面向功能安全的嵌入式开发工具

IAR Systems, Shanghai
Cynthia Hu

IAR SYSTEMS

议题

- IAR Systems公司介绍
- IoT嵌入式软件设计
- 功能安全认证
- C语言的安全性讨论
- C-STAT：静态代码分析工具
- C-RUN：动态代码分析工具

IAR Systems公司介绍

IAR Systems

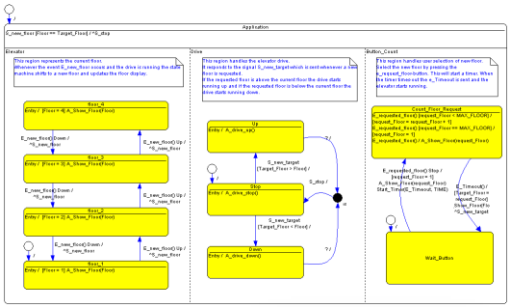
- IAR Systems公司成立于1983年
- 总部位于瑞典乌普萨拉
- 为全球30多个国家地区提供本地服务
- 支持多达10,000款微控制器
    - 支持3000多款 ARM芯片
- 全球领先的嵌入式开发工具供应商
- 主要产品
    - IAR Embedded Workbench: C/C++ 编译调试工具
    - IAR visualSTATE: 状态机建模和软件设计工具
    - IAR I-jet / I-scope / JTAGjet Trace: 跟踪调试器
- IAR Systems中国办事处：
    - 上海，021-63758658

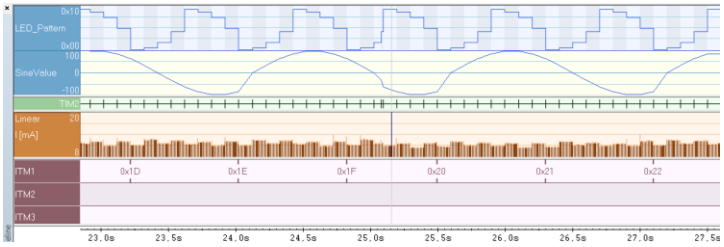
IoT嵌入式软件设计特点

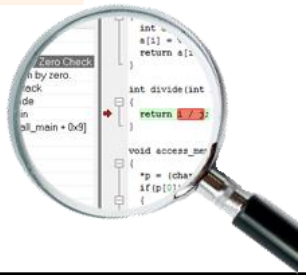
IAR SYSTEMS

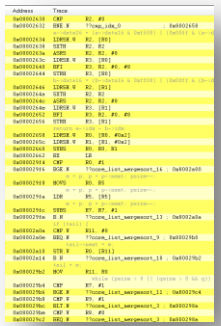
IoT嵌入式软件设计特点

IoT嵌入式软件设计特点	IAR工具提供的帮助
RTOS & Middleware	多种成熟RTOS partner方案和Debug Awareness技术
软件设计复杂度较高	建模工具
安全性需求（开放架构）	Functional Safety功能安全认证
可靠性要求高	静态分析和运行时分析工具
调试难度高	高级调试功能和Trace功能
低功耗要求	Power debugging技术
实时响应速度	速度优化
代码尺寸	代码尺寸优化









IAR Connect

Alliances in a smarter world

All relations are important to us; either they are with customers, partners or investors. With strategic alliances in the connected world, we are showing that we want do more as well as invest more in technology and offer more to our customers.

ALLIANCES

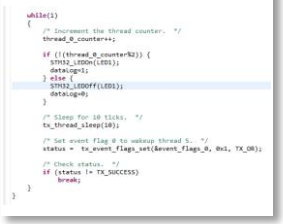
IAR Embedded Workbench for ARM

IAR SYSTEMS

IDE

eclipse READY

Editors



Source code control systems

SUBVERSION

PLUGINS

IAR Embedded Workbench IDE

IDE tools	Build tools	IAR C-SPY Debugger
Editor	IAR C/C++ Compiler	Simulator driver
Project manager	Assembler	Hardware system drivers
Library builder	Linker	Power debugging
Librarian		RTOS plug-ins

µC/Probe™ Run-Time Monitoring

Micrium

expresslogic

SEGGER freeRTOS eForce

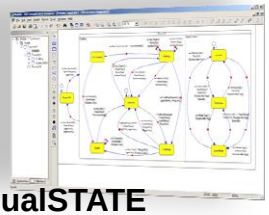
Micro Digital freescale semiconductor Quadros Systems Inc.

SCIOPTA CMX SYSTEMS WITTENSTEIN

Configuration tools

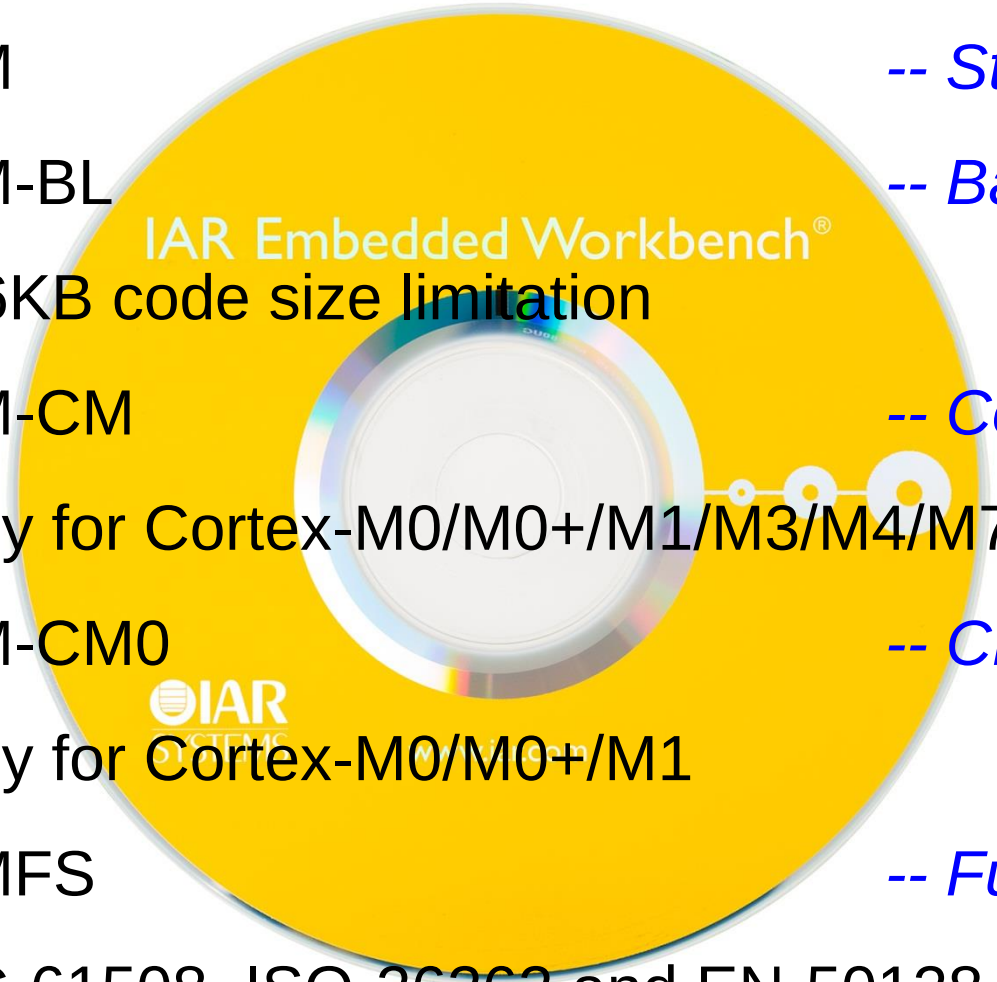
infineon DAVE™

freescale Processor Expert Software



visualSTATE

EWARM版本分类



- EWARM -- *Standard edition*
- EWARM-BL -- *Baseline edition*
  - 256KB code size limitation
- EWARM-CM -- *Cortex-M edition*
  - Only for Cortex-M0/M0+/M1/M3/M4/M7
- EWARM-CM0 -- *CM0/CM1 only*
  - Only for Cortex-M0/M0+/M1
- EWARMFS -- *Functional Safety*
  - IEC-61508, ISO-26262 and EN-50128

功能安全认证

IAR SYSTEMS

什么是功能安全?

- 所谓功能安全，简要来说，就是嵌入式系统在面临以下情况时，是否能够执行正确的操作并准确响应：
  - 错误的输入
  - 硬件故障
- 一般来说，以下这些行业对功能安全有高度需求：
  - 汽车行业
  - 航天电子
  - 医疗电子
- 当然，其他行业的应用方案也受益于功能安全设计

功能安全标准


IEC 61513
核電設備
IEC 60800
核電軟體
IEC 62061
機械設備
ISO 13849
機械設備
ISO 26262
道路車輛
IEC 61508 E/E/PE功能安全
IEC 60335-1
家電軟體
(附錄R)
IEC 60601
醫療儀器
EN 62304
醫療軟體
IEC 61800-5-2
電氣驅動系統
IEC 50156
熱爐設備
IEC 61511
製程工業

IEC 61508标准

IAR SYSTEMS

Table A.3 – Software design and development – support tools and programming language

(See 7.4.4)

	Technique/Measure *	Ref.	SIL 1	SIL 2	SIL 3	SIL 4
1	Suitable programming language	C.4.5	HR	HR	HR	HR
2	Strongly typed programming language	C.4.1	HR	HR	HR	HR
3	Language subset	C.4.2	---	---	HR	HR
4a	Certified tools and certified translators	C.4.3	R	HR	HR	HR
4b	Tools and translators: increased confidence from use	C.4.4	HR	HR	HR	HR

NOTE 1 See Table C.3.

