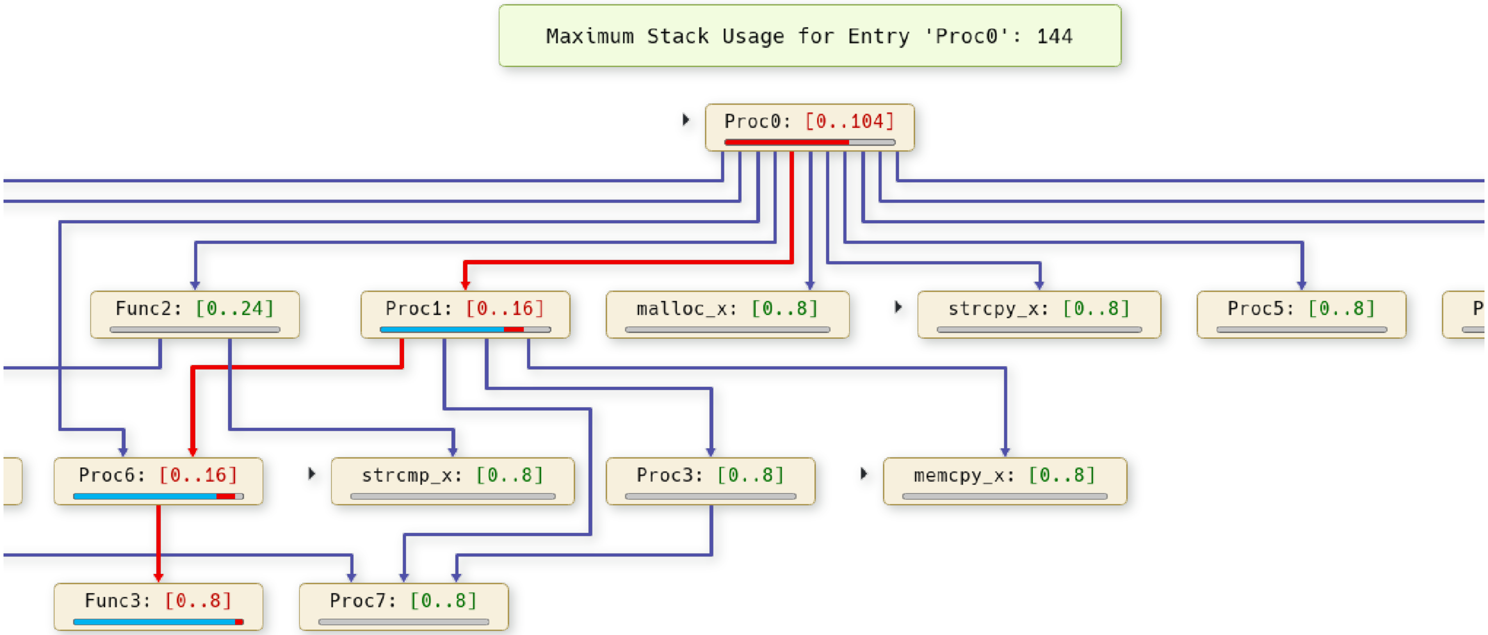
Static stack-usage analysis for ARM

StackAnalyzer for ARM statically determines the worst-case stack usage of tasks in safety-critical applications written or generated in C, C++ or Ada and compiled to run on 32-bit or 64-bit ARM processors.



Maximum Stack Usage for Entry 'Proc0': 144

Your benefits

- StackAnalyzer lets you prevent stack overflow for all possible inputs and task executions under any circumstances — without wasting hardware resources.
- StackAnalyzer requires no code instrumentation, no testing, no measuring, no modification of your system, no modification of your toolchain, and it will not be misled by potential flaws in debug information.
- Using StackAnalyzer is essential in meeting current safety standards such as ISO 26262, DO-178B/C, IEC-61508, and EN-50128, where statically analyzing your stack usage is part of the architectural safety requirements.

