Developing high-quality software is tough. ECLAIR is designed to help development, QA, and safety teams reach their quality goals.

MC2: MISRA C:2004 + Essentials

The *ECLAIR MC2* package is a combination of several of the many applications that run on top of ECLAIR, a powerful platform for the automatic analysis, verification, testing and transformation of C and C++ programs. This particular package combines:

- a state-of-the-art, medium-weight static analyzer that almost completely automates the assessment of compliance with **MISRA C:2004**, **BARR-C:2018**, **AUTOSAR-C:2009**, as well as other, complementary coding rules;
- a precise and flexible implementation of the source code metrics defined by HIS;
- the **ECLAIR Bug Finder**, a very fast static analyzer able to detect bugs and weaknesses that can lead to crashes, misbehaviors, and security vulnerabilities;
- the **ECLAIR Project Organization Checker**, for the automatic checking of the system architecture, independence and *freedom from interference* of software components.

1 Highlights

- The highest coverage of MISRA-C:2004 available on the market.
- No time wasted in writing compiler personality files (often of questionable correctness).
- Automatic production of accurate, faithful and detailed compliance reports.
- Easy-to-use yet powerful graphical user interface.
- *Real-time* use from within most popular IDEs or *batch* use with analysis results stored in a database.
- Guideline violation and metric reports optionally available to the entire development team and management using ordinary web-based technology.
- Powerful mechanisms of differential reporting allow correlating changes in the code and the appearance/disappearance of violations (with possible interfaces to issue-tracking systems).
- No stress: free consultancy services for the initial configuration. This includes full assistance to help your company make the transition to the *MC2* package.

2 MISRA-C:2004

MISRA-C:2004 is a mature, well-understood software development C subset developed by MISRA for the motor industry, which is now a de facto standard for safety-, life-, and mission-critical embedded applications in many industries including aerospace, railway, medical, telecommunications and others.[1]

2.1 Coverage and Precision

The *ECLAIR MC2* package offers the most extensive MISRA-C:2004 coverage available on the market, by providing support for all the guidelines.

Guidelines are enforced using very general and *accurate* checkers, which operate on the precise sequences of tokens and abstract syntax trees that are manipulated by the compiler. Coupled with the fact that ECLAIR always checks each guideline in the appropriate context (at the token, declaration, translation unit, whole program or whole system levels), this makes sure that the checkers for decidable rules are *exact* (neither false positives nor false negatives). For undecidable rules, ECLAIR's *MC2* package provides a medium-weight solution to the tradeoff among computational complexity, number of false positives and number of false negatives.[2] In any case, when false negatives are possible, they are always clearly and unambiguously delimited.

Coverage of the MISRA-C:2004 guidelines is summarized in the following table:

Support level	**#**
Fully supported (without false negatives)	127
Partially supported (with possible false negatives)	15
Total	142

3 Compliance Reports

ECLAIR can be configured to automatically produce compliance reports required to meet contractual obligations and industrial standards such as ISO 26262. The compliance report is obtained from the actual configuration, which, if properly done, will contain the reason for each deviation. Thus, carrying its rationale, any deviation goes straight from the configuration to the report.

In addition, thanks to ECLAIR's ability to intercept and fully understand the communication with the toolchain, the compliance report contains full details about the code and its analysis: which files have been compiled and/or analyzed (with full path and a cryptographic hash of their contents), the compiler/linker options, the full version of ECLAIR, ..., with even a cryptographic hash of the generated executables. All this allows standard-compliant, fully-reliable tracing of the compliance reports to the executable software that is actually run on the device.

4 BARR-C:2018, ECLAIR Bug Finder, and Other Essentials

The *Barr Group's Embedded C Coding Standard*,[3] BARR-C:2018, is, for coding standards used by the embedded system industry, second only in popularity to MISRA C. The adoption of the stylistic subset of BARR-C:2018 (79 out of 143 rules) can be part of complying with the MISRA requirement that a consistent programming style is adopted and systematically used as part of the software development

[1] Motor Industry Software Reliability Association. *MISRA-C:2004 — Guidelines for the use of the C language in critical systems*. MIRA Limited, Nuneaton, Warwickshire CV10 0TU, UK, October 2004.