NOTE 2 The references (which are informative, not normative) "B.x.x.x", "C.x.x.x" in column 3 (Ref.) indicate detailed descriptions of techniques/measures given in Annexes B and C of IEC 61508-7.

* Appropriate techniques/measures shall be selected according to the safety integrity level. Alternate or equivalent techniques/measures are indicated by a letter following the number. It is intended the only one of the alternate or equivalent techniques/measures should be satisfied. The choice of alternative technique should be justified in accordance with the properties, given in Annex C, desirable in the particular application.

IAR Systems功能安全认证工具

功能安全标准

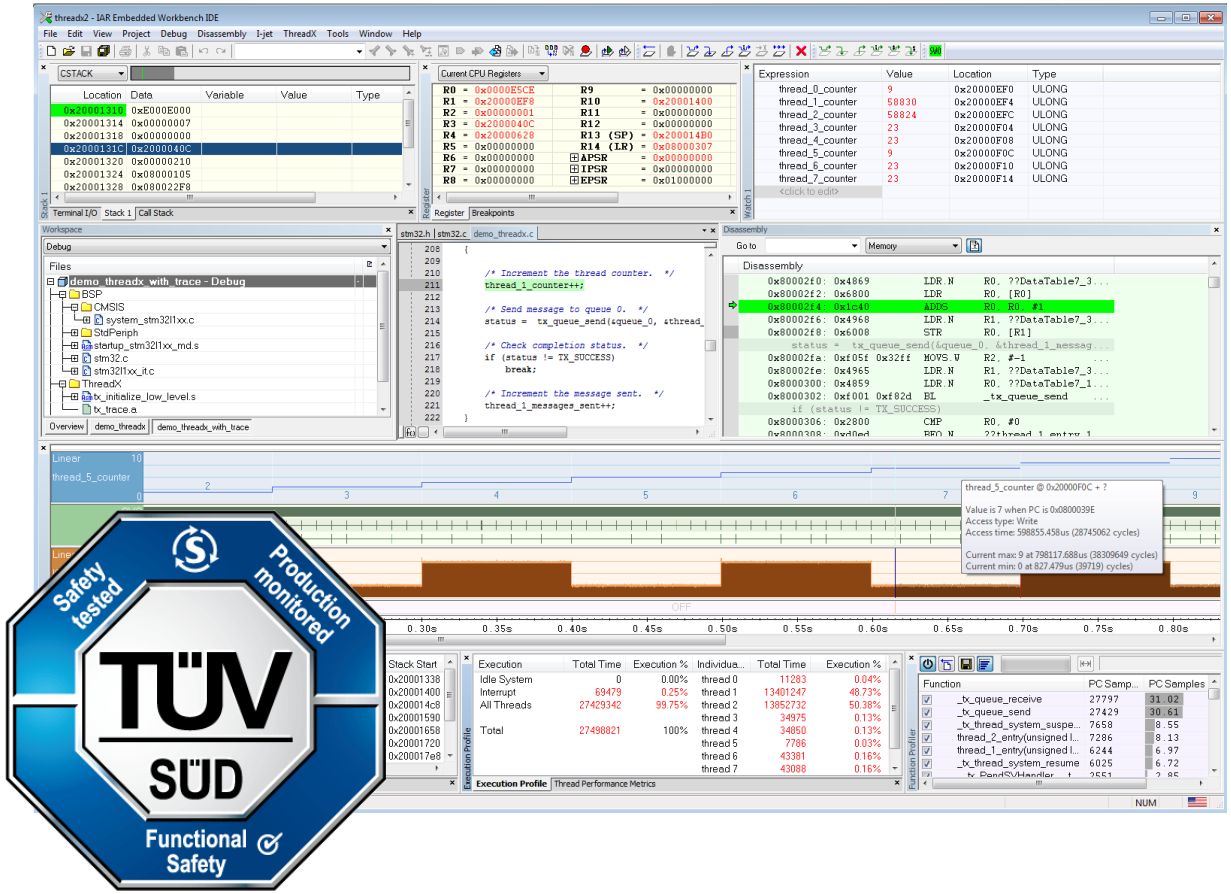
- IEC 61508
- ISO 26262
- EN 50128

功能安全认证版本提供

- 功能安全证书
- 认证报告
- 安全手册

可靠的支持和升级服务

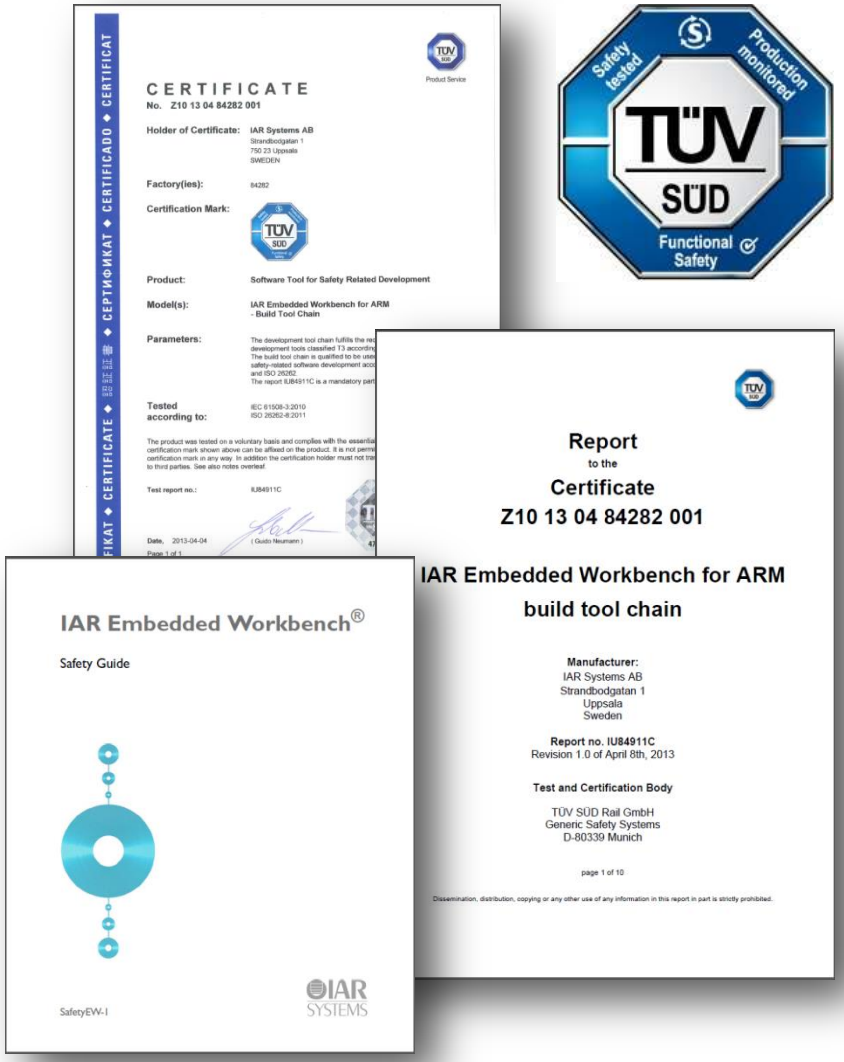
- 涵盖整个产品生命周期
- 高优先级的技术支持服务
- 经过认证的服务升级包
- 已知问题报告



www.iar.com/safety

EWARMFS

- EWARMFS
  - IAR Embedded Workbench for ARM 功能安全认证版本
- 通过TÜV SÜD认证
- 功能安全认证标准
  - IEC 61508-3:2010 (SIL 3)
    面向所有行业的电子，电气以及可编程系统
  - ISO 26262-8:2011 (ASIL D)
    由IEC 61508标准衍生出来的面向道路车辆的安全标准
  - EN 50128
    铁路控制和保护系统安全标准



C语言安全性的讨论

IEC 61508标准

IAR SYSTEMS

Table A.3 – Software design and development – support tools and programming language

(See 7.4.4)

	Technique/Measure *	Ref.	SIL 1	SIL 2	SIL 3	SIL 4
1	Suitable programming language	C.4.5	HR	HR	HR	HR
2	Strongly typed programming language	C.4.1	HR	HR	HR	HR
3	Language subset	C.4.2	---	---	HR	HR
4a	Certified tools and certified translators	C.4.3	R	HR	HR	HR
4b	Tools and translators: increased confidence from use	C.4.4	HR	HR	HR	HR

NOTE 1 See Table C.3.

NOTE 2 The references (which are informative, not normative) "B.x.x.x", "C.x.x.x" in column 3 (Ref.) indicate detailed descriptions of techniques/measures given in Annexes B and C of IEC 61508-7.

* Appropriate techniques/measures shall be selected according to the safety integrity level. Alternate or equivalent techniques/measures are indicated by a letter following the number. It is intended the only one of the alternate or equivalent techniques/measures should be satisfied. The choice of alternative technique should be justified in accordance with the properties, given in Annex C, desirable in the particular application.

