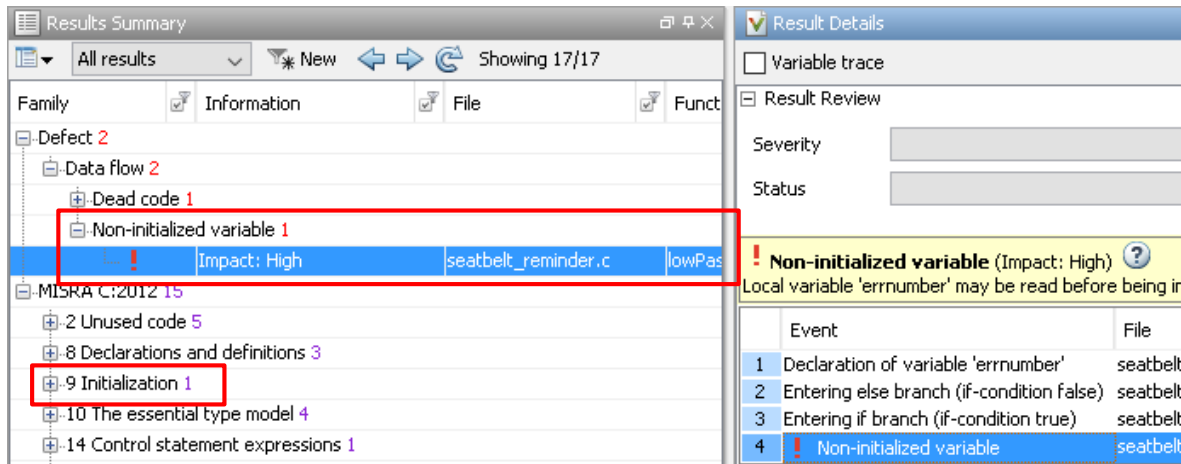


The screenshot illustrates the process of justifying MISRA violations in the Polyspace client. On the left, the 'Unused code' pane shows violations for Rule 2.3 (unused type declarations) and Rule 2.5 (unused macro declarations). The central 'Source' window displays the C code for 'seatbelt_reminder.c', with a context menu open over a line of code. The menu options 'Add Pre-Justification To Clipboard' (marked with a red circle 1) and 'Open Editor' (marked with a red circle 2) are highlighted. On the right, two examples of justified code are shown, separated by the word 'OR'. The first example shows a line of code with a pre-justification comment: `/* polyspace<MISRA-C3:2.3:Not a defect:Justified>` followed by the code line. The second example shows a code block enclosed in `/* polyspace:begin<MISRA-C3:2.3:Not a defect:Just` and `/* polyspace:end<MISRA-C3:2.3:Not a defect:Justif` comments.

- Justify violations through Polyspace client GUI or comments
- Add comments in source code to justify violations.
 - You can use pre-justification for a line or a section.

Exercise 2 – MISRA检查和软件缺陷查找

- **Review all MISRA violations**
 - You may modify source code to remove violations and defects!
 - Refer documentation through Contextual Help
- **Review all defects in the same way**
 - If you remove a defect, MISRA violation can be disappeared.



Results Summary: All results (Showing 17/17)

Family: Information File Funct

Defect 2

- Data flow 2
 - Dead code 1
 - Non-initialized variable 1 (Impact: High) seatbelt_reminder.c lowPas

MISRA C:2012 15

- 2 Unused code 5
- 8 Declarations and definitions 3
- 9 Initialization 1
- 10 The essential type model 4
- 14 Control statement expressions 1

Result Details

Variable trace

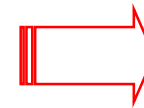
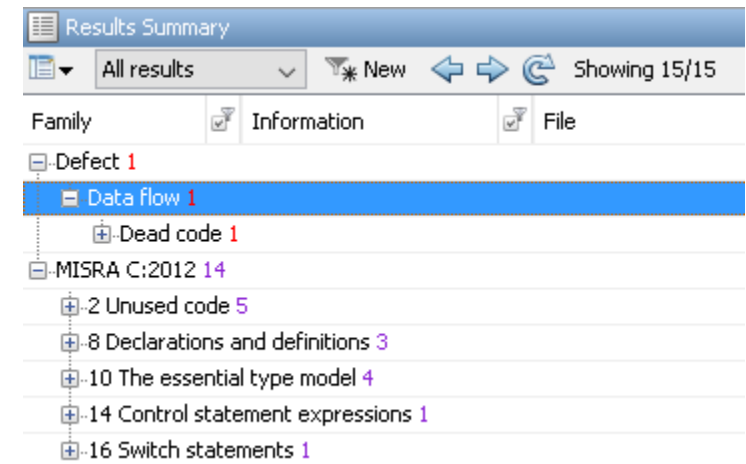
Result Review

Severity

Status

Non-initialized variable (Impact: High) ?
Local variable 'errnumber' may be read before being initialized

Event	File
1 Declaration of variable 'errnumber'	seatbelt
2 Entering else branch (if-condition false)	seatbelt
3 Entering if branch (if-condition true)	seatbelt
4 Non-initialized variable	seatbelt

Results Summary: All results (Showing 15/15)

Family: Information File

Defect 1

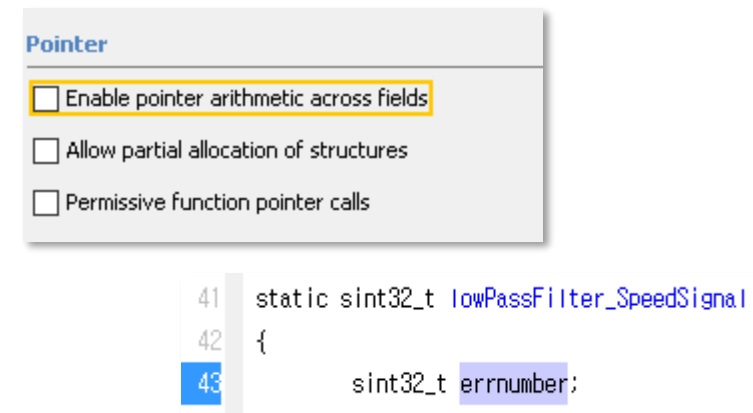
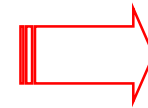
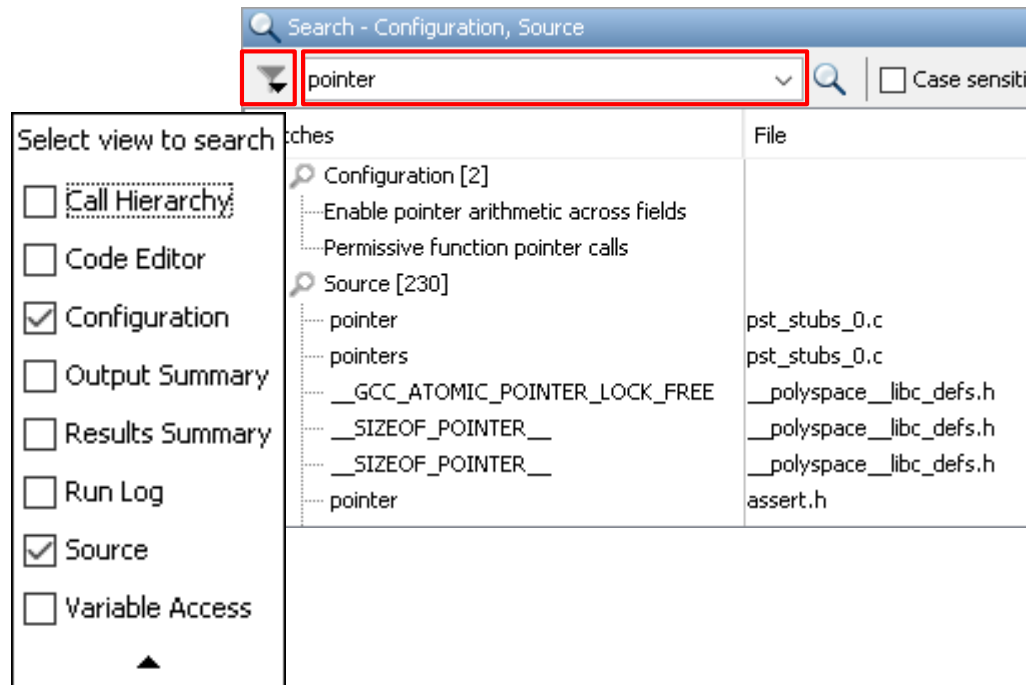
- Data flow 1
 - Dead code 1

MISRA C:2012 14

- 2 Unused code 5
- 8 Declarations and definitions 3
- 10 The essential type model 4
- 14 Control statement expressions 1
- 16 Switch statements 1

Tip #3 – 查找功能

- Search whatever you want in Polyspace
 - Ctrl + F** enables you may search anything in Polyspace project
 - When you select a item, it highlights the location of the item