[2] R. Bagnara, A. Bagnara, and P. M. Hill. Coding guidelines and undecidability. In DESIGN&ELEKTRONIK, editor, *embedded world Conference 2023 — Proceedings*, pages 488–499, Nuremberg, Germany, 2023. WEKA FACHMEDIEN, Richard-Reitzner-Allee 2, 85540 Haar, Germany.

[3] M. Barr. *BARR-C:2018 — Embedded C Coding Standard*. Barr Group, 2018.

process.[4] ECLAIR support for BARR-C:2018 has no equals on the market. The *ECLAIR Bug Finder* identifies security vulnerabilities, dead code, API misuses and other errors in C and C++ source code, including buffer overflows, dereferences of null pointers, pointer arithmetic errors, use of uninitialized variables, uninitialized or invalid return values, divisions by zero, undefined operations, dead stores, leaks of stack memory addresses, memory leaks, unreachable code, double-free, use-after-free, other dynamic memory allocation issues, lossy implicit conversions, excessive padding (memory waste), vararg functions mistakes, string manipulation errors, library API violations, insecure use of library functions, multithreading issues, dynamic type errors.

ECLAIR MC2 includes a requirements management tool as well as a service for the automatic checking of the traceability between requirements and program entities: this is prescribed by MISRA C and all functional-safety standards.

ECLAIR MC2 also provides a very general service to automatically verify architectural constraints at the software level. It is able to check, by control and data flow static analyses, all interactions between user-defined software elements occurring via read or write accesses to shared memory, function calls, passing and returning of data, as well as static dependencies due to header file inclusion and macro expansion. It can thus be used, e.g., to enforce constraints about layering and to prevent bypassing of software interfaces. Most importantly, it can be used to provide evidence of *independence/isolation/segregation/freedom from interference* as required, in one form or another, by all safety and security standards.

The *ECLAIR MC2* package includes dozens of other useful services, among which are those supporting the AUTOSAR-C:2009 implementation rules for the development and maintenance of all AUTOSAR *Basic Software* modules written in C, customizable naming rules, as well as additional software metrics.

5 HIS and Other Source Code Metrics

Source code metrics are recognized by many software process standards (and from MISRA) as providing an objective foundation to efficient project and quality management. One well known set of metrics has been defined by HIS (Herstellerinitiative Software, an interest group set up by Audi, BMW, Daimler, Porsche and Volkswagen).[5]

The *HIS source code metrics*, while well established, include some metrics that are obsolete and miss others that are required or recommended by software process standards, such as those that allow estimating function coupling. For this reason, HIS source code metrics are supplemented by numerous other metrics that allow software quality to be assessed in terms of complexity, testability, readability, maintainability and so forth. Keeping track of these metrics also provides an effective and objective method to assess the quality of the software development process.

5.1 Coverage

ECLAIR's *MC2* package provides very precise and flexible coverage for the 12 HIS metrics with boundary limits: CALLING, CALLS, COMF, GOTO, LEVEL, PARAM, PATH, v(G), RETURN, STMT, VOCF and ap_cg_cycle. In addition, it contains 41 non-HIS metrics: the 53 metrics provided make ECLAIR's *MC2* package a complete software measurement solution.

All metrics may be incrementally reported, showing exactly where in the code the value was computed

[4] R. Bagnara, M. Barr, and P. M. Hill. *BARR-C:2018 and MISRA C:2012 (with Amendment 2): Synergy Between the Two Most Widely Used C Coding Standards*. embedded world Conference 2021 DIGITAL — Proceedings. WEKA FACHMEDIEN, Germany, 2021.

[5] HIS source code metrics. Report *HIS-SC-Metriken.1.3.1-e*, Herstellerinitiative Software, Software Test Working Group, April 2008. Version 1.3.1.

or aggregated (e.g., maximized, summed, averaged) over a single function, translation unit, program, or the whole project.

If a limiting value for a metric is provided, ECLAIR can report where this value is attained and also, if needed, each subsequent point in the code where a value that breaches the limit is computed.

6 Proper Integration with the Toolchain

ECLAIR MC2, like all packages that run on ECLAIR, intercepts every invocation of the toolchain components (compilers, linker, assembler, archive manager) and it automatically extracts and interprets the options that the build system has passed to them. This allows for the seamless integration with any build system.