选择合适的编程语言

IEC 61508-7标准表格 C.4.5给出了何为“合适的编程语言”的具体描述定义：

- 编程语言应该是定义清晰且无异议的

……

- 编程语言最好满足如下特性：
  - 可分解为易于管理的小型软件模块
  - 在软件模块之间限制数据的访问权限
  - 限制定义变量的使用范围
  - 其他各种可以限制错误发生的结构类型

编程语言的选择建议

在选择合适的C语言子集，遵循编程标准和使用静态代码分析工具的前提下，SIL所有的四个等级都高度推荐使用C编程语言。

Table C.1 – Recommendations for specific programming languages

9	C	R	–	NR	NR
10	C with subset and coding standard, and use of static analysis tools	HR	HR	HR	HR

NR: Not Recommended
HR: Highly Recommended

编程语言子集

IEC 61508-7标准表格 C.4.2给出了编程语言子集的目标定义和相关描述：

目标

- 降低引入编程错误的可能性
- 增加发现潜在编程错误的可能性

描述

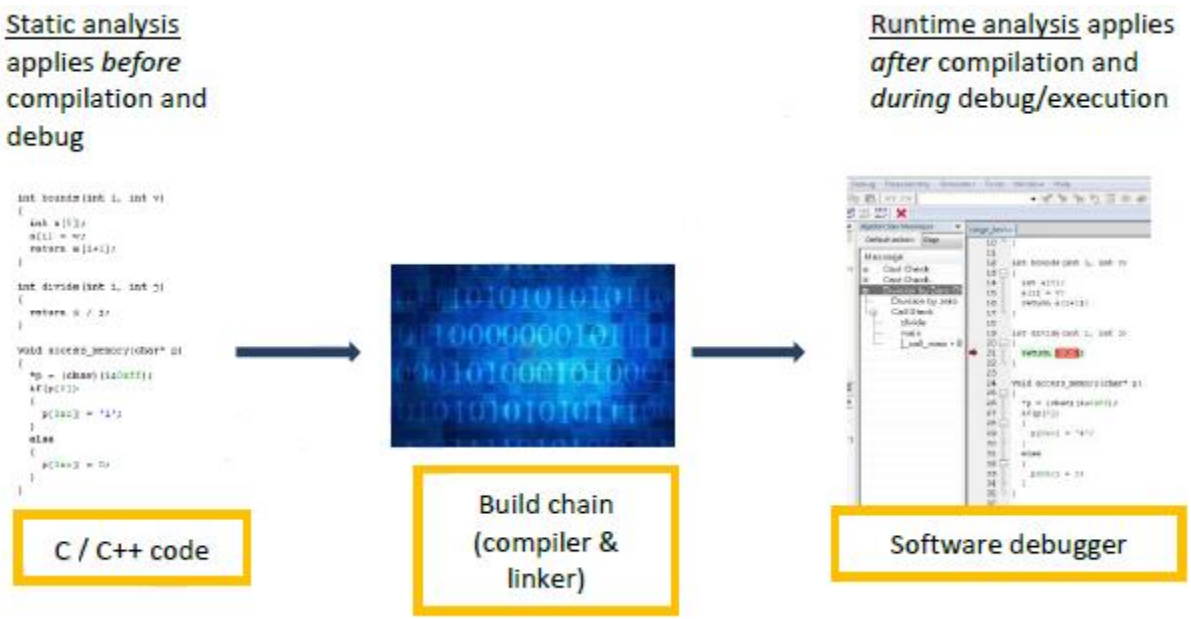
- 使用编程语言子集来排除一些容易犯错的结构类型
- 使用静态分析方法来发现编程语言的缺陷和错误

C-STAT: 静态代码分析

IAR SYSTEMS

代码分析工具

- 代码分析工具– 检测应用程序中的错误代码
  - 静态分析工具
    - 不执行目标代码的情况下分析代码
  - 运行时分析工具
    - 动态执行目标代码的过程中分析代码



C-STAT: 静态代码分析

- C-STAT是由IAR Systems自主开发的静态分析工具
  - 2015年2月首发
  - 支持C 和C++ 源码
- C-STAT是IAR Embedded Workbench的附加产品
  - 完全集成于EW
  - 无需额外安装程序
  - 不需要额外的license
  - 不支持第三方的编译和调试工具
- 软件类别和版本支持
  - IAR Embedded Workbench for ARM, from version 7.40以上
  - IAR Embedded Workbench for TI MSP430, from version 6.30以上
  - IAR Embedded Workbench for Atmel AVR32, from version 4.30以上
  - IAR Embedded Workbench for Renesas V850 4.20以上

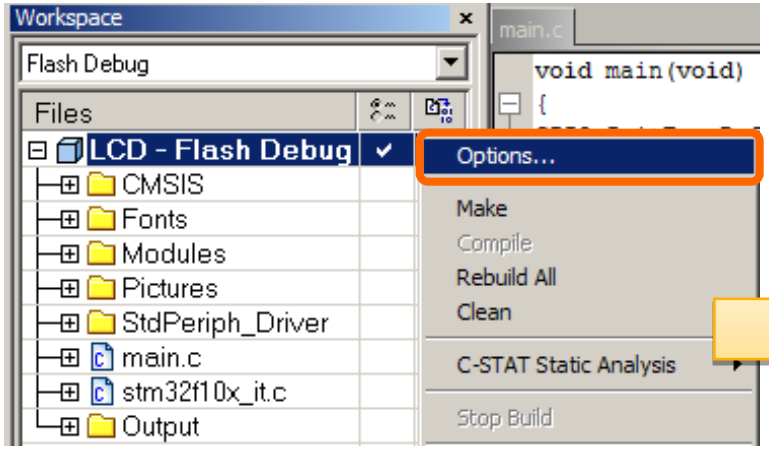
C-STAT检查规则

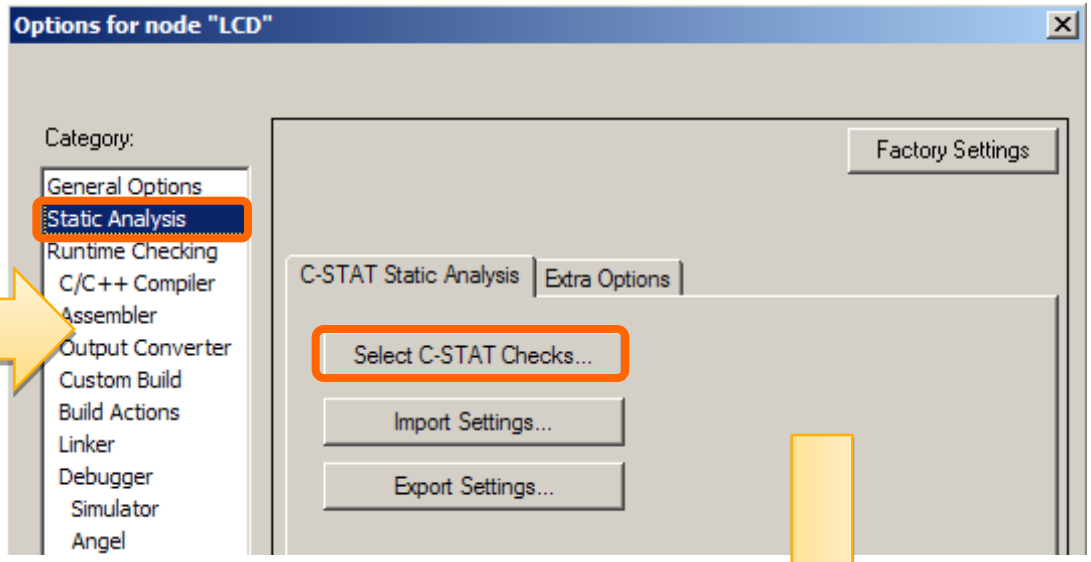
- Common Weakness Enumeration
- cwe.mitre.org
- An unified and measurable set of software weaknesses.
- Enumerate design and architecture weaknesses, as well as low-level coding errors.

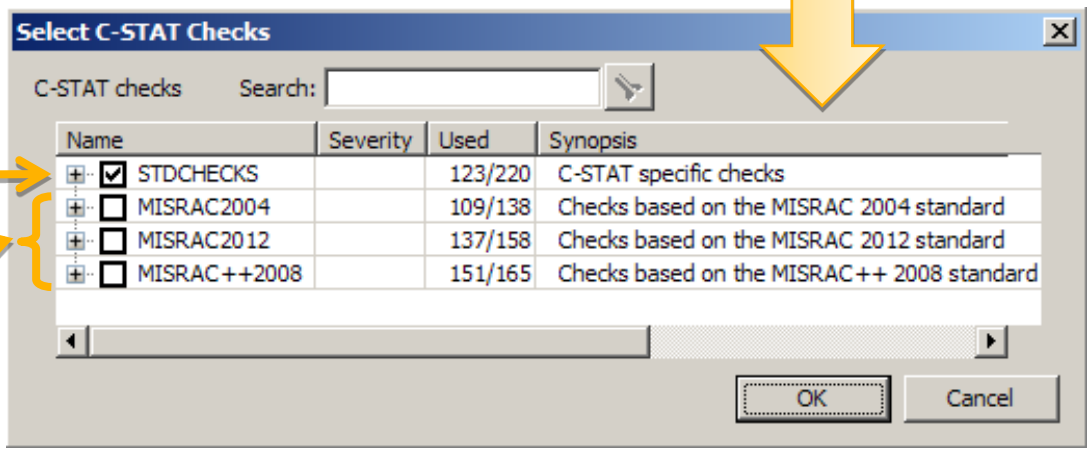
- Computer Emergency Response Team
- www.cert.org
- C/C++ secure coding standards, identifying insecure constructs which could expose a weakness or vulnerability in the software.
- Guidelines to avoid implementation, coding as well as low-level design errors.

- Motor Industry Software Reliability Association
- www.misra.org.uk
- MISRA C:2004 (MISRA C2): Identify unsafe code constructs in the C89 standard.
- MISRA C:2012 (MISRA C3): Extend the support to C99 version of the programming language whilst maintaining the guidelines for C89 standard.
- MISRA C++:2008: Identify unsafe code constructs in the 1998 C++ standard.