Exercise 2 – MISRA检查和软件缺陷查找

- Example answer - MISRA C:2012 - Rule 8.2

```
extern sint32_t reminder_control (sint16_t , float *, uint8_t *);  
extern sint32_t reminder_control (sint16_t inputs, float32_t *curr_speed, uint8_t * output);
```

- Example answer - MISRA C:2012 - Rule 8.7

```
float32_t gPrev_speed;  
static float32_t gPrev_speed;
```

- Example answer - MISRA C:2012 - Rule 8.13

```
static sint32_t lowPassFilter_SpeedSignal (sint32_t * output_Filtered_speed, float32_t * input_curr_speed)  
static sint32_t lowPassFilter_SpeedSignal (sint32_t * output_Filtered_speed, const float32_t * input_curr_speed)
```

Exercise 2 – MISRA检查和软件缺陷查找

- Example answer - MISRA C:2012 - Rule 9.1 and Non-initialized variable

```
} else {  
    if (tmp_filtered_speed < 0.5F) {  
        *output_Filtered_speed = 0;  
    } else {  
        *output_Filtered_speed = 0;  
    }  
}  
  
gPrev_speed = tmp_filtered_speed;  
return errnumber;  
}
```




```
} else {  
    if (tmp_filtered_speed < 0.5F) {  
        *output_Filtered_speed = 0;  
    } else {  
        *output_Filtered_speed = 0;  
    }  
    errnumber = 0;  
}  
  
gPrev_speed = tmp_filtered_speed;  
return errnumber;  
}
```


Exercise 2 – MISRA检查和软件缺陷查找

- Example answer - MISRA C:2012 - Rule 16.4

```
switch (((uint16_t)inputs&0xFF00)>>8) {  
case 0:  
    input_KEY = 0;  
    break;  
case 1:  
    input_KEY = 1;  
    break;  
case 2:  
    input_KEY = 2;  
    break;  
}
```



```
switch (((uint16_t)inputs&0xFF00)>>8) {  
case 0:  
    input_KEY = 0;  
    break;  
case 1:  
    input_KEY = 1;  
    break;  
case 2:  
    input_KEY = 2;  
    break;  
default:  
    input_KEY = 3;  
    break;  
}
```

Exercise 2 – MISRA检查和软件缺陷查找

- Example answer - MISRA C:2012 - Rule 10.1 and 10.4

```
if (((uint16_t)inputs&0x00FF) == 1) {
    input_SeatBeltFasten = 1;
} else {
    input_SeatBeltFasten = 0;
}

switch (((uint16_t)inputs&0xFF00)>>8) {
```

```
if (((uint16_t)inputs&0x00FFU) == 1) {
    input_SeatBeltFasten = 1;
} else {
    input_SeatBeltFasten = 0;
}

switch (((uint16_t)inputs&0xFF00U)>>8) {
```

- Example answer - MISRA C:2012 - Rule 16.4 and Dead Code

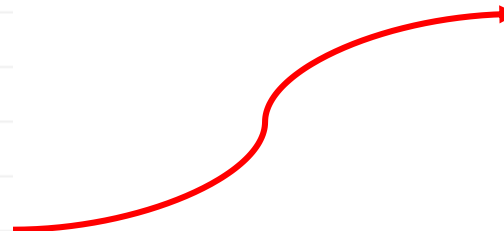
```
} else {
    if (tmp_filtered_speed < 0.5F) {
        *output_Filtered_speed = 0;
    } else {
        *output_Filtered_speed = 0;
    }
    errnumber = 0;
}
```

```
} else {
    *output_Filtered_speed = 0;
    errnumber = 0;
}
```

Exercise 2 – 结果对比

Results List		
All results	New	Show
Family	ID	Information
Defect 2		
+ Data flow 2		
MISRA C:2012 22		
+ 2 Unused code 6		
+ 4 Code design 8		
+ 7 Literals and constants 1		
+ 8 Declarations and definitions 3		
+ 9 Initialization 1		
+ 10 The essential type model 1		
+ 14 Control statement expressions 1		
+ 16 Switch statements 1		

Results List		
All results	New	Show
Family	ID	Information
MISRA C:2012 6		
+ 2 Unused code 3		
+ 4 Code design 2		
+ 8 Declarations and definitions 1		



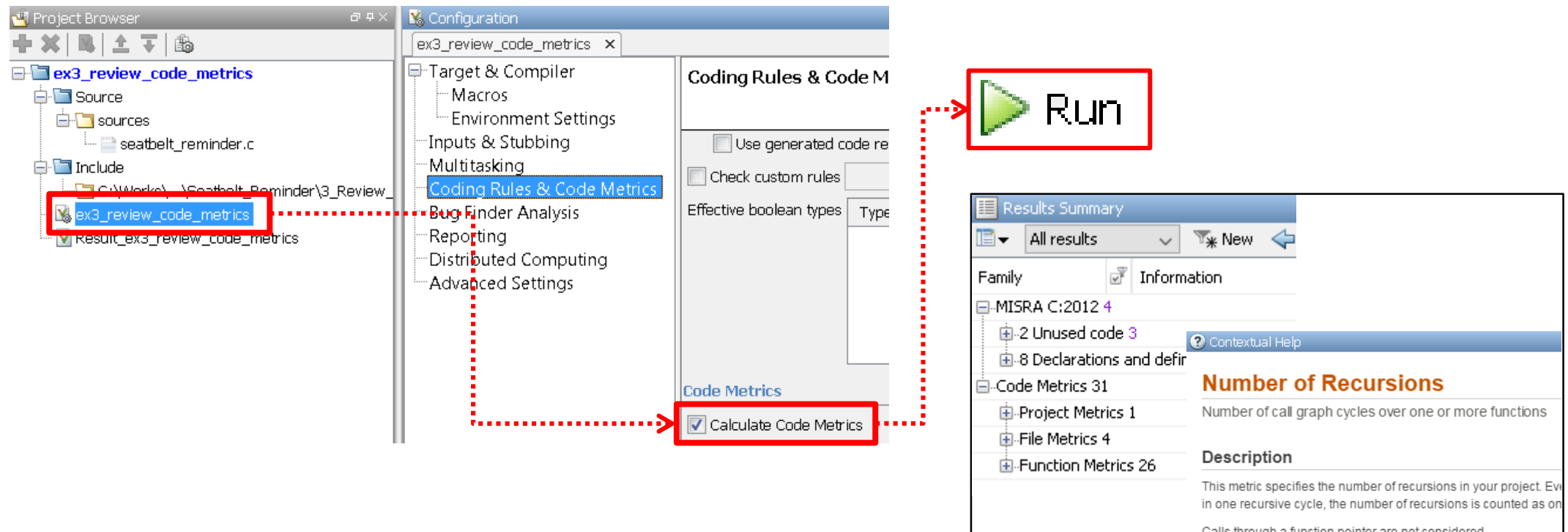
Exercise 3

Review Code Metrics and Increase/Decrease it

- Create new project file for this exercise
- Comply with HIS metrics

Exercise 3 - 代码度量

- Review all Code Metrics and refer documentation through Contextual Help
 - Create new project with source files in *3_Review_Code_Metrics*
 - Enable option to calculate Code Metrics and Run analysis

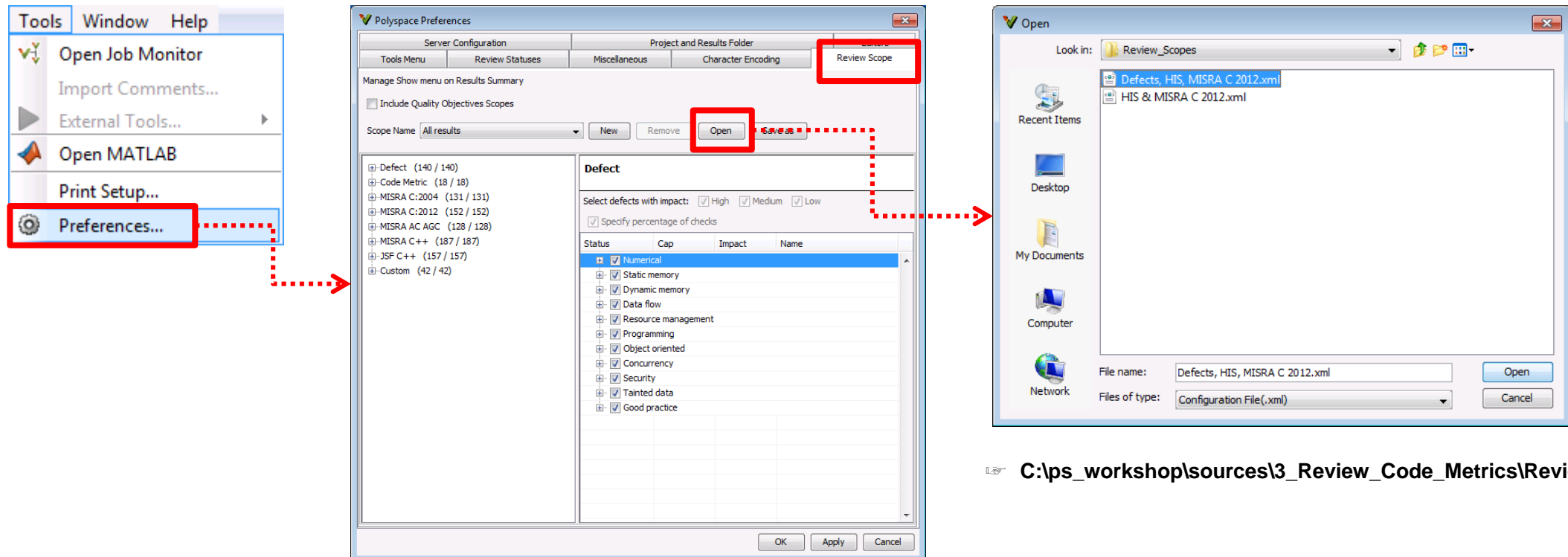


The screenshot illustrates the process of configuring and running code metrics analysis in MATLAB. It is divided into three main sections:

- Project Browser:** Shows a project named `ex3_review_code_metrics` with subfolders for `Source` (containing `sources` and `seatbelt_reminder.c`) and `Include`. The project name is highlighted with a red box.
- Configuration:** The `ex3_review_code_metrics` configuration window is open. In the left sidebar, `Coding Rules & Code Metrics` is selected and highlighted with a blue box. In the right pane, under the `Code Metrics` section, the checkbox `Calculate Code Metrics` is checked and highlighted with a red box. A red dotted arrow points from this checkbox to the `Run` button.
- Run Button:** A green play button icon with the text `Run` is highlighted with a red box. A red dotted arrow points from the `Calculate Code Metrics` checkbox to this button.
- Results Summary:** A window showing the analysis results. It lists metrics such as `MISRA C:2012 4`, `2 Unused code 3`, `8 Declarations and defin`, `Code Metrics 31`, `Project Metrics 1`, `File Metrics 4`, and `Function Metrics 26`. A `Contextual Help` button is visible. The `Number of Recursions` section is expanded, showing a description: "This metric specifies the number of recursions in your project. Even in one recursive cycle, the number of recursions is counted as one. Calls through a function pointer are not considered."

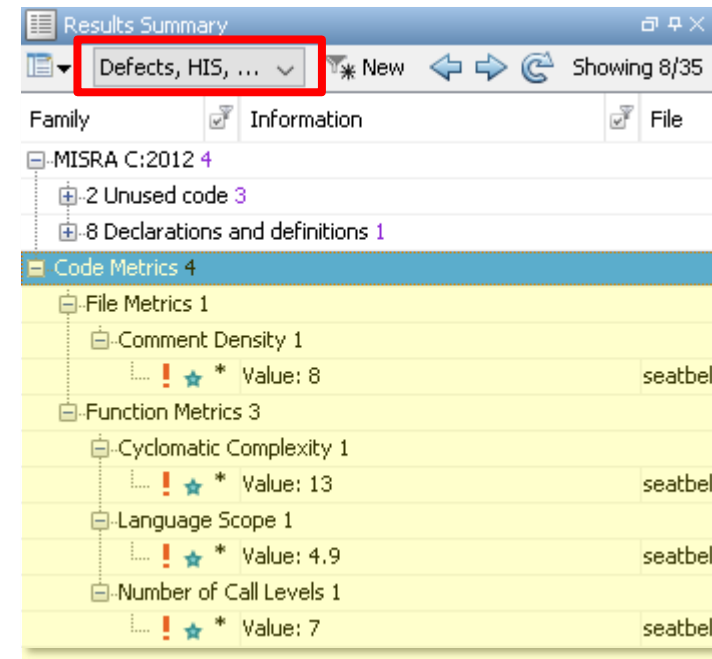
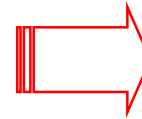
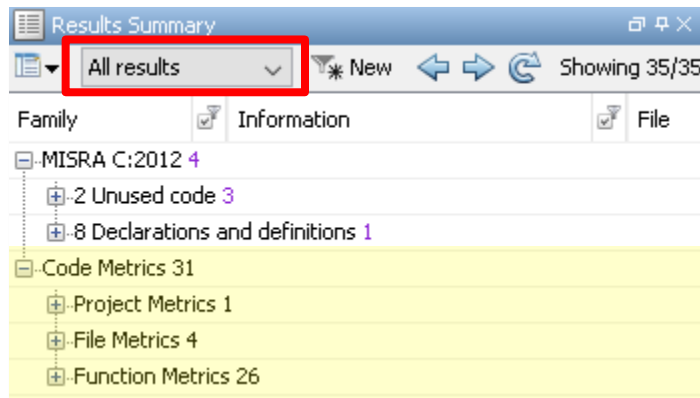
Exercise 3 - 代码度量

- Check which metrics cannot comply with [HIS](#) metrics
 - Apply review scope for HIS
 - Open user-defined review scope to see MISRA violations and Code Metrics together.



Exercise 3 - 代码度量

- Apply user-defined review scope for Defects, MISRA and HIS metrics
 - You may make a new review scope to see MISRA violations and Code Metrics together.



- How do you make it to comply with HIS metrics?

Exercise 3 - 代码度量

- Consider how to decrease/increase code metrics
 - Let's see Cyclomatic Complexity and Number of Call Levels

Cyclomatic Complexity

Number of linearly independent paths through source code [expand all in page](#)

Description

This metric specifies the number of linearly independent paths through the source code.

To calculate this metric, add 1 to the number of decision points in your code. A decision point is a statement that causes your program to branch into two paths. For example, at an if statement, your program can either enter the if branch or not.

The recommended upper limit for this metric is 10. If the cyclomatic complexity is high, the code is both difficult to read and can cause more orange checks. Therefore, try to limit the value of this metric.

Number of Call Levels

Maximum depth of nesting of control flow structures [expand all in page](#)

Description

This metric specifies the maximum nesting depth of control flow statements such as if, switch, for, or while in a function. A function without control-flow statements has a call level 1.

The recommended upper limit for this metric is 4. For better readability of your code, try to enforce an upper limit for this metric.

- Recommended upper limit : 10 for Cyclomatic Complexity, 4 for Number of Call Levels
- If you want to decrease calculated value,
 - ✓ Decrease the number of decision points such as if condition, switch/case and for/while loop.

Exercise 3 - 代码度量

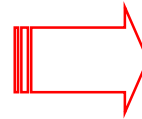
- Let's see example code to improve code metrics - 1
 - This is not the best answer but an example answer.
 - Replacement of left code with right code can decrease another metrics.

```

/* 0(LSB) ~ 7th bit of inputs variable means status of SeatBeltFasten. */
if (((uint16_t)inputs&0x00FFU) == 1) {
    input_SeatBeltFasten = 1;
} else {
    input_SeatBeltFasten = 0;
}

/* 8 ~ 15th bit(MSB) of inputs variable means status of Key. */
switch (((uint16_t)inputs&0xFF00U)>>8) {
case 0:
    /* Key is at Off position */
    input_KEY = 0;
    break;
case 1:
    /* Key is at On position */
    input_KEY = 1;
    break;
case 2:
    /* Key is at Crank position */
    input_KEY = 2;
    break;
default:
    /* erroneous status */
    input_KEY = 3;
    break;
}

```



```

/*
 * Below code separates the incorporated inputs for Key and SeatBeltFasten.
 * - 0(LSB) ~ 7th bit of inputs variable means status of SeatBeltFasten.
 * - 8 ~ 15th bit(MSB) of inputs variable means status of Key.
 */
input_KEY = (uint8_t)(((uint16_t)inputs&0xFF00U)>>8);
input_SeatBeltFasten = (uint8_t)((uint16_t)inputs&0x00FFU);

if ((input_KEY > 2)&&(input_SeatBeltFasten > 1)){
    /* all inputs are strange */
    errnumber = -4;
} else if (input_KEY > 2) {
    /* input_KEY input is strange */
    errnumber = -5;
} else if (input_SeatBeltFasten > 1) {
    /* input_SeatBeltFasten is strange */
    errnumber = -6;
} else {
    errnumber = 0;
}

```

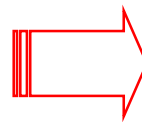
Exercise 3 - 代码度量

- Let's see example code to improve code metrics - 1
 - Replacement of left code with right code can decrease some metrics.

```

if (errnumber == 0) {
    if (KEY_OFF == input_KEY) {
        output_SeatBeltIcon = SB_ICON_OFF;
    } else if (KEY_ON == input_KEY) {
        if (SB_UNFASTENED == input_SeatBeltFasten) {
            if (gFiltered_speed < 15) {
                output_SeatBeltIcon = SB_ICON_ON;
            } else {
                if (timer_cnt > COUNT_FOR_BLINK)
                {
                    if (output_SeatBeltIcon == SB_ICON_OFF) {
                        output_SeatBeltIcon = SB_ICON_ON;
                    } else {
                        output_SeatBeltIcon = SB_ICON_OFF;
                    }
                }
                timer_cnt++;
            }
        } else {
            output_SeatBeltIcon = SB_ICON_OFF;
        }
    } else if (KEY_CRANK == input_KEY) {
        output_SeatBeltIcon = SB_ICON_ON;
    } else {
        errnumber = -4;
    }
}

