The Easier Way To Deliver World Class Crestron Programs for Your Customers

Who Is The Intended Audience?

Anyone responsible for delivering high quality Crestron® programming solutions based upon the SIMPL logic engine.

How Do We Intend To Raise The Quality of Crestron Solutions?

In your role, it's essential you deliver the best possible Crestron® projects - whether that is to:

- Minimise service calls.
- Reduce maintenance costs.
- Win competitive advantage.

Until now, without painstaking effort, it has been impossible to gauge whether a Crestron® SIMPL program is robust and adaptable. This can mean:

- General maintenance is hard or impossible.
- Defects are harder to diagnose and eliminate.
- Reliability issues - you get inconsistent outputs from the same inputs – bringing unexpected, costly results.

SIMPLified™ 2 addresses these concerns – using our "static analysis" capability provides a transparent measure of program conformity.

While this does not provide a measure of functional correctness – a program may achieve a high conformity score and yet still have defects – it does build confidence that the programming has been executed with a degree of care and attention and standards have been adhered to.

How Does It Work?

If you are responsible for commissioning the programming or validating a programming deliverable, it is simply a case of instructing your Crestron integrator to provide a SIMPLified program analysis certificate along with the completed program and source code archives. The certificate (see Figure 1) will summarise a qualified measure of the program under analysis' *structure* to build confidence in the quality of the product.

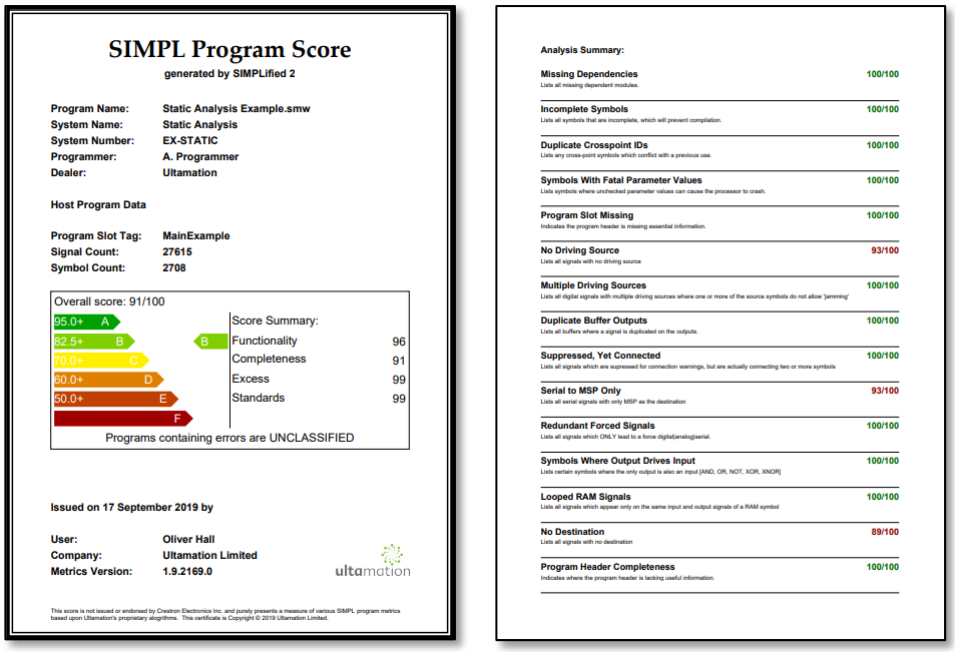
For integrators, SIMPLified™ 2's design entitlement is all that is required to generate the certificate and associated narrative. Just point SIMPLified™ at your program and click on "Generate Certificate" to create a PDF which can be attached to your hand-over pack. If your program is scoring poorly in any area, the narrative and program analysis pane[1] will point you quickly to where the issues are being identified.

That's all there is to it – if you wish for further information around the types of analysis that are performed and a detailed description of the scoring algorithm please read the following pages.

"At A-Knowledge, Simplified has helped us improve our Crestron programming and has given us the means to achieve clean, bug free solutions at a much higher rate."

Niko Brasseur – A-Knowledge

[1] Other entitlements may be required to provide full analysis detail. See http://shop.ultamation.com



SIMPL Program Score

generated by SIMPLified 2

Program Name: Static Analysis Example.smw
System Name: Static Analysis
System Number: EX-STATIC
Programmer: A. Programmer
Dealer: Ultamation

Host Program Data

Program Slot Tag: MainExample
Signal Count: 27615
Symbol Count: 2708

Overall score: 91/100

95.0+ A
82.5+ B
70.0+ C
60.0+ D
50.0+ E
F

Score Summary:

Functionality	96
Completeness	91
Excess	99
Standards	99

Programs containing errors are UNCLASSIFIED

Issued on 17 September 2019 by

User: Oliver Hall
Company: Ultamation Limited
Metrics Version: 1.9.2169.0

ultamation

This score is not issued or endorsed by Crestron Electronics Inc. and purely presents a measure of various SIMPL program metrics based upon Ultamation's proprietary algorithms. This certificate is Copyright © 2019 Ultamation Limited.