IAR EWARM C-STAT选项



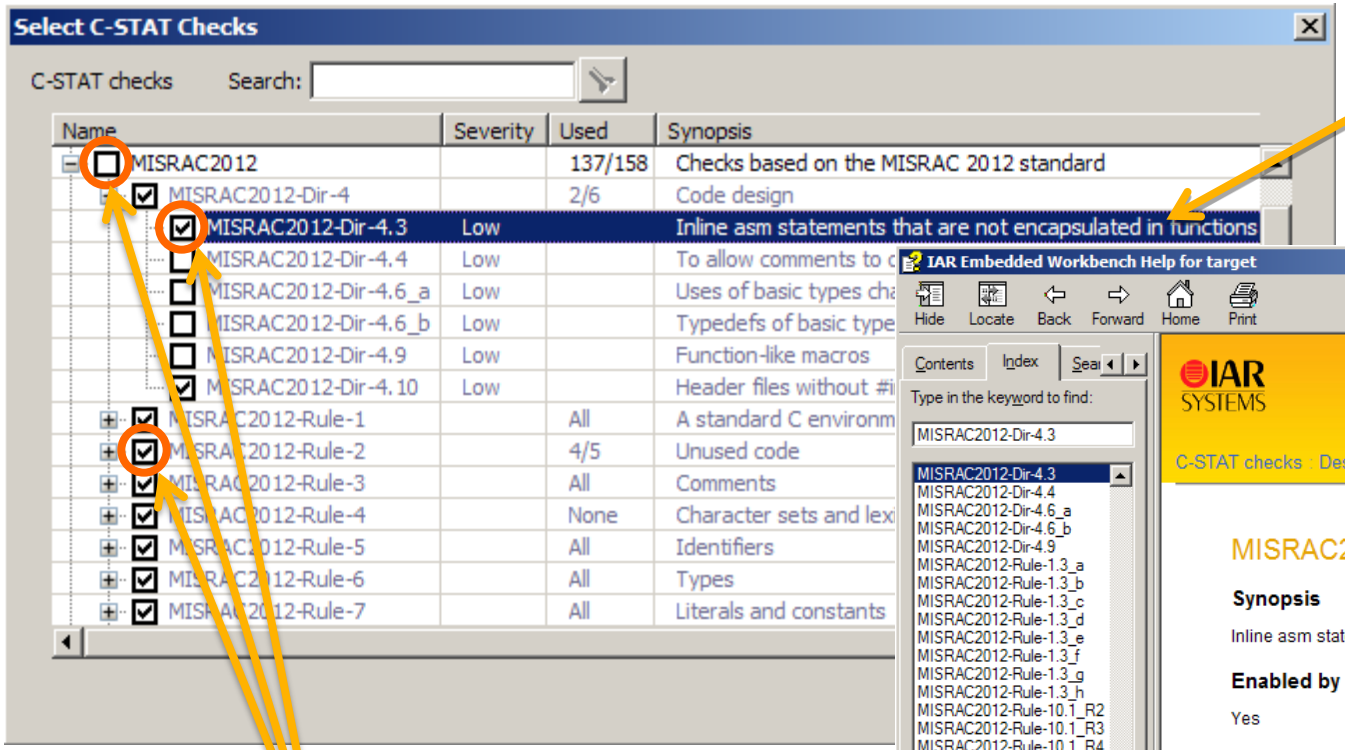




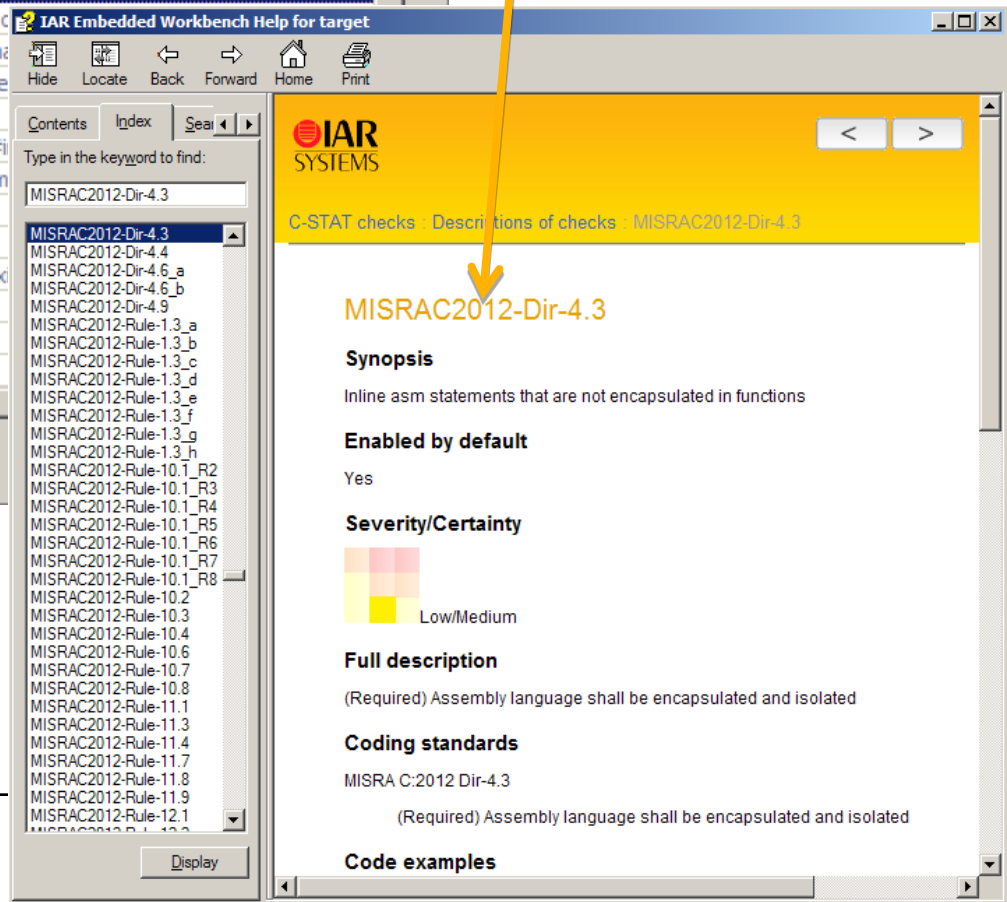
CWE/CERT rules

MISRA C/C++ rules

C-STAT: 规则配置

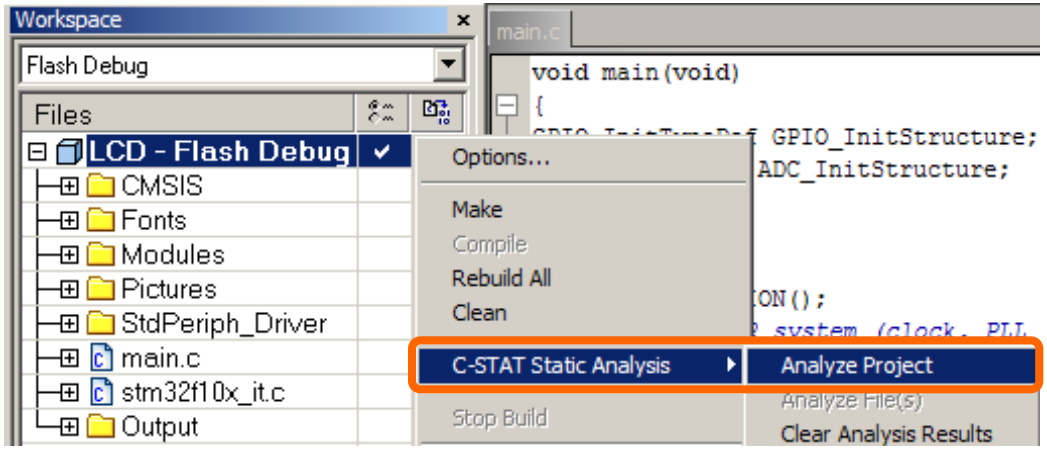


Highlight a rule and press F1 to show the detailed description.

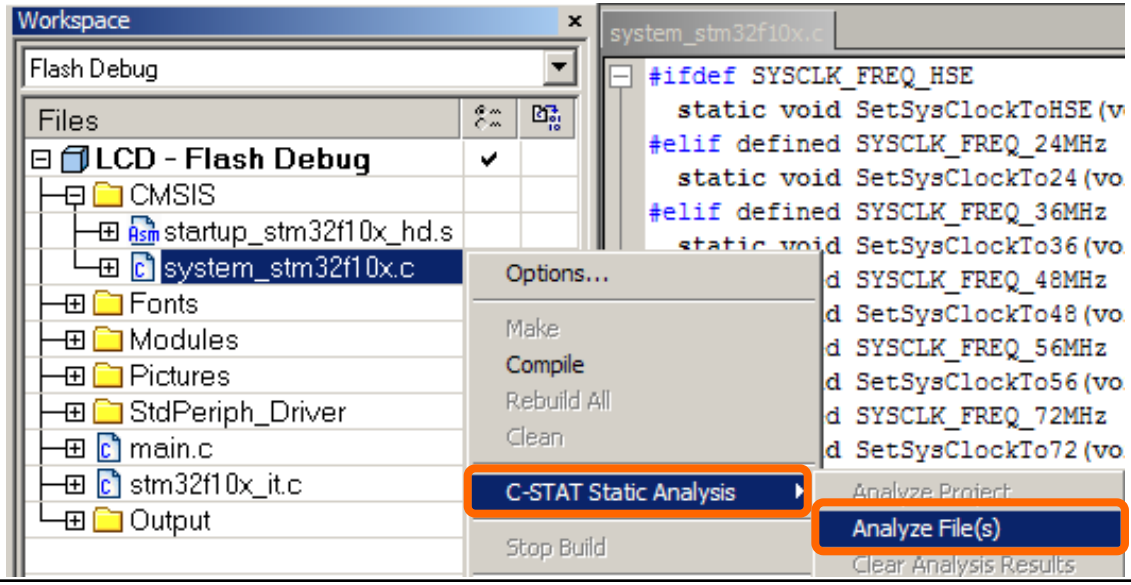


Enable or disable a set of rules or any individual rule.