```



```

if (errnumber == 0) {
    if ((KEY_ON == input_KEY) && (SB_UNFASTENED == input_SeatBeltFasten)) {
        if (gFiltered_speed >= 15) {
            if (timer_cnt > COUNT_FOR_BLINK)
            {
                output_SeatBeltIcon ^= 0x1U;
            }
            timer_cnt++;
        } else {
            output_SeatBeltIcon = SB_ICON_ON;
        }
    } else if (KEY_CRANK == input_KEY) {
        output_SeatBeltIcon = SB_ICON_ON;
    } else {
        output_SeatBeltIcon = SB_ICON_OFF;
    }
}

```

Exercise 3 - 代码度量

■ Consider how to decrease Language Scope

Language Scope

Language scope

[expand all in page](#)

Description

This metric measures the cost of maintaining or changing a function. It is calculated as:

$$(N1 + N2) / (n1 + n2)$$

Here:

- N1 is the number of occurrences of operators.
- N2 is the number of occurrences of operands.
- n1 is the number of distinct operators.
- n2 is the number of distinct operands.

The recommended upper limit for this metric is 10. For lower maintenance cost for a function, try to enforce an upper limit on this metric. For instance, if the same operand occurs many times, to change the operand name, you have to make many substitutions.

- Recommended upper limit : 4
- If you want to decrease calculated value,
 - ✓ Increase the number of distinct operators/operands
 - ✓ Decrease the number of occurrences of operator/operands
 - ❖ If a function has very high value, it's better to separate the function to multiple functions.

- Previous activities already affected this metrics to be decreased.
- BUT! *It doesn't comply with limit of HIS metrics. What will you do?*

Exercise 3 - 代码度量

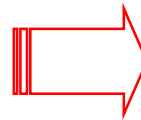
- Let's see example code to improve Language Scope
 - Separate a part of source code as a function.

```
errnumber = separate_inputs_to_Key_and_SeatBeltFasten(inputs, &input_SeatBeltFasten, &input_KEY);
```



```
input_KEY = (uint8_t)((((uint16_t)inputs&0xFF00U)>>8);
input_SeatBeltFasten = (uint8_t)((uint16_t)inputs&0x00FFU);

if ((input_KEY > 2)&&(input_SeatBeltFasten > 1)){
    /* all inputs are strange */
    errnumber = -4;
} else if (input_KEY > 2) {
    /* input_KEY input is strange */
    errnumber = -5;
} else if (input_SeatBeltFasten > 1) {
    /* input_SeatBeltFasten is strange */
    errnumber = -6;
} else {
    errnumber = 0;
}
```



```
static sint32_t separate_inputs_to_Key_and_SeatBeltFasten (sint16_t inputs,
{
    sint32_t errnumber = 0;
    uint8_t SeatBeltFasten, KEY;

    KEY = (uint8_t)((((uint16_t)inputs&0xFF00U)>>8);
    SeatBeltFasten = (uint8_t)((uint16_t)inputs&0x00FFU);

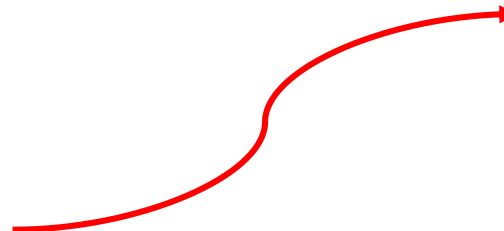
    if ((KEY > 2)&&(SeatBeltFasten > 1)){
        /* all inputs are strange */
        errnumber = -4;
    } else if (KEY > 2) {
        /* Key input is strange */
        errnumber = -5;
    } else if (SeatBeltFasten > 1) {
        /* SeatBeltFasten is strange */
        errnumber = -6;
    } else {
        *input_SeatBeltFasten = SeatBeltFasten;
        *input_KEY = KEY;
        errnumber = 0;
    }

    return errnumber;
}
```

Exercise 3 – 结果对比

Defects, HIS, ... ▾ * New ▾ ⏪ ⏩ ↺			
Family	ID	Information	
[-] MISRA C:2012 4			
[-] 2 Unused code 3			
[-] 8 Declarations and definitions 1			
[-] Code Metrics 4			
[-] File Metrics 1			
[-] Comment Density 1			
[-] ! ★ * 8		Value: 8	
[-] Function Metrics 3			
[-] Cyclomatic Complexity 1			
[-] ! ★ * 12		Value: 13	
[-] Language Scope 1			
[-] ! ★ * 22		Value: 4.9	
[-] Number of Call Levels 1			
[-] ! ★ * 20		Value: 7	

Defects, HIS, ... ▾ * New ▾ ⏪ ⏩ ↺			
Family	ID	Information	
[-] MISRA C:2012 4			
[-] 2 Unused code 3			
[-] 8 Declarations and definitions 1			

