ICS 43.040

T 35

团 体 标 准

T/JSSAE 011—2025

汽车电子功能安全软件测试及评价方法

Testing and evaluation methods for automotive electronic functional safety software

2025-11-19 发布 2025-11-30 实施

江苏省汽车工程学会 发 布

目　　次

前　言

本文件按照GB/T 1.1—2020《标准化工作导则　第1部分：标准化文件的结构和起草规则》的规定起草。

请注意本文件的某些内容可能涉及专利。本文件的发布机构不承担识别专利的责任。

本文件由江苏省汽车工程学会提出。

本文件起草单位：中汽院（江苏）汽车工程研究院有限公司、中国汽车工程研究院股份有限公司、重庆青山工业有限责任公司、重庆大学、江苏天安智联科技股份有限公司、无锡车联天下信息技术有限公司、延锋伟世通电子科技（南京）有限公司。

本文件主要起草人：雷剑梅、孟璋劼、卢凡、韩松、范开伟、余处和、田甜、李庆庆、杨雷、韩庆文、龚静、陈冬梅、谭凯、赵泽蛟、刘柳、宋朴萱、王维、陈婉枫、张瑞平、雷灿、谭浩然、姚银花。

本文件为首次发布。

汽车电子功能安全软件测试及评价方法

1 范围

本标准规定了汽车电子功能安全软件在单元验证阶段、集成验证阶段及嵌入式软件测试阶段的测试及评价方法，用于指导汽车电子功能安全软件测试及评价的具体实施。

本标准适用于按照 GB/T 34590.6-2022 开发的软件，不是按照 GB/T34590.6-2022 开发的软件可参照执行。

2 规范性引用文件

下列文件中的内容通过文中的规范性引用而构成本文件必不可少的条款。其中，注日期的引用文件，仅该日期对应的版本适用于本文件；不注日期的引用文件，其最新版本（包括所有的修改单）适用于本文件。

GB/T 34590.1-2022 道路车辆 功能安全 第1部分：术语

GB/T 34590.6-2022 道路车辆 功能安全 第6部分：产品开发：软件层面

3 术语和定义

GB/T 34590.1-2022界定的以及下列术语和定义适用于本文件。

3.1

功能安全软件 functional safety software

按照GB/T 34590.6-2022开发的，ASIL等级为A至D的软件。

3.2

结构覆盖率 structural coverage

度量测试用例执行对程序源代码内部逻辑结构（如语句、分支等）覆盖程度的指标。

3.3

分支覆盖率 branch coverage

在测试中，已执行计算机程序的控制流分支所占的比率。

注1:100% 分支覆盖率意味着100% 语句覆盖率(3.160)，反之则不成立。

注2:一个 if 语句总有两个分支，即条件为 true 和条件为 false，不依赖于 else 语句是否存在。

[来源：GB/T 34590.1-2022，3.13，有修改]测试用例执行对程序源代码内部逻辑结构（如语句、分支等）覆盖程度的指标。

3.4

语句覆盖率 statement coverage

软件中已执行语句所占的百分比。

[来源：GB/T 34590.1-2022，3.160]

3.5

修改条件/判定覆盖率 modified condition/decision coverage; MC/DC

在控制流中已执行的、可以独立影响判定结果的全部单一条件结果的百分比。

注:MC/DC 是一种建立在分支覆盖率之上的代码覆盖率分析。因此,它也要求所有的代码块和所有的执行路径都经过测试。

[来源：GB/T 34590.1-2022，3.92]

3.6

走查　walk-through

为了发现安全异常,对工作成果的系统性检查。

注 1:走查是验证的一种方法，包括代码走查及模型走查。

注 2:走查和测试的区别在于,走查通常不涉及相关项或要素的运行。

注 3:被发现的任何异常通常通过重做来处理，并对重做的工作成果进行走查。

示例:在走查过程中,开发者向一个或多个评审员逐步地阐述工作成果。其目的是建立对工作成果的共同理解和识别工作成果中的任何安全异常。审查和走查均属于同级评审,其中走查的严格性弱于审查。

[来源：GB/T 34590.1-2022，3.182，有修改]

3.7

审查　inspection

为发现安全异常而依据一个正式的流程对工作成果进行的考查。

注 1:检查是验证的一种方式，包括代码审查及模型审查。

注 2:检查不同于测试检查通常不包括对相关项或要素的操作。

注 3:一项正式的检查流程通常包括预先定义的步骤、检查列表、核对人员及对结果的评审。

[来源：GB/T 34590.1-2022，3.82，有修改]