C-STAT: 分析代码



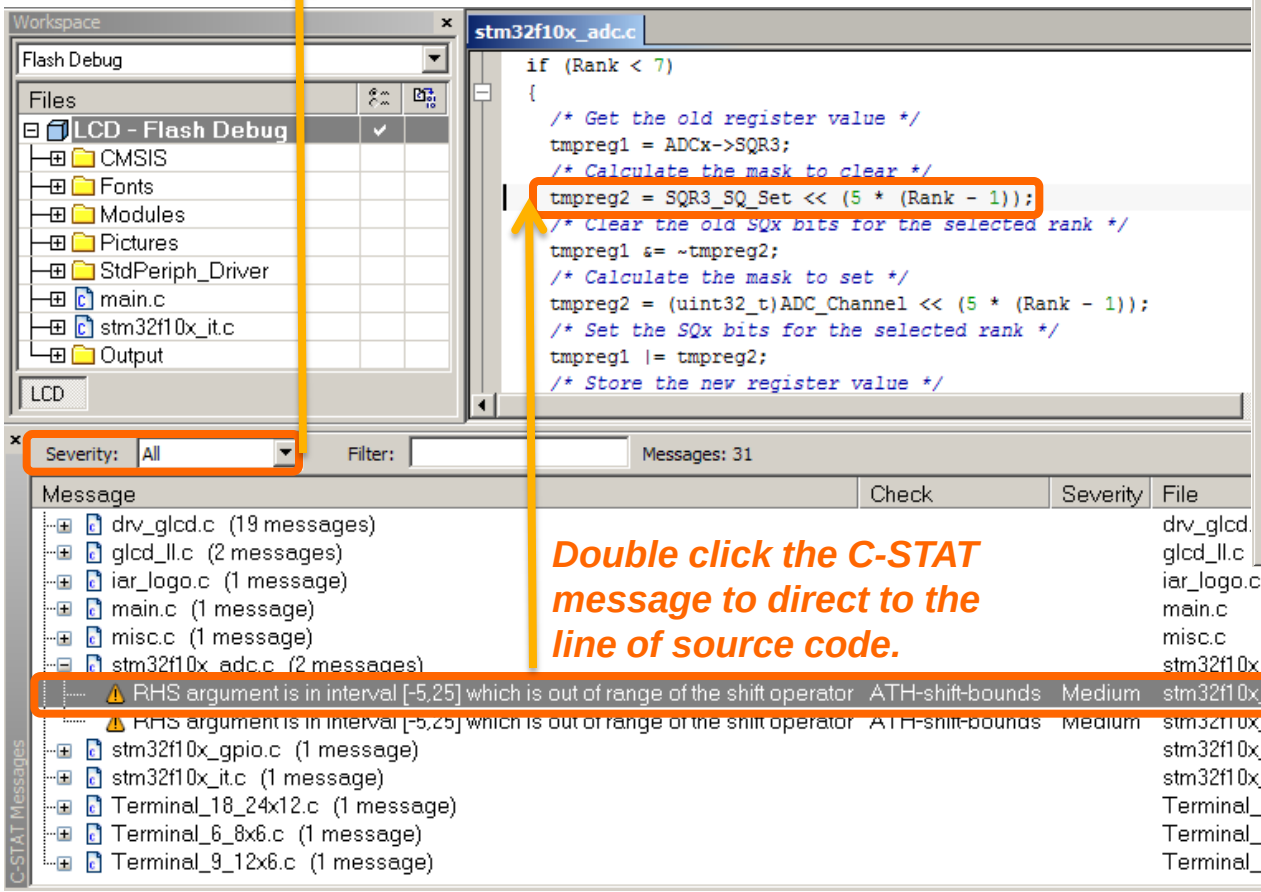
Analyze the whole project



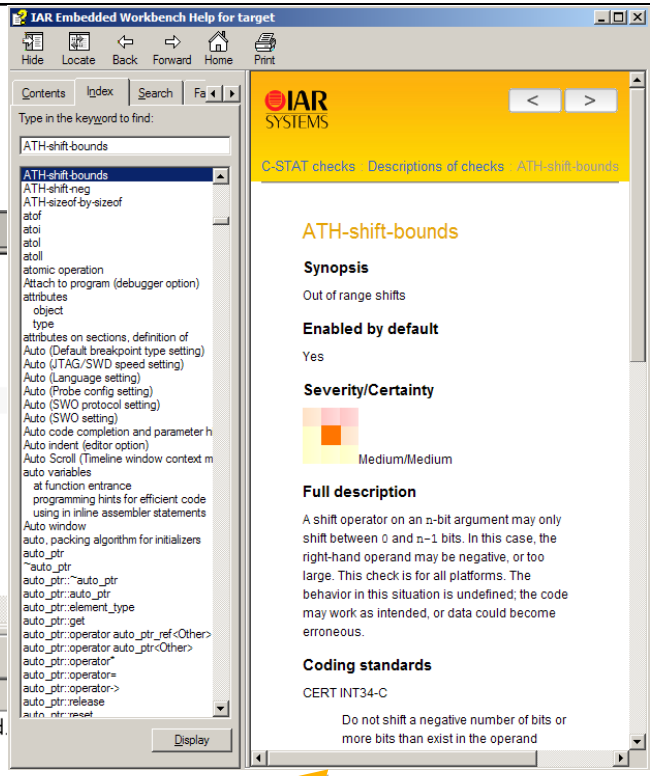
Analyze an individual source file or a group of source files

C-STAT: 分析结果

Filter the C-STAT messages by selecting a level of severity: All, Low, Medium or High.



Double click the C-STAT message to direct to the line of source code.

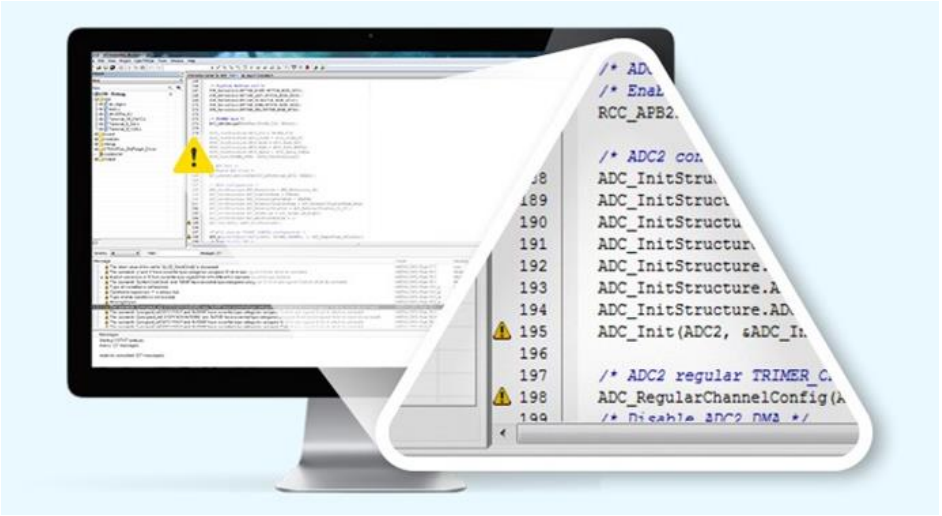


Highlight the C-STAT message and press F1 to show the related rules information.

C-RUN: 运行时分析工具

IAR SYSTEMS

IAR C-STAT和IAR C-RUN



C-STAT Static analysis

C-STAT performs advanced analysis of your C/C++ code and finds potential issues. It helps you improve your code quality as well as prove alignment with standards such as MISRA C:2012.