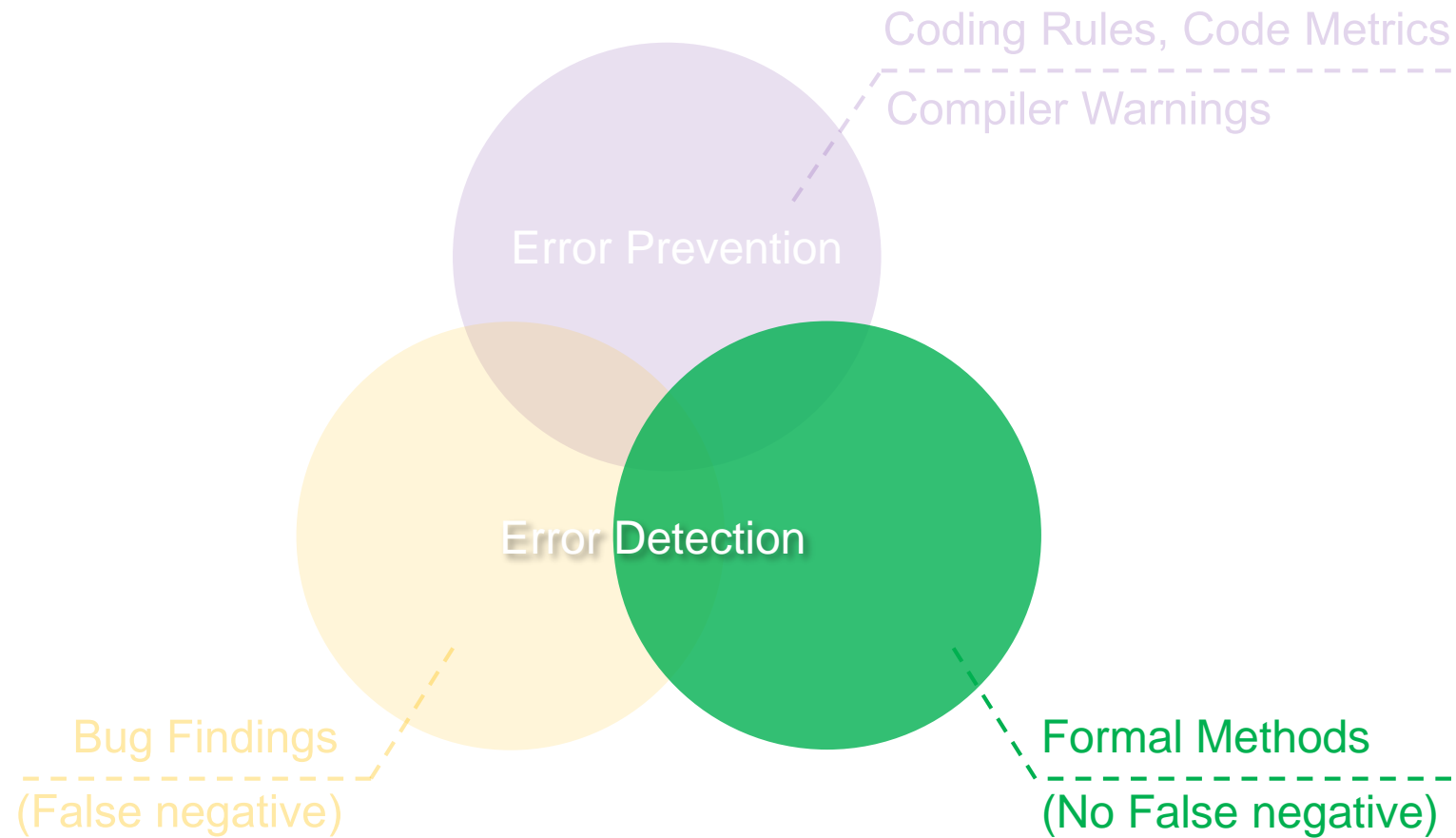


Exercise 4

Review Runtime errors and additional issues

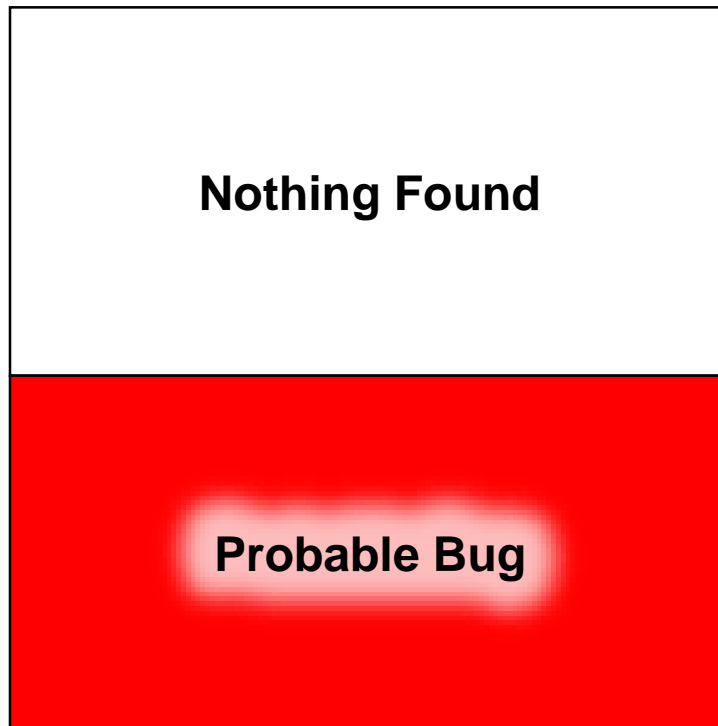
- Import previously used project file for this exercise
 - Review red or orange checks
 - Do anything to make ALL **GREEN**

Exercise 4 – 内容



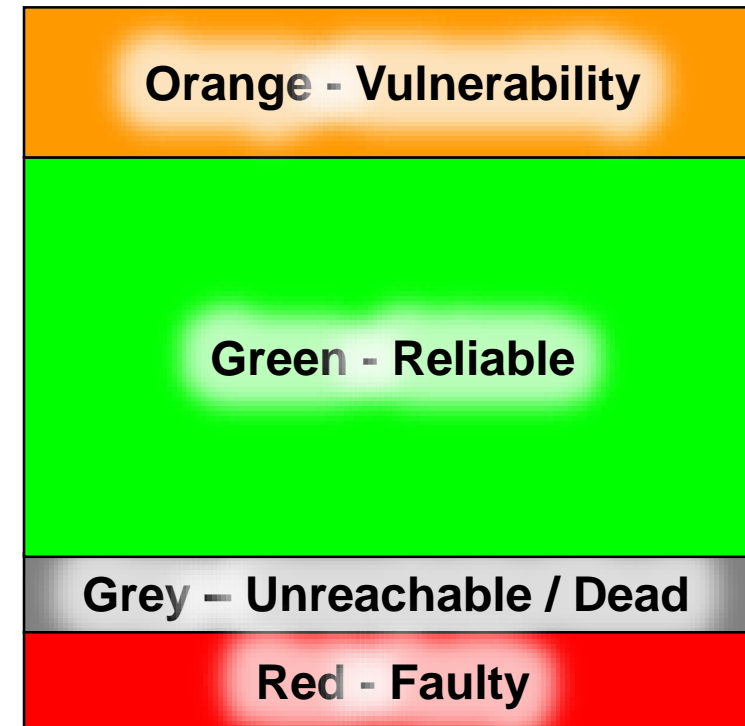
功能对比

Bug Finder



VS.

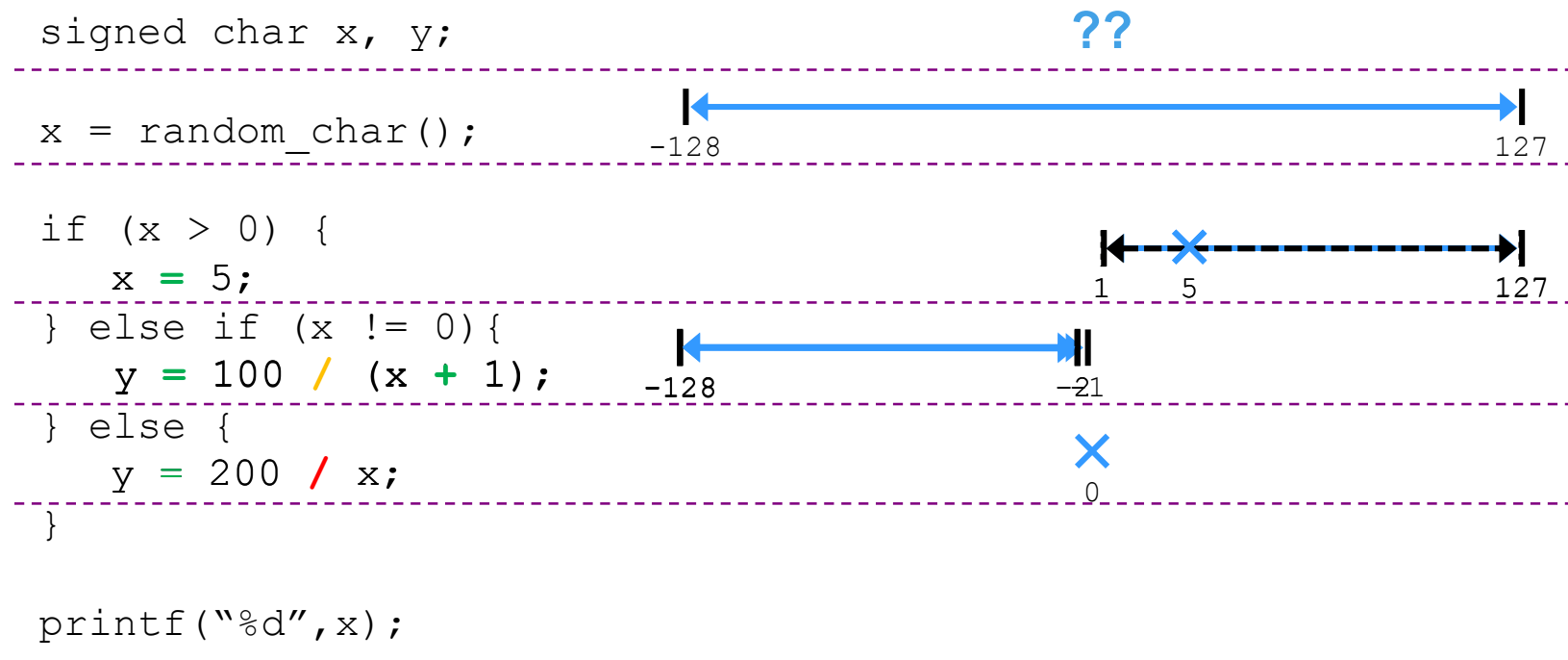
Code Prover



Purple - coding rule violations

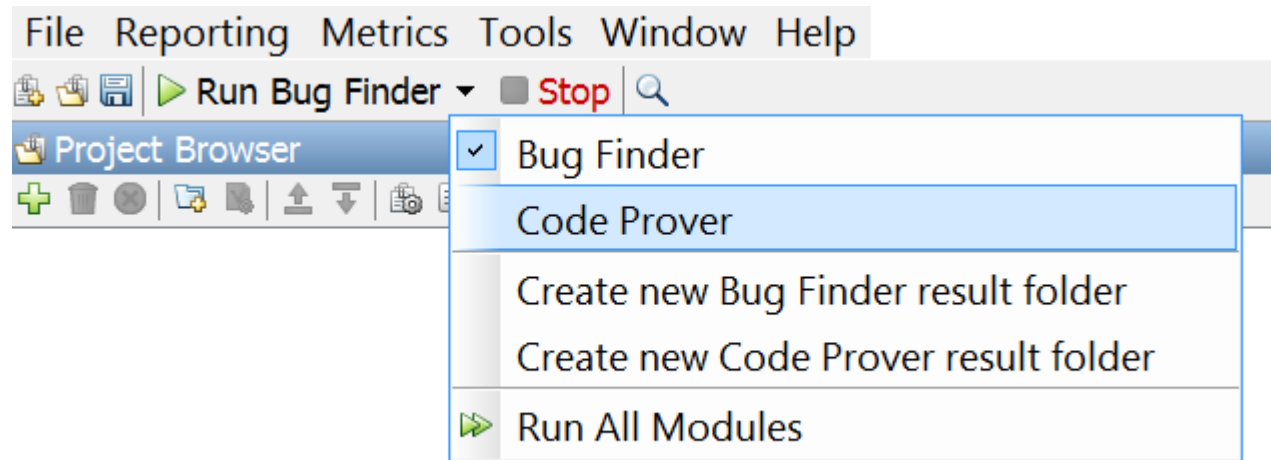
理解抽象解释

- To prove the absence of errors, the Polyspace verification accounts for all possible execution paths using abstract interpretation.