Analysis Summary:

Missing Dependencies Lists all missing dependant modules.	100/100
Incomplete Symbols Lists all symbols that are incomplete, which will prevent compilation.	100/100
Duplicate Crosspoint IDs Lists any cross-point symbols which conflict with a previous use.	100/100
Symbols With Fatal Parameter Values Lists symbols where unchecked parameter values can cause the processor to crash.	100/100
Program Slot Missing Indicates the program header is missing essential information.	100/100
No Driving Source Lists all signals with no driving source	93/100
Multiple Driving Sources Lists all digital signals with multiple driving sources where one or more of the source symbols do not allow 'jamming'	100/100
Duplicate Buffer Outputs Lists all buffers where a signal is duplicated on the outputs.	100/100
Suppressed, Yet Connected Lists all signals which are supressed for connection warnings, but are actually connecting two or more symbols	100/100
Serial to MSP Only Lists all serial signals with only MSP as the destination	93/100
Redundant Forced Signals Lists all signals which ONLY lead to a force digital[analog]serial.	100/100
Symbols Where Output Drives Input Lists certain symbols where the only output is also an input [AND, OR, NOT, XOR, XNOR]	100/100
Looped RAM Signals Lists all signals which appear only on the same input and output signals of a RAM symbol	100/100
No Destination Lists all signals with no destination	89/100
Program Header Completeness Indicates where the program header is lacking useful information.	100/100

Figure 1

What is "Static Analysis"?

Static analysis is a technique used in software development where code is evaluated without actually executing. This type of analysis can identify things such as non-conformity to standards, syntactic or referential errors and even identify certain patterns in programming which are considered obsolete, inefficient or poor practice.

SIMPLified™ 2 provides a static analysis capability for SIMPL programs which can be used to provide a measure of program aspects from completeness, to standards, to highlighting potential issues.

Analysers

The foundation of the grading system is based upon discrete analysers which are being regularly added to. Some analysers look at the program as a whole – such as program meta-data to ensure that programmers complete optional, yet useful, information that can sometimes be overlooked. Other analysers will consider program structure or naming conventions, or test for symbolic patterns that may be considered deprecated or indicate a potential error which would not be highlighted in the normal program compilation phase of Crestron® development.

The analysers currently fall into one of 3 categories:

- Global program analyser.
  This covers general program structure and meta-data.
- Symbol analyser.
  This looks at individual symbols, and potentially their near neighbours.
- Signal analyser.
  This looks at signals and their driving sources and destinations. Some analysers score against an absolute threshold while others score as a proportion of total program signals.

Each analysers implements a `Score(…)` method which takes a SIMPL program structure as its input, and returns a score and a list of potential issues that have been identified.

Each analyser is also associated with a group category. These categories are currently:

- Functionality.
  Where the measure reflects actual, or potential, functionality issues. There are certainly cases where, in knowledgeable hands, these tests may identify benign program elements (such as jamming signals safely) there is always the possibility of a less capable programmer picking up the program in the future and there are always ways to construct the program to avoid such patterns. Some analysers do have the capability to have symbols marked for exclusion by specifying "`[S2:EXCLUDE]`" in the extended comments.
- Completeness.
  This measure identifies program completeness, such as unconnected signals or items marked as outstanding.
- Excess.
  Identifies unnecessary bloat in a program, or logic that makes no contribution to the functionality of the program.
- Standards.
  These are tests that identify deviation from house standards or lack of completion of useful meta-data.

The Analysers Score

The score generated by each analyser is proportional to the size of the program under analysis and the number of potential issues identified.

The Category Score

These scores accumulate under each category to form a normalised (0.0-1.0) aggregate score. This provides a category score which offers the opportunity to make additional judgement calls over the summary score. For example, a poor score on 'standards' may be acceptable since it makes no contribution to system performance or functionality.

The Summary Score

Finally, we take the sum of all analyser scores. This is the overall program score which is used in the certificate grading. By taking an aggregate of all analysis scores, this means that any and all improvements will favourably impact on the overall score, though the impact may be less obvious for larger programs.

The analysis is targeted at SIMPL Windows programming for 3 and 4-Series processors only. While it can be used for older 2-Series programs, some of the analysis will be less-appropriate for the older platform.