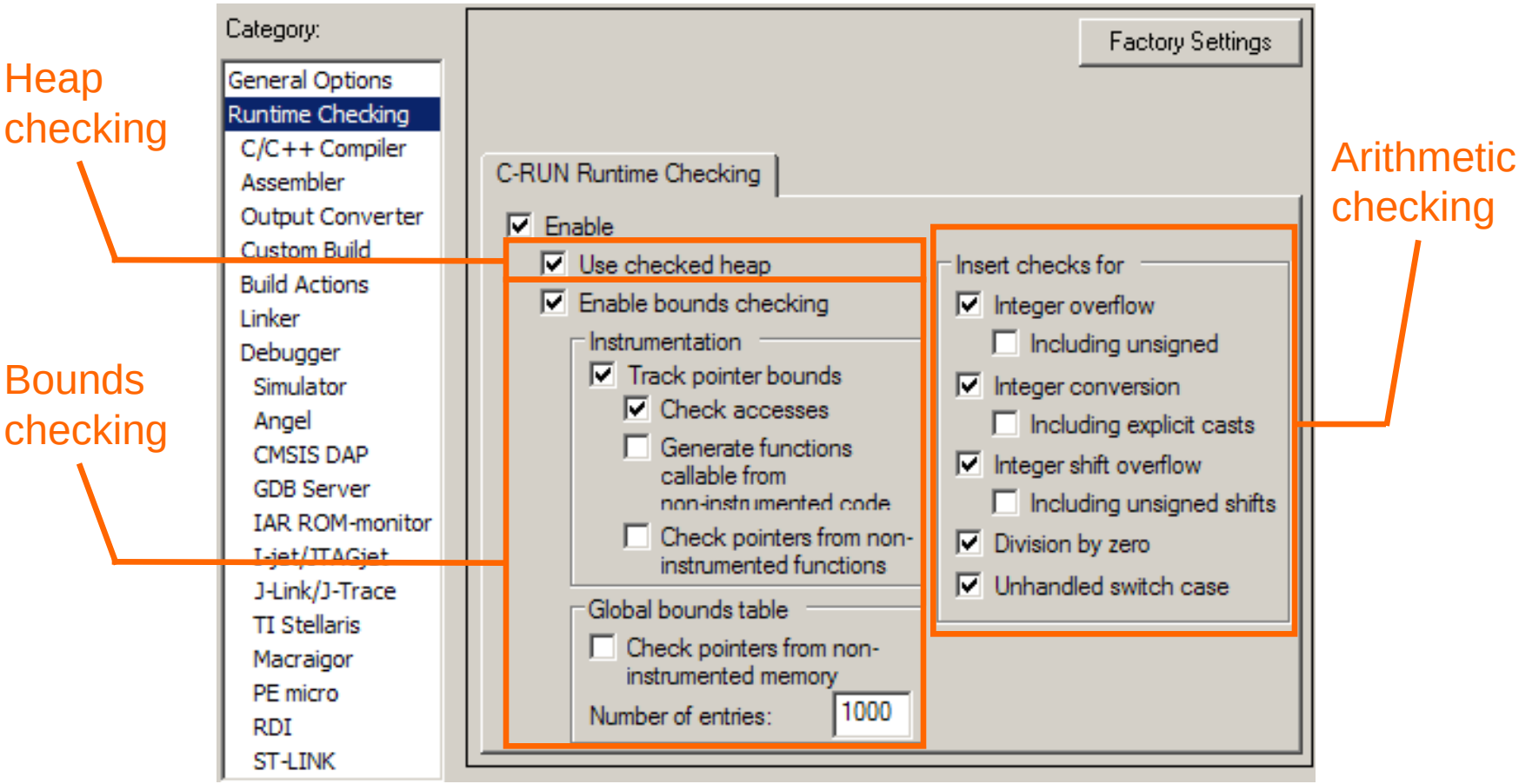
C-RUN Runtime analysis

C-RUN helps you find errors at an early stage. It is completely integrated with IAR Embedded Workbench for ARM, and provides detailed runtime error information.

C-RUN: Runtime code analysis

- C-RUN是由IAR Systems自主开发的动态分析工具
  - 2014年5月首发
  - 支持C 和C++ 源码
- C-RUN是IAR Embedded Workbench的附加产品
  - 完全集成于EW
  - 无需额外安装程序
  - 不需要额外的license
  - 不支持第三方的编译和调试工具
- 目标支持
  - IAR Embedded Workbench for ARM, from version 7.20以上
  - 支持所有的ARM内核

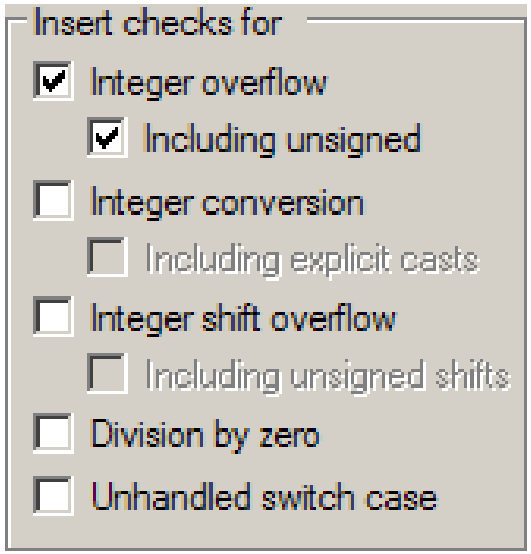
IAR EWARM C-RUN选项

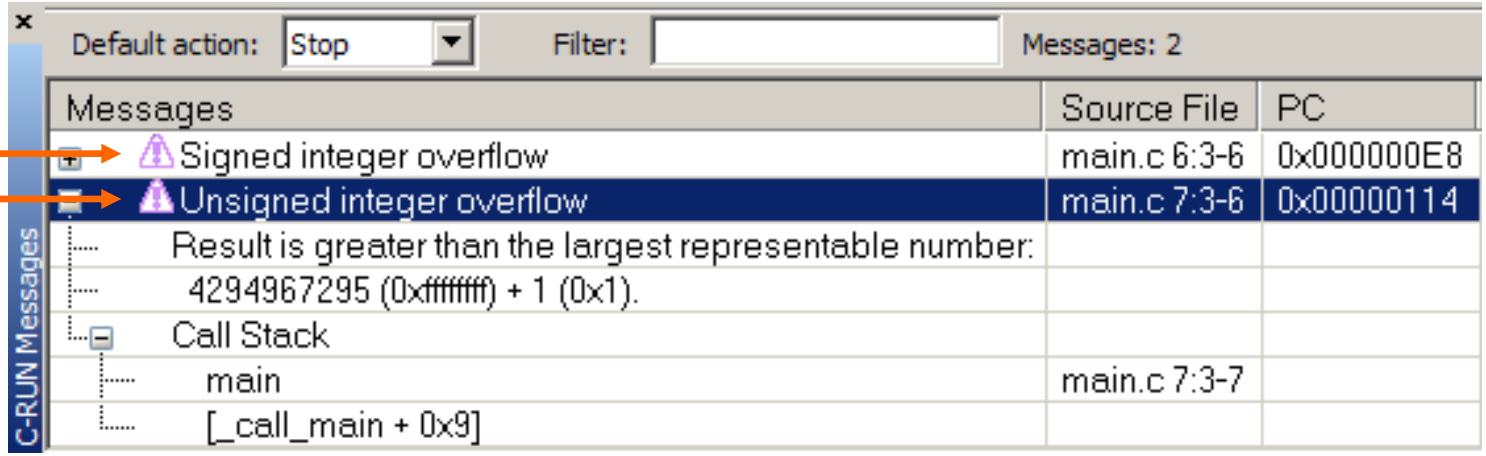


检测整型溢出

```
void main (void)
{
  int v1 = 0x7fffffff;
  unsigned int v2 = 0xffffffff;

  v1++; /* signed integer overflow */
  v2++; /* unsigned integer overflow */
}
```

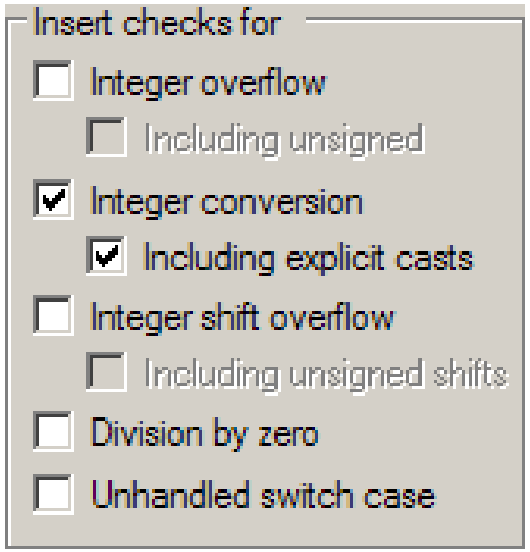


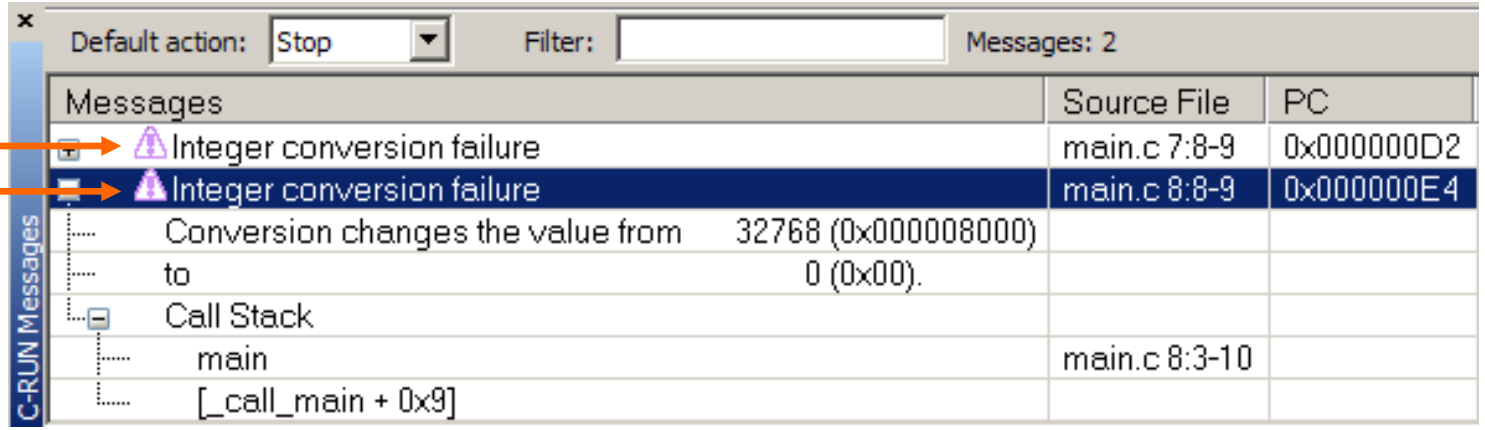


检测整型转换

```
void main (void)
{
  int v1 = 0x8000;
  short v2;
  char v3;

  v2 = v1; /* 32-bit → 16-bit */
  v3 = v1; /* 32-bit → 8-bit */
}
```