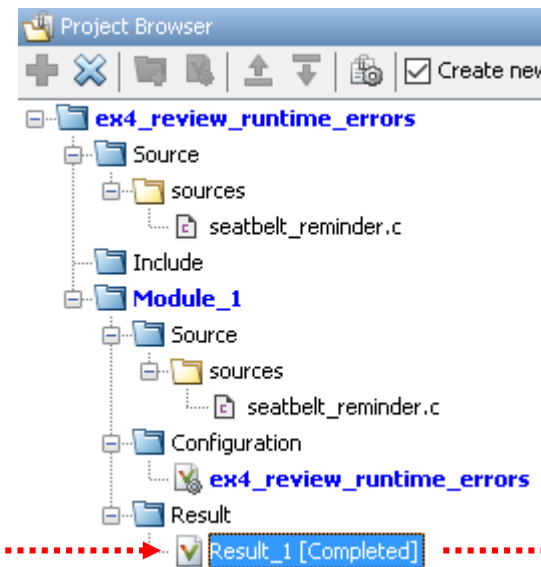
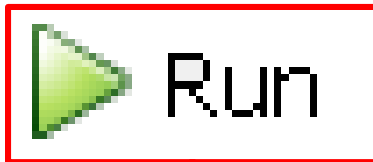
Exercise 4 – 运行时错误检查

- How to run from Bug Finder to Code Prover in 2017a



Exercise 4 – 运行时错误检查

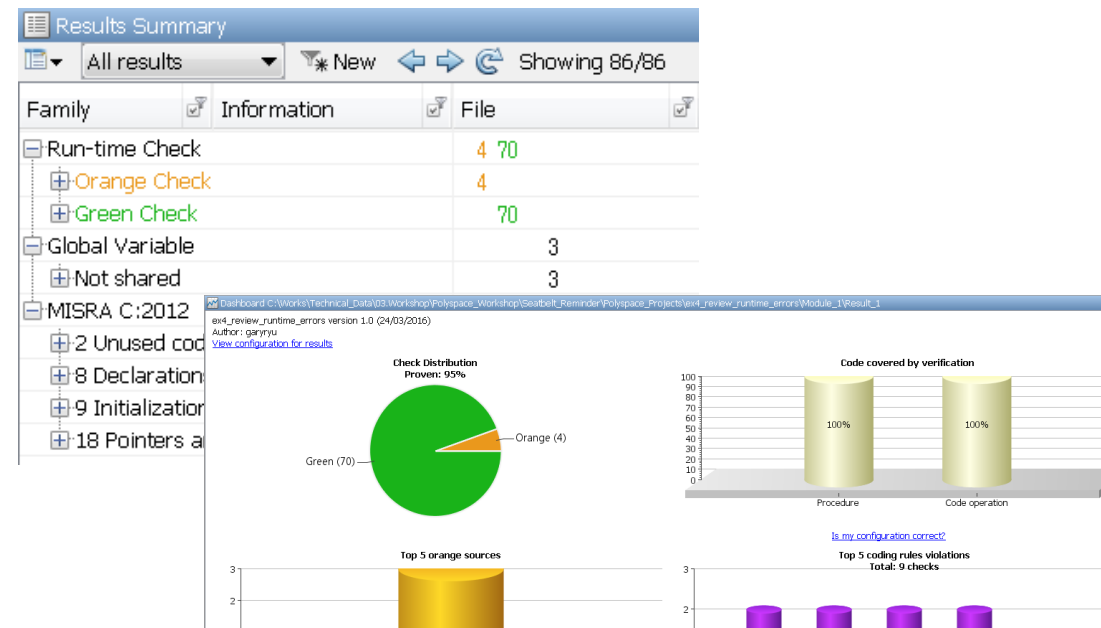
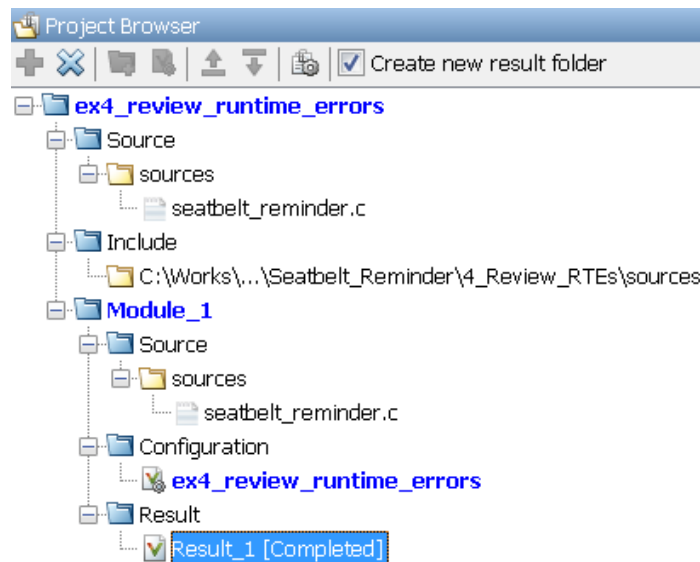
- Run the project in Code Prover



Checks & Rules			New	Showing 80/141
Family	Information	File		
Run-time Check		3 71		
+ Orange Check		3		
+ Green Check		71		
MISRA C:2012		6		
+ 2 Unused code		3		
+ 8 Declarations and definitions		1		
+ 18 Pointers and arrays		2		

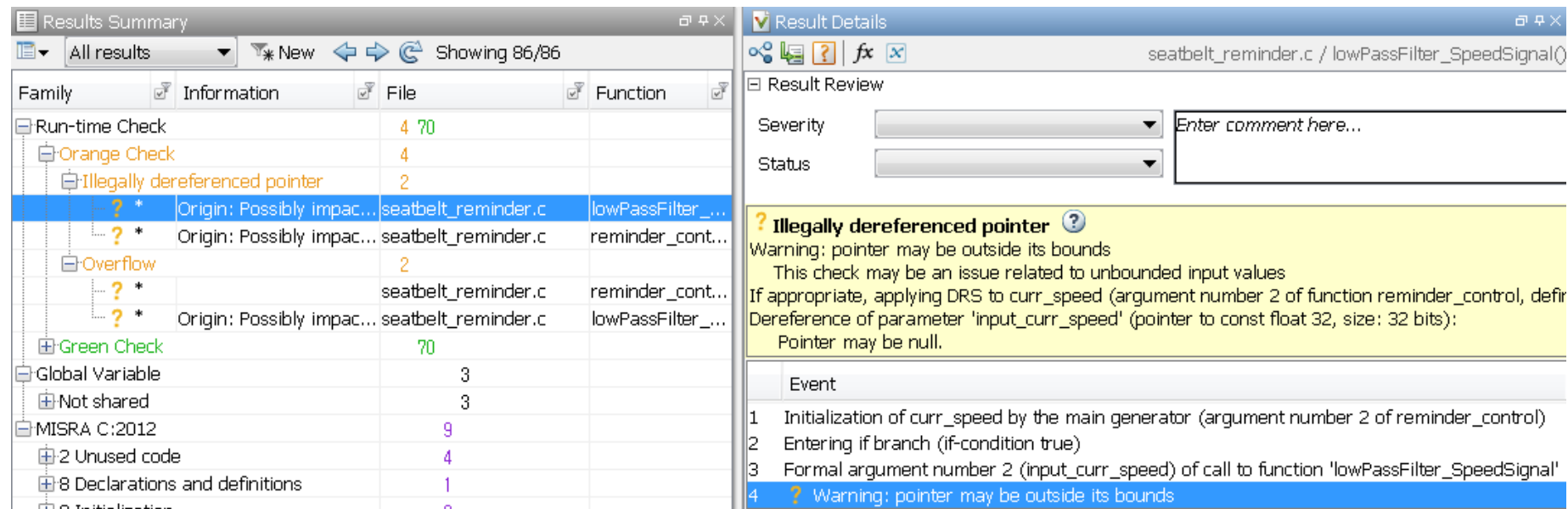
Exercise 4 – 运行时错误检查

- Review all orange checks and additional MISRA violations
 - Use the imported project
 - Or Create new project with source files in *4_Review_RTEs*
 - Code Prover considers all of possible inputs and paths by default



Exercise 4 – 运行时错误检查

- Review all orange checks and additional MISRA violations
 - Analyze orange checks if it's really problem or not
 - For instance, let's see Illegally dereferenced pointer (IDP)



The screenshot displays the MATLAB Static Code Analysis interface. The 'Results Summary' window on the left shows a table of findings, with the 'Illegally dereferenced pointer' warning highlighted. The 'Result Details' window on the right provides a detailed view of this specific warning.

