ECLAIR intercepts the command line options: Why is this valuable?

Frank Büchner, 003 – 2020-06-08

Contents

1 Introduction

This example shows that ECLAIR intercepts the command line of the call to the compiler and considers it.

This example is made as simple as possible.

2 Prerequisites

This example assumes that

- ECLAIR (i.e. eclair_env.exe and eclair_report.exe)
- gcc

are in the path.

3 Application Files

3.1 Source File: main.c

There is only one source file.

```
struct bf {
    int a : 1;
    int b : 2;
};

int main(void)
{
    struct bf my_bf;

    my_bf.a = 1;
    my_bf.b = 2;

    return(my_bf.a+my_bf.b); // just to use the bitfields
}
```

Fig. 1 The contents of main.c

3.2 Batch File: compile.bat

The batch file compile.bat compile main.c to a.exe using gcc.

```
gcc -Wall -std=c99 main.c
```

Fig. 2 The contents of compile.bat

The command line option “-Wall” enables all warnings.

The command line option “-std=c99” sets the C language version to C99.

```
c:\Projects\ECLAIR\Hitex\signed-bitfield\work>compile.bat

c:\Projects\ECLAIR\Hitex\signed-bitfield\work>gcc -Wall -std=c99 main.c

c:\Projects\ECLAIR\Hitex\signed-bitfield\work>
```

Fig. 3 Calling “compile.bat” creates a.exe without producing any messages (ECLAIR is not yet involved)

4 Analyzing with ECLAIR

4.1 Batch File: analyze.bat

Usually a batch file “analyze.bat” conducts analysis with ECLAIR.

4.1.1 First Part

In the first part of this batch file defines some environment variables.

```
set CC_ALIASES="gcc"
```

Fig. 4 This environment variable directs ECLAIR to intercept gcc (and the command line options of gcc)

4.1.2 Middle Part

In the middle part of this batch file conducts the analysis with ECLAIR.

```
eclair_env -eval_file=analyze.ecl -- compile.bat
```

Fig. 5 The command "eclair_env" conducts the actual analysis

The file "analyze.ecl" controls the analysis. In this file, ECLAIR is set up to check for violations of the MISRA C:2012 guidelines.

The part of the command line after "--" is the command that is used to build the project that shall be analyzed by ECLAIR. In our case, compile.bat contains the call to gcc. ECLAIR intercepts the call to gcc and the command line options of the call to gcc. For instance, ECLAIR recognizes that the software shall be compiled according to the C99 standard (-std=c99).

```
c:\Projects\ECLAIR\Hitex\signed-bitfield\work>eclair_env -eval_file=analyze.ecl -- compile.bat

c:\Projects\ECLAIR\Hitex\signed-bitfield\work>gcc -Wall -std=c99 main.c
```

Fig. 6 The effects of the command "eclair_env"

4.1.3 Last Part

The last part of this batch file generates the ECLAIR reports.

5 Results of Analysis with ECLAIR

ECLAIR is setup to check for violations of rules in MISRA C:2012. The results shows the violation of several rules.

service	violation	caution	information
B.EXPLAIN			1
MC3.D1.1	6	3	
MC3.D4.1	27		
MC3.R6.1	2		
MC3.R6.2	1		
MC3.R10.6	1		

Fig. 7 The list of the MISRA rules that are violated

We are especially interested in the violations of rule 6.2 (MC3.R6.2). There is only one such violation.

service MC3.R6.2: Single-bit named bit fields shall not be of a signed type (1 violation)
☰ violation for MC3.R6.2
main.c:11.9-11.17: non-compliant bit-field has type `int' and width 1

Fig. 8 Details of the violation of rule 6.2

```
10  struct bf {
11          int a : 1;
☰ MC3.R6.2 non-compliant bit-field has type `int' and width 1
12          int b : 2;
13  };
```

Fig. 9 The source code where rule 6.2 is violated

The bit-field designator *a* is of type *int* and *int* is implicitly signed. Signed variables need at least one bit to store the sign. However, *a* is only one bit wide. Hence, there is no bit left to store the value (or there is only space for the value, but not for the sign). In consequence, this declaration is most probably wrong, because it makes no sense.

6 Results of Analysis with ECLAIR and a Different Batch File compile.bat

6.1 A Different Batch File compile.bat

We add the compiler option `-funsigned-bitfields`.

```
gcc -Wall -std=c99 -funsigned-bitfields main.c
```

Fig. 10 The new contents of compile.bat (compare with fig. 2)

The additional compiler option `-funsigned-bitfields` tells the compiler to treat all bit-fields as unsigned.

6.2 Re-Analysis with ECLAIR

As before, executing analyze.bat conducts the analysis with ECLAIR. Again, the command “eclair_env” in analyze.bat conducts the actual analysis.

```
c:\Projects\ECLAIR\Hitex\signed-bitfield\work>eclair_env -eval_file=analyze.ecl -- compile.bat

c:\Projects\ECLAIR\Hitex\signed-bitfield\work>gcc -Wall -std=c99 -funsigned-bitfields main.c
```

Fig. 11 The effects of the command “eclair_env” with different contents of compile.bat (compare with fig. 6)

Due to the changed contents of compile.bat, the option `-funsigned-bitfields` is used for gcc. Because ECLAIR intercepts the command line, ECLAIR knows that the bit-field designator *a* is no longer signed, but unsigned. This means, no sign bit is needed any more, and the remaining bit can be used for the value. In other words, the problem is gone. And so is the violation of rule 6.2.

service	violation	caution	information
B.EXPLAIN			1
MC3.D1.1	6	3	
MC3.D4.1	27		
MC3.R6.1	2		
MC3.R10.3	1		

Fig. 12 Due to the changed compile.bat, the violation of rule 6.2 is no longer reported

7 Conclusion

Neither the source code, nor the analysis setup of ECLAIR (i.e. the analyze.bat and analyze.ecl) nor the report generation of ECLAIR has changed. The only thing that has changed was how the project was build, i.e. the contents of compile.bat, where an option was added. In consequence, a violation of a MISRA rule has disappeared.

ECLAIR intercepts compiler options and considers them. Considering just the source code is not enough to decide if a violation is present or not. Considering just the source code can lead to false positives (or false negatives).

8 Version Used

This was made using ECLAIR 3.8.0.

9 Author

Frank Büchner
Dipl.-Inform.
Hitex GmbH
Greschbachstr. 12
D-76229 Karlsruhe
Tel.: +49-721-9628-125
frank.buechner@hitex.de

Any comments are appreciated.