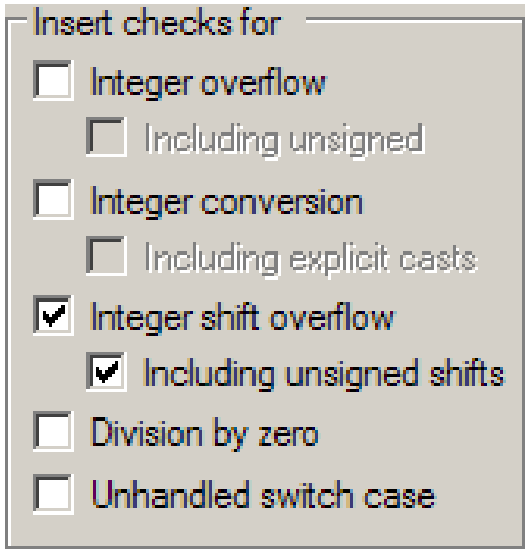


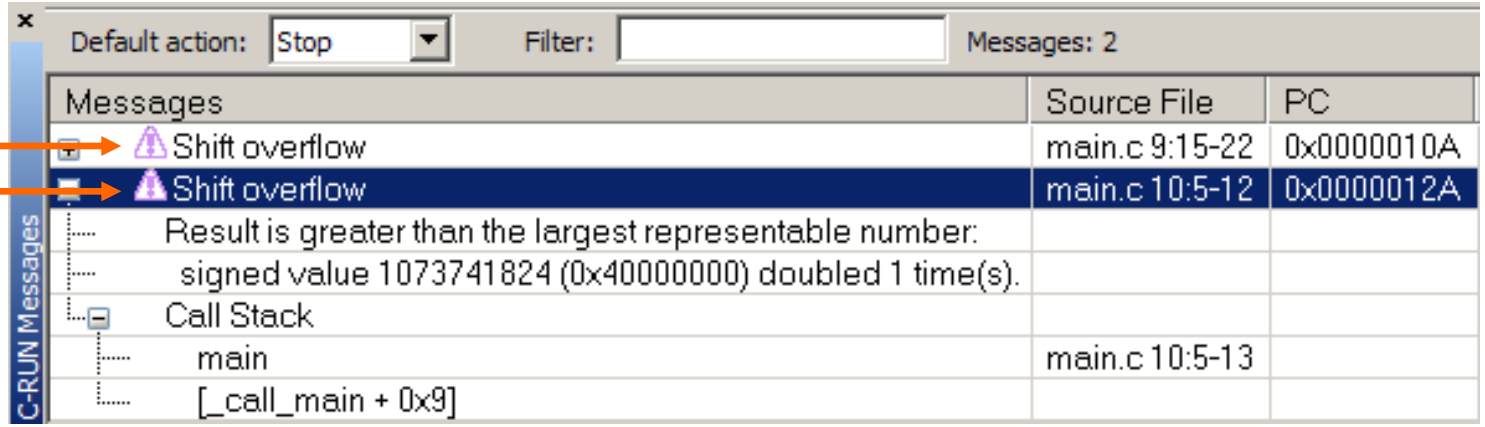


检测移位溢出

```
void main (void)
{
  int i;
  short v1 = 1;
  int v2 = 1;

  for (i=0; i<32; i++)
  {
    v1 <<= 1; /* overflow: i>14 */
    v2 <<= 1; /* overflow: i>30 */
  }
}
```

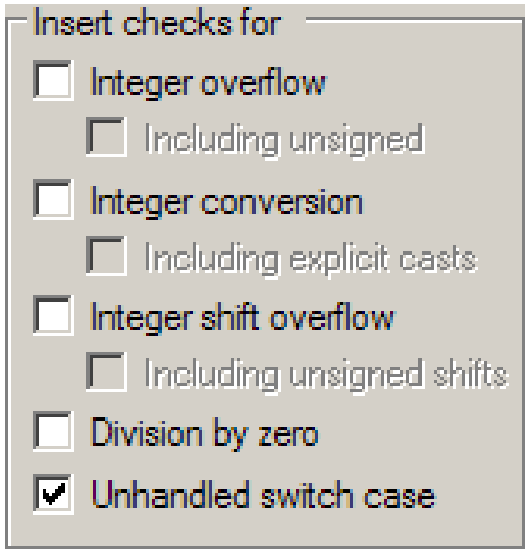


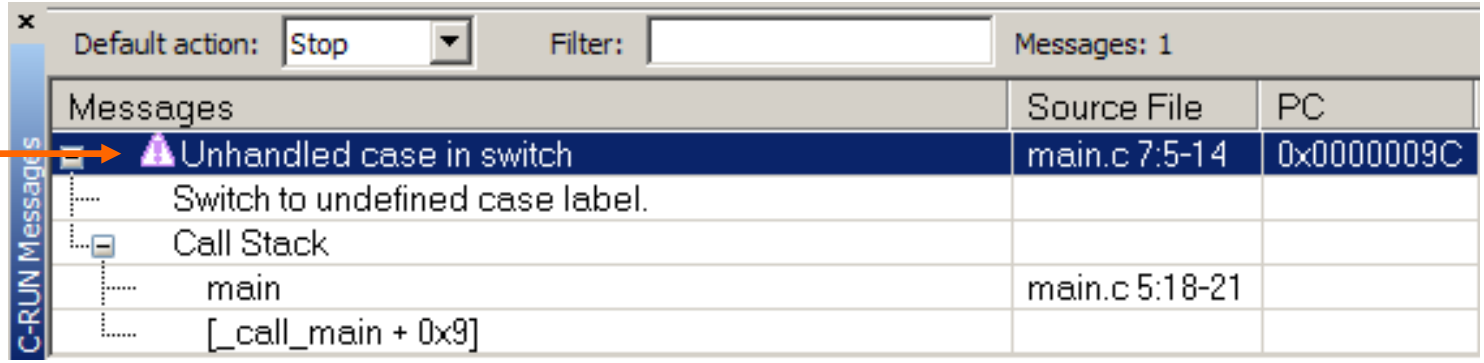


检测遗漏的switch-case语句

```
void main (void)
{
  int i;

  for (i=0; i<2; i++)
  {
    switch (i) /* case 1 is not handled */
    {
      case 0:
        break;
    }
  }
}
```



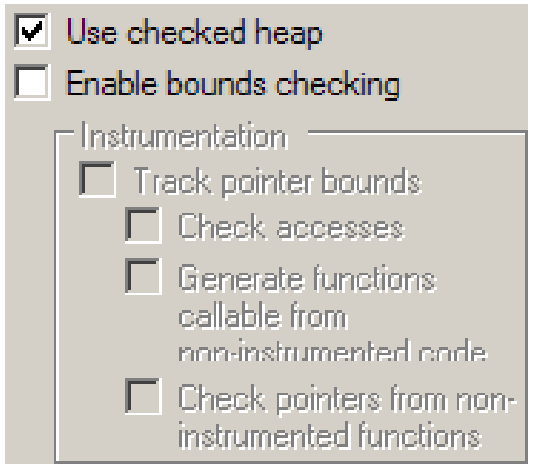


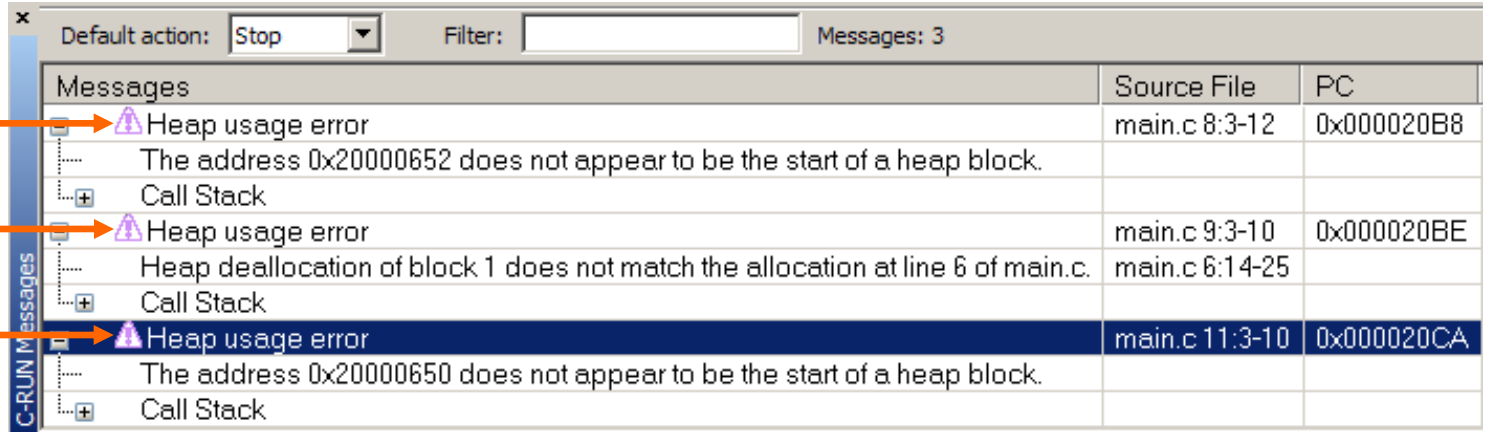
检测heap错误 - 1

```c
#include <stdlib.h>

void main (void)
{
  char *c1 = (char *)malloc(10);
  char *c2 = new char[10];

  free(c1+2); /* not the start of a block */
  free(c2);   /* non-matched new and free */
  free(c1);
  free(c1);   /* free a block more than once */
}
```