Family	Information	File	Function
Run-time Check	4 70		
Orange Check	4		
Illegally dereferenced pointer	2		
Overflow	2		
Green Check	70		
Global Variable	3		
Not shared	3		
MISRA C:2012	9		
2 Unused code	4		
8 Declarations and definitions	1		
0 Initialization	2		

Result Details: Illegally dereferenced pointer

Warning: pointer may be outside its bounds
 This check may be an issue related to unbounded input values
 If appropriate, applying DRS to curr_speed (argument number 2 of function reminder_control, definition of curr_speed in seatbelt_reminder.c)
 Dereference of parameter 'input_curr_speed' (pointer to const float 32, size: 32 bits):
 Pointer may be null.

Event

- 1 Initialization of curr_speed by the main generator (argument number 2 of reminder_control)
- 2 Entering if branch (if-condition true)
- 3 Formal argument number 2 (input_curr_speed) of call to function 'lowPassFilter_SpeedSignal'
- 4 ? Warning: pointer may be outside its bounds

Exercise 4 – 运行时错误检查

- Review all orange checks and additional MISRA violations
 - Analyze orange checks if it's really problem or not
 - For instance, let's see Illegally dereferenced pointer (IDP)

```

41 static sint32_t lowPassFilter_SpeedSignal (sint32_t * output_Filtered_speed, const float32_t * input_curr_speed)
42 {
43     sint32_t errnumber;
44     float32_t tmp_filtered_speed;
45
46     /* This is relevant to requirement 5.5.
47      * weight for previous input is 0.9 and weight for current input is 0.1.
48      */
49     tmp_filtered_speed = (0.1F * (*input_curr_speed)) + (0.9F * (gPrev_speed));

```

? Illegally dereferenced pointer ?

Warning: pointer may be outside its bounds

This check may be an issue related to unbounded input values

If appropriate, applying DRS to curr_speed (argument number 2 of function reminder_control, defir

Dereference of parameter 'input_curr_speed' (pointer to const float 32, size: 32 bits):

Pointer may be null.

Exercise 4 – 运行时错误检查

- Add a code to check NULL pointer
 - Modify source code to add NULL_PTR
 - Check pointer before use if it's not NULLs

```
tmp_filtered_speed = (0.1F * (*input_curr_speed)) + (0.9F * (gPrev_speed));
```



```
if (((void*)0) != input_curr_speed) {  
    tmp_filtered_speed = (0.1F * (*input_curr_speed)) + (0.9F * (gPrev_speed));  
}
```

- You may use Constraint setup to specify data range for function inputs
 - It can specify data range for global variable and return values of stubbed function

```
tmp_filtered_speed = (0.1F * (*input_curr_speed)) + (0.9F * (gPrev_speed));
```

```
if (tmp_filtered_speed >= 0.5F)
```

```
    if (tmp_filtered_speed
```

```
        errnumber = -1;
```

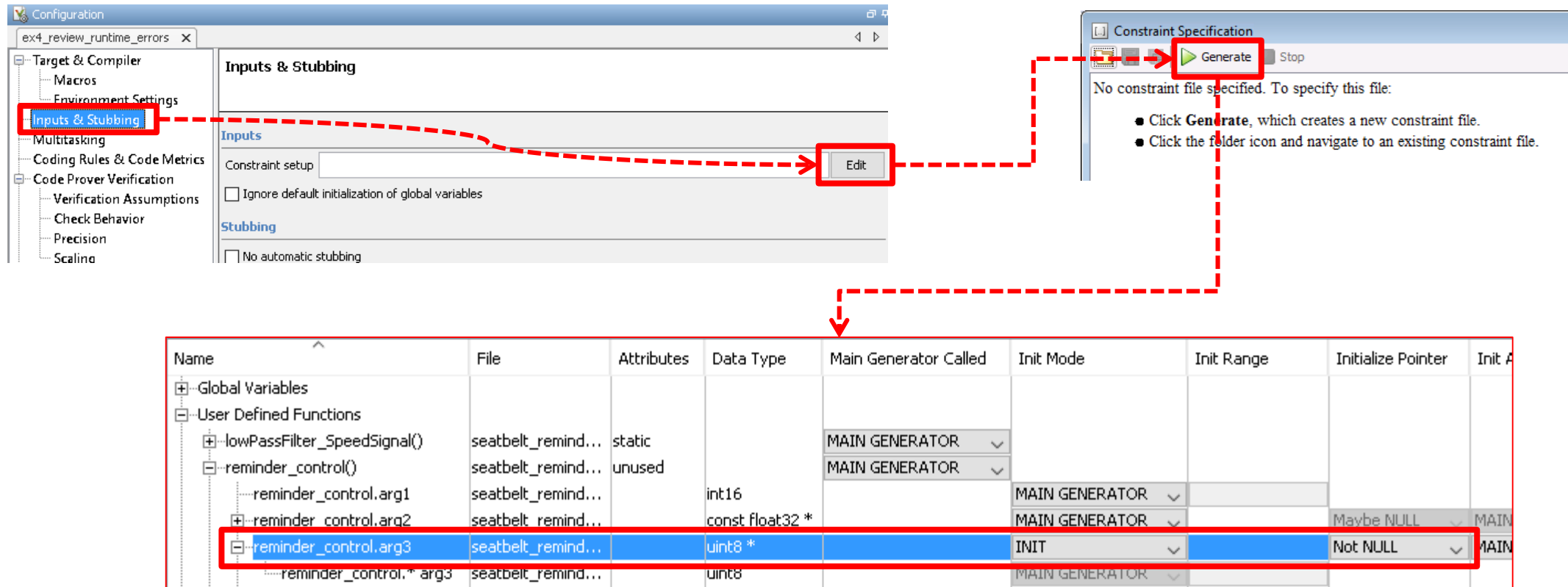
Dereference of parameter 'input_curr_speed' (pointer to const float 32, size: 32 bits):
Pointer is not null.
 Dereferenced value (float 32): full-range [-3.4029E⁺³⁸ .. 3.4029E⁺³⁸]

- What will you do for the variable for `*output` ?

Tip #5 - 数据约束设置



- Specify constraints for global variables, function inputs and return values of stubbed functions



The screenshot shows the Configuration window for 'ex4_review_runtime_errors'. The 'Inputs & Stubbing' tab is selected. The 'Edit' button is highlighted with a red box. A red dashed arrow points from the 'Edit' button to the 'Generate' button in the Constraint Specification dialog. Another red dashed arrow points from the 'Generate' button to the 'Generate' button in the Constraint Specification dialog. The 'Generate' button is highlighted with a red box.

The Constraint Specification dialog shows the following instructions:

- Click **Generate**, which creates a new constraint file.
- Click the folder icon and navigate to an existing constraint file.

The table below shows the configuration for the 'reminder_control' function and its arguments.

Name	File	Attributes	Data Type	Main Generator Called	Init Mode	Init Range	Initialize Pointer	Init A
Global Variables								
User Defined Functions								
lowPassFilter_SpeedSignal()	seatbelt_remind...	static		MAIN GENERATOR				
reminder_control()	seatbelt_remind...	unused		MAIN GENERATOR				
reminder_control.arg1	seatbelt_remind...		int16		MAIN GENERATOR			
reminder_control.arg2	seatbelt_remind...		const float32 *		MAIN GENERATOR		Maybe NULL	MAIN
reminder_control.arg3	seatbelt_remind...		uint8 *		INIT		Not NULL	MAIN
reminder_control.* arg3	seatbelt_remind...		uint8		MAIN GENERATOR			

Exercise 4 –运行时错误检查

- Review another orange check (Overflow)
 - What is the problem of orange overflow check?

```
if (gFiltered_speed >= 15) {
    if (timer_cnt > COUNT_FOR_BLINK)
    {
        output_SeatBeltIcon ^= 0x1U;
    }
    timer_cnt++;
}
```

? Overflow ?
 Unproven: operation [+] on scalar may overflow (result strictly greater than MAX INT32)
 operator + on type int 32
 left: [0 .. 2³¹-1]
 right: 1
 result: [1 .. 2³¹-1]
 (result is truncated)

- Signed integer overflow leads to undefined behavior. **How can I fix it?**
- Oops**, I forgot to set it to zero when the value exceeds COUNT_FOR_BLINK.

Add this code ----->

```
if (gFiltered_speed >= 15) {
    if (timer_cnt > COUNT_FOR_BLINK)
    {
        output_SeatBeltIcon ^= 0x1U;
        timer_cnt = 0;
    }
    timer_cnt++;
}
```

✓ Overflow ?
 Operation [+] on scalar does not overflow in INT32 range
 operator + on type int 32
 left: [0 .. 2000]
 right: 1
 result: [1 .. 2001]
 (result is truncated)

Exercise 4 – 结果对比

[-] Run-time Check	3 71
[+] Orange Check	3
[+] Green Check	71
[-] MISRA C:2012	6
[+] 2 Unused code	3
[+] 8 Declarations and definitions	1
[+] 18 Pointers and arrays	2

[-] Run-time Check	76
[+] Green Check	76
[-] MISRA C:2012	4
[+] 2 Unused code	3
[+] 8 Declarations and definitions	1

Two MISRA violations are disappeared by removal of orange checks.