Key features

- Static analysis of binary files, exactly as they are executed in the final system.

- Clear and precise information on the worst-case stack usage by tasks, procedures, basic blocks, and individual instructions.
- Recursions, function pointers, inline assembly code, and library-function calls are all taken into account.
- Automatic recognition of dead code.
- Exceptionally fast analysis of complex real-world software.
- Freely selectable entry points for the analysis, so you can focus on the worst-case path or other areas of interest, and speed up the analysis even further.
- Difference analysis for comparing the stack usage before and after making changes to your system.
- State-of-the-art GUI with interactive views for analysis results, statistics, code coverage, control flow, source code, assembly code, DWARF debug info, symbol tables, analysis configuration, and more.
- Command-line mode and customizable XML reports for easy integration into automated build processes.
- Plugins for TargetLink and Jenkins.
- Seamless integration with other analysis tools from AbsInt — e.g. TimingProfiler for profiling the worst-case execution time.

Supported ARM families

- ARM7TDMI
- ARM9TDMI
- StrongARM
- ARM9E
- ARM10E
- XScale
- Cortex-A
- Cortex-M
- Cortex-R

Supported architecture variants and extensions

- ARMv4

- ARMv4T
- ARMv5T
- ARMv5TE
- ARMv6
- ARMv6T2
- ARMv7
- ARMv8
- ARMv9 up to and including ARMv9.3
- Advanced SIMD/“NEON”
- Thumb/Thumb-2
- VFP

For the purposes of licensing and pricing, StackAnalyzer for 32-bit ARM and StackAnalyzer for 64-bit ARM are treated as two separate products.

Also available

- Static WCET analysis for ARM
- Hybrid WCET analysis for ARM
- Timing profiling for Cortex
- Memory safety analysis for ARM
- Formally verified compilation for ARM and AArch64

Supported compilers

- ARM Developer Suite C/C++ compiler
- The formally verified CompCert compiler
- GNU C/C++ compiler (GCC)
- Green Hills MULTI for ARM (C, C++ or Ada)
- IAR C/C++ compiler
- KEIL MDK-ARM C/C++ compiler suite
- HighTec LLVM
- Other LLVM/Clang-based ARM compilers
- Tasking C/C++ compiler
- Texas Instruments C/C++ compiler
- Wind River Diab C/C++ compiler

The standard license covers one compiler of your choice. Additional compilers can be unlocked for a surcharge.

Qualification support

Your usage of StackAnalyzer for ARM can be qualified according to ISO 26262, DO-178B/C, and other safety standards. We offer Qualification Support Kits that help you simplify and automate your qualification process:

- Base QSK
- Optional compiler-specific add-on QSKs for
  - GCC 4.7.4
  - GCC 4.9.3
  - GCC 4.9.4
  - GHS 2019.1.4
  - GHS 2020.1.4
  - KEIL 3.1.0.939
  - KEIL 5.02.0.28
  - LLVM 10.0.1.1 Helix 23.06
  - TI 4.9.1
  - TI 20.2.1.lts

Compiler-specific QSKs for other compilers can be developed on request.

Recent improvements

21.10 22.04 22.10 23.04 23.10 24.10

- Improved PC-relative switch-table decoding
- Added a switch-table pattern that heuristically guesses the switch-table size
- The `UDF` instruction is now handled as a program end
- Improved handling of invalid guarded code blocks for Thumb

- See the complete release notes

System requirements

- Windows: 64-bit Windows 10 or 11
- Linux: 64-bit RHEL 9 or compatible
- 4 GB of RAM (16 GB recommended)
- 4 GB of disk space
- The Linux version requires the `libxcb-*` family of libraries to be installed
- Support for macOS is possible on request for a surcharge

Download
this factsheet
Download
StackAnalyzer flyer

Free trial

You can try StackAnalyzer for free, on your own applications, for a period of 30 days. Your free-trial package includes online training and tech support.

Get started today

© AbsInt.