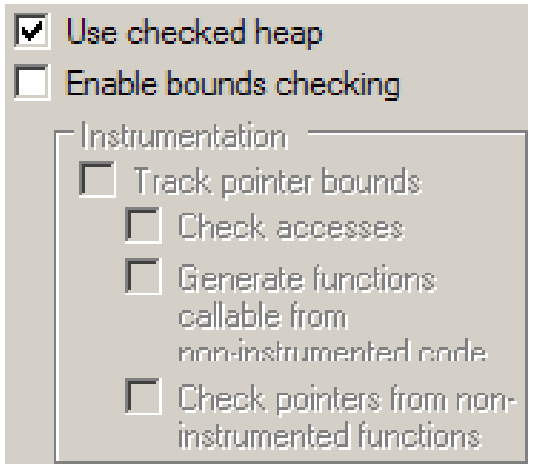
检测heap错误 - 2

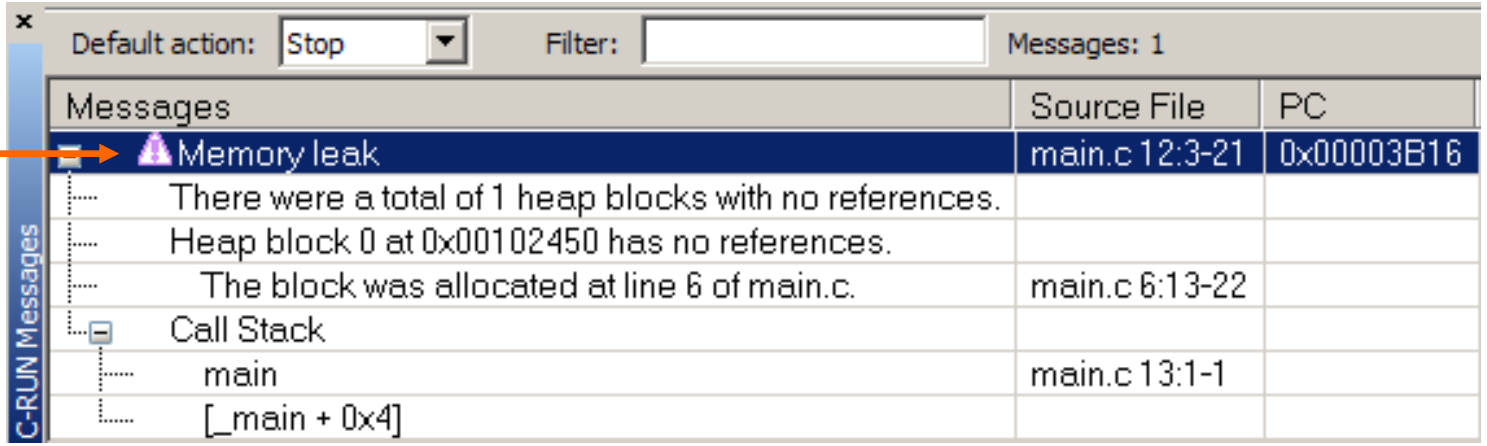
```c
#include <stdlib.h>
#include <iar_dlmalloc.h>

void main (void)
{
  char *c = malloc(10);
  c = malloc(20);              /* memory leak */

  free(c);

  /* check for memory leaks, manually called */
  __iar_check_leaks();
}
```



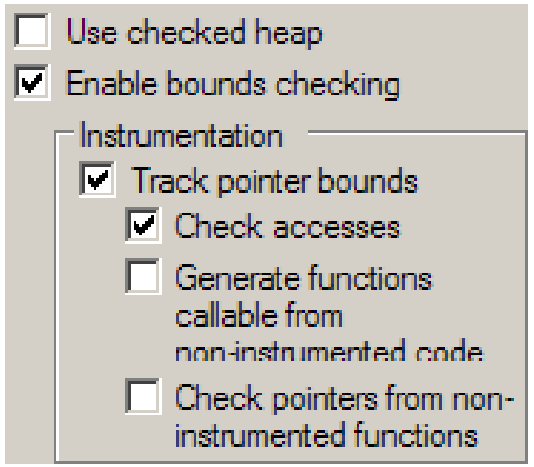


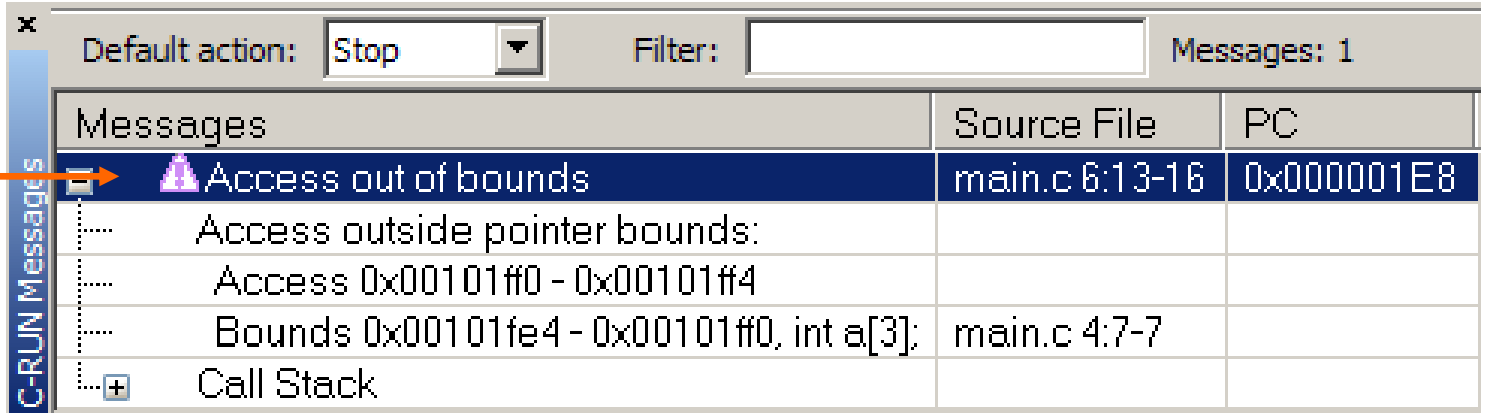
检测越界访问

```
int main (void)
{
  int i, j;
  int a[3] = {1, 2, 3};

  for (i=0; a[i]!=0; i++) /* out of bounds */
  {                       /* when i==3     */
    j = a[i];
  }

  return j;
}
```

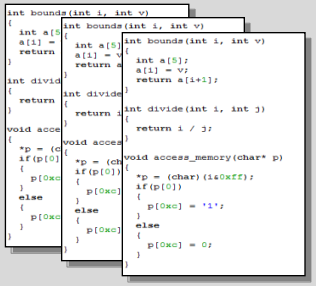




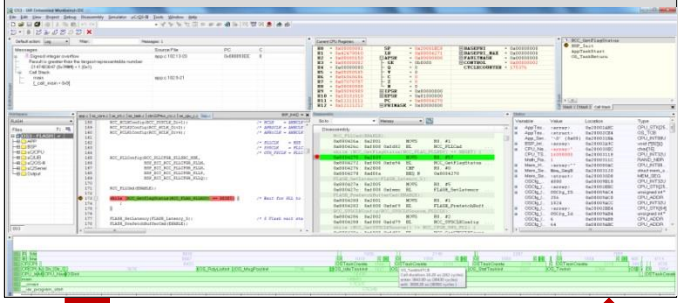
优化你的开发流程

IAR SYSTEMS

Implement your design in code

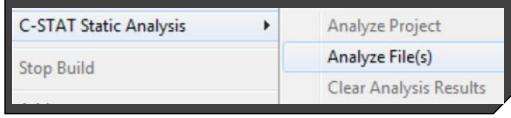


Build and debug the application

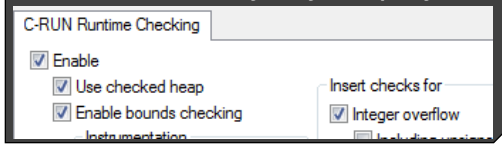


Release the application

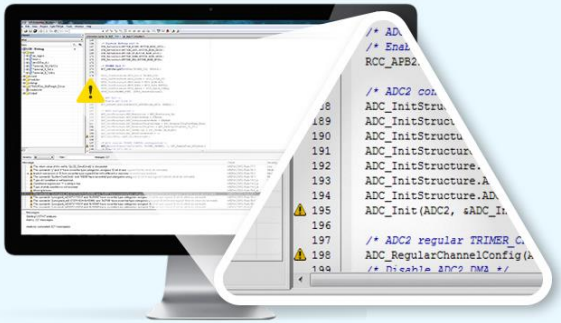
Let C-STAT analyze your code



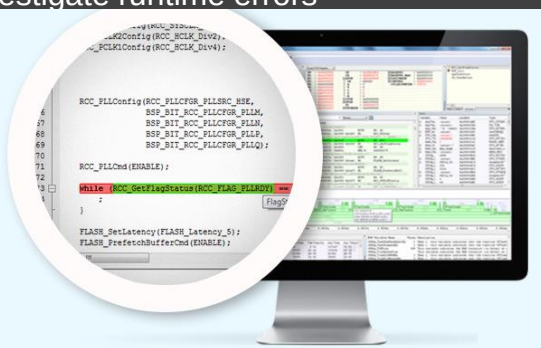
Let C-RUN analyze your project



Review potential issues



Investigate runtime errors



Requirements → Design → Implementation → Verification → Maintenance

IAR Systems: 值得信赖的战略伙伴

- 多种架构，一站式解决方案
- 高效高性能代码
- RTOS & Middleware集成
- 高级跟踪调试
- 功耗调试
- C-STAT静态代码分析
- C-RUN运行时代码分析
- 堆栈分析和跟踪
- 功能安全认证
- 全球专业技术支持