内容

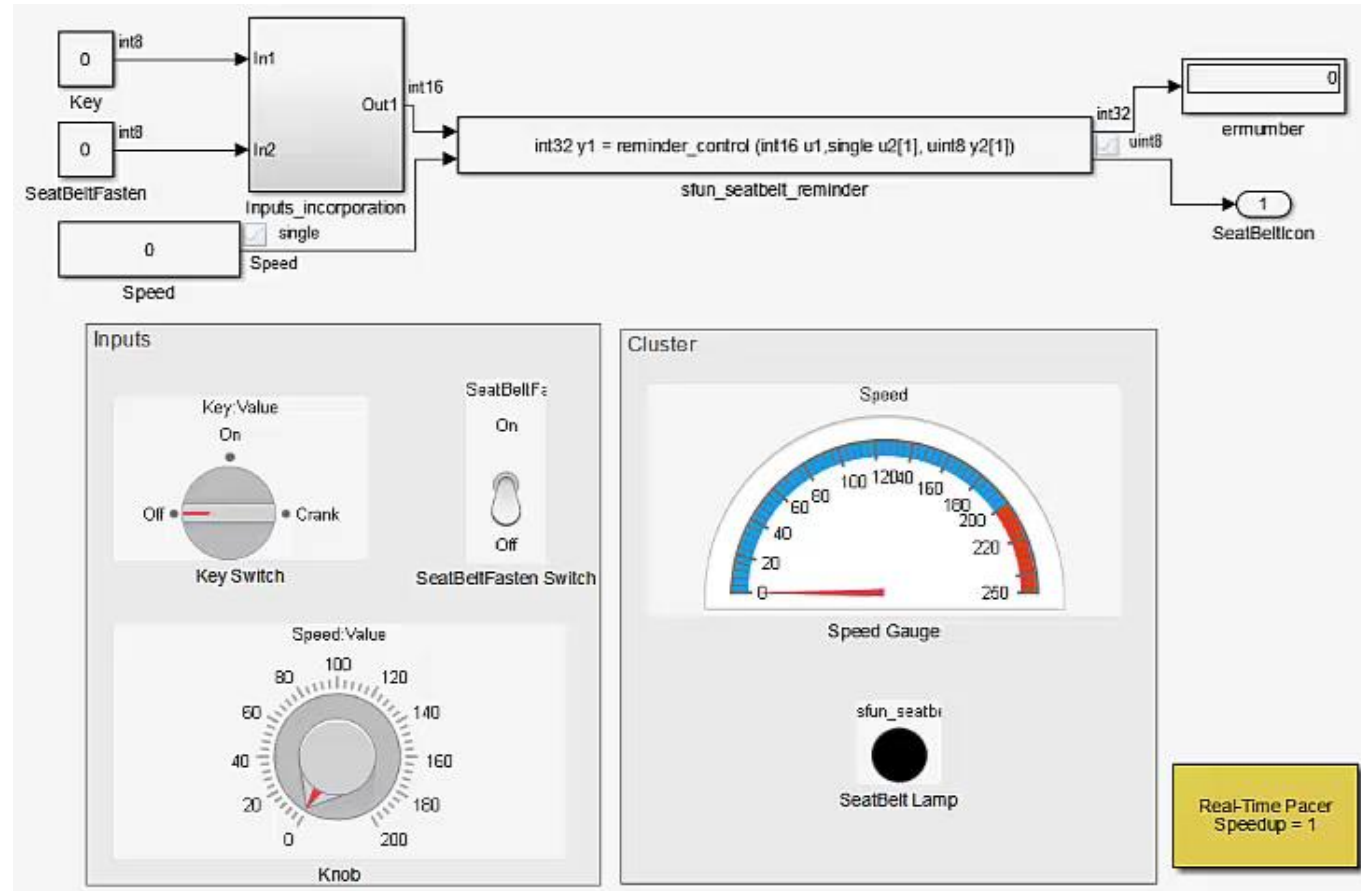
- Polyspace Overview
 - Category of Static analysis
 - Use case of Polyspace products
- **Polyspace Hands-on workshop**
 - Description of the logic
 - Exercise 1 / 2 / 3 / 4 (Review MISRA, Defect, Code Metrics, Runtime Error)
 - **Conclusion**
- Q&A

Conclusion

Let's simulate the final version of source code

Conclusion – 最终结果仿真

- Simulate the final version of source code



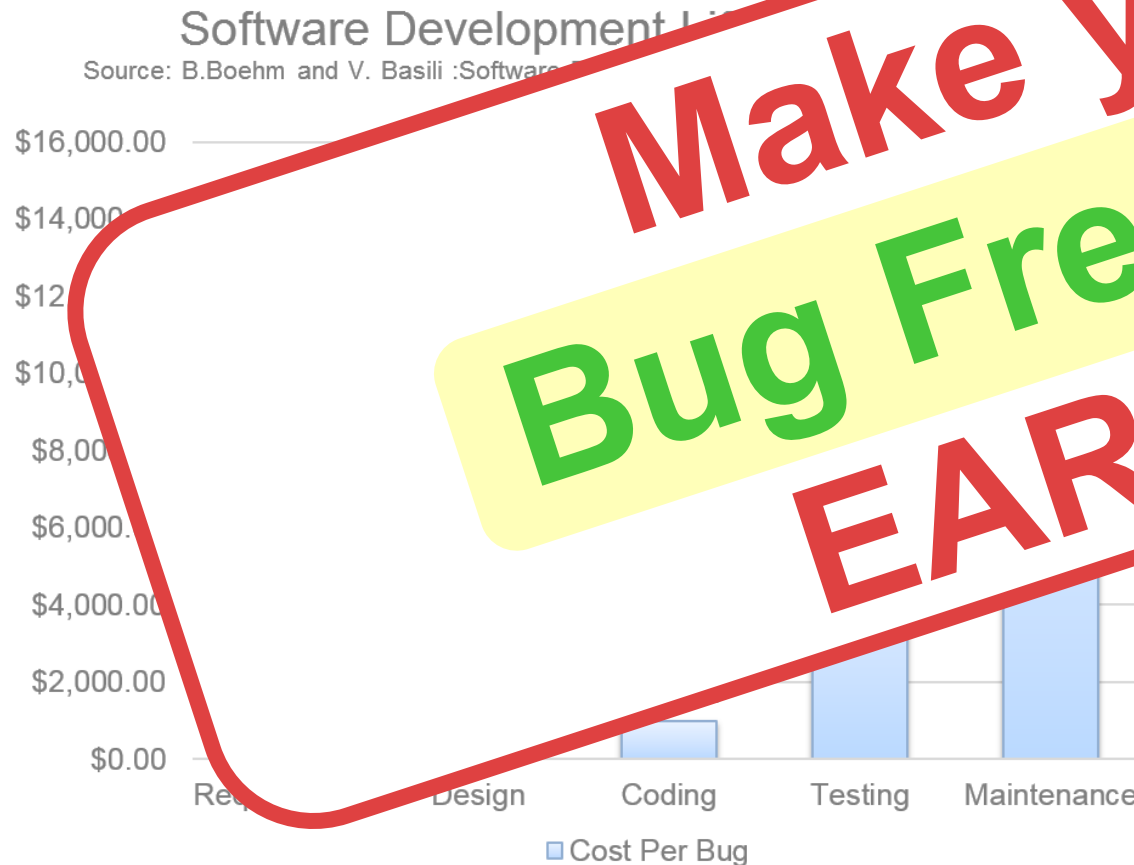
问题回顾

Summary of Issues...

Issues we faced!	What's improved!
Lamp doesn't work if speed is slower than 0.5. or Errnumber shows strange value.	Remove Non-initialized variable and dead code <ul style="list-style-type: none"> Initialize <i>errnumber</i> in <i>lowPassFilter_SpeedSignal</i> This is detected by Bug Finder's Defects and MISRA C:2012 checker. Dead code is directly related to this bug but it's meaningless.
Lamp flickers too fast if speed is faster than 15.	Remove overflow occurs by timer_cnt <ul style="list-style-type: none"> Set <i>timer_cnt</i> to zero(0) in <i>reminder_control</i> function. This runtime error is detected as orange overflow check by Code Prover.
MISRA C:2012 Guide line	Modify / Justify source code <ul style="list-style-type: none"> It's helpful to avoid runtime errors.
Code Metrics	Improve structure of source code by modification <ul style="list-style-type: none"> It's good to improve readability and maintenance cost.

早期验证

Use Verification tools early in the life cycle



**Make yours
Bug Free ZONE
EARLIER!**

- progress of a project
- External quality assessor
 - Audit supplier's code
- Model-Based Design
 - On generated code

Q & A