Moreover, you do not need to engage in error-prone activities such as (a) specifying which files make up the application, and where the right header files are located; (b) configuring the static analyzer so that the analysis parameters match the options given to the compilers (several options *do* affect the program semantics); (c) writing down predefined macros and the architectural parameters such as sizes, alignment constraints, address spaces and so forth. All this is automatic and supports build processes that involve the automatic generation of source files that depend on the configuration, without the need to develop and maintain a separate analysis procedure: with ECLAIR the existing build procedure can be used verbatim.

ECLAIR is available on most modern flavors of UNIX®, Linux, macOS® and Windows®, including Cygwin and MinGW, and can be used with just about any development environment. ECLAIR supports parallel and distributed program analysis, to leverage available computing resources. Most popular C/C++ compilers and cross compilers are supported, including AMD Xilinx™, Andes, ARM®, CAES™, CodeWarrior™, Cosmic Software, CrossWorks™, Emscripten, Espressif™, Freescale™, GCC and its derivatives, Green Hills®, HighTec, IAR™, Infineon™, Intel®, Keil Software®, Melexis™, Microsoft®, MPLAB®, NXP™, QNX™, Renesas Electronics, SOFTUNE™, TASKING®, Texas Instruments™, xPack, Wind River®, as well as clang/LLVM and its derivatives.

7 Graphical User Interface

All the verification tasks supported by ECLAIR can be specified and refined incrementally by means of a very convenient graphical user interface. This allows, for instance: finding coding rules using a powerful tag-based selection logic; activating and customizing coding rules, possibly restricting their use to only part of the project; selecting and customizing the kind of outputs to be generated; defining project deviations and specific deviations (all deviations will in any case be reported into the final analysis report); choosing to run the verification task immediately or save the task for later.

Detailed analysis reports can be very conveniently browsed within or outside the GUI using any web browser, possibly in connection with popular IDEs like *Eclipse* and its derivatives,[6], *NetBeans* and its derivatives,[7], *IntelliJ IDEA* and its derivatives,[8], or extensible editors like *Visual Studio Code*, *Microsoft Visual Studio*, and *GNU Emacs*. With a suitable license, detailed outputs in HTML format can also be generated for publication on the LAN.

[6] Such as *Arm Development Studio*, *CodeWarrior Development Studio*, *CrossCore Embedded Studio*, *HighTec Development Platform*, *MCUXpresso IDE*, *QNX Momentics Tool Suite*, *Renesas e^2 studio*, *SiFive Freedom Studio*, *Silicon Labs Simplicity Studio*, *STM32CubeIDE*, *TASKING TriCore Eclipse IDE*, *Texas Intruments Code Composer Studio*, and *Xilinx Vitis IDE*.

[7] Such as *MPLAB X IDE*.

[8] Such as *Android Studio* and *CLion*.

8 Qualifiability for Safety-Related Development

ECLAIR's *MC2* package is qualifiable for safety-related development. ECLAIR qualification kits support tool qualification following the prescriptions of all major functional safety standards: CENELEC EN 50128 (railway), ECSS-Q-ST-80C (space), IEC 61508 (industrial), IEC 62304 (medical), ISO 26262 (automotive), RTCA DO-178C/DO-330 (aerospace).

9 The Bigger Picture

ECLAIR is very flexible and highly configurable: it supports all kinds of software development workflows and environments.

ECLAIR is fit for use in mission- and safety-critical software projects: it has been designed from the outset to exclude configuration errors that would undermine the significance of the obtained results.

ECLAIR is developed in a rigorous way and carefully checked with extensive internal test suites (tens of thousands of test cases) and industry-standard validation suites.

ECLAIR is based on solid scientific research results and on the best practices of software development.

ECLAIR's unique features and BUGSENG's strong commitment to the customer, allow for a smooth transition to ECLAIR from any other tool.

BUGSENG's quality system has been certified by TÜV Italia (TÜV SÜD Group) to comply with the requirements of UNI EN ISO 9001:2015 for the "Design, development, maintenance and support of tools for software verification and validation" (IAF 33).

BUGSENG is an Arm's Functional Safety Partner, and is thus recognized as a partner who can reliably support their customers with industry leading functional safety products and services.

Similar packages with MISRA C:2012 and MISRA C++:2008 are also available!

For More Information

BUGSENG srl
Via Marco dell'Arpa 8/B
I-43121 Parma, Italy
Email: info@bugseng.com
Web: http://bugseng.com
Tel.: +39 0521 461640

bugseng

**no shortcuts,
no compromises,
no excuses:
software verification done right**