3.8

故障注入　fault injection

评估要素内故障影响的方法,该方法通过注入故障、错误或失效从而通过观测点对注入后的响应进行观测。

注:故障注入可以在不同的抽象层面进行,包括相关项层面或要素层面,这取决于范围、可行性、可观测性和所需细节的层面。取决于目的,可以在安全生命周期的不同阶段,考虑不同的故障模型,进行故障注入。

示例 1:在运行期间注入故障以验证作为探测潜伏故障策略一部分的某个安全机制正在正常工作。

示例 2:在进行集成测试时,通过硬件调试端口或专用软件命令注入故障来测试软硬件接口。示例 3:在硬件组件层面对卡滞故障或瞬态故障进行仿真,以验证安全机制的诊断覆盖率,或识别出可引起错误或失效故障。

[来源：GB/T 34590.1-2022，3.57]

4　软件单元验证

4.1　模型单元验证

4.1.1　模型静态分析

4.1.1.1　测试方法

应在模型在环测试环境中，使用模型静态分析工具对被测软件模型进行静态扫描，以检查建模规范符合性和运行时错误，并统计建模规范符合性缺陷及运行时错误的数量。

注 1：选用的模型静态分析工具应具备控制流分析、数据流分析、形式化验证等能力（参见 GB/T 34590.6-2022 中相关要求）。

注 2：常见的建模规范有 MathWorks Automotive Advisory Board、Matlab-High -Integrity-Modelling-Guideline 等，企业可在参考行业通用规范（如前述）的基础上，依据具体项目需求，定义或裁剪适用的建模规范集。

注 3：常见的模型运行时错误有死代码、数组越界、整数或定点数据溢出、除数为零等错误。

4.1.1.2 评价方法

完成 4.1.1.1 测试后，应满足：

(a)运行时错误的数量为 0；

(b)建模规范符合性缺陷的数量为 0。

4.1.2 模型质量度量

4.1.2.1 测试方法：

应在模型在环测试环境中，采用模型度量工具对被测软件模型进行静态扫描，统计以下指标：

a) 子系统数量；

b) 子系统深度；

c) 模块数量；

d) 圈复杂度；

e) 输入/输出接口数量。

注 1：企业可根据自身的开发需求对模型度量指标进行扩充及调整。

4.1.2.2 评价方法：

完成 4.1.2.1 测试后，度量的指标应符合企业定义的阈值要求。

示例：子系统的数量≤4，子系统深度≤4，模块的数量≤500、圈复杂度≤35、输入接口数量≤20、输出接口数量≤10。

4.1.3 模型走查

4.1.3.1 测试方法：

应由模型走查测试人员与被测模型的开发人员面对面沟通，依据软件单元需求，逐条验证被测软件模型的逻辑和完整性。

注：应根据软件单元需求提前设计模型走查用例，提高模型走查的全面性及效率。

4.1.3.2 评价方法：

完成 4.1.3.1 测试后，走查的通过率应为 100%。

4.1.4 模型审查

4.1.4.1 测试方法：

应由项目经理、开发人员、测试人员、质量保证人员等组建的模型审查专项小组设计模型审查单，逐条验证被测软件模型是否符合模型审查单的要求。

注：模型审查单示例见附录 A，企业可根据自身的开发需求，对模型审查单进行调整及补充。

4.1.4.2 评价方法：

经过 4.1.4.1 测试后，审查的通过率应为 100%。

4.1.5 基于需求的测试

4.1.5.1 测试方法：

应根据软件单元设计规范和分配给软件单元的软件安全要求，分别设计正常及异常的测试用例，在模型在环测试环境中，采用模型动态测试工具对软件模型进行测试，以验证被测软件模型与需求的一致性。

注 1：应将需求分解至原子需求。

注 2：一般采用开环的测试方法对软件模型进行测试。

4.1.5.2 评价方法：

经过 4.1.5.1 测试后，需求覆盖率应为 100%，测试用例通过率应为 100%。

4.1.6 接口测试

4.1.6.1 测试方法：

应根据软件单元的接口设计规范，分别设计正常及异常的测试用例，在模型在环测试环境中，采用模型动态测试工具，对软件模型进行测试，以验证被测软件模型的接口与需求的一致性。接口测试应符合以下要求：

a) 应包括被测软件模型的输入接口、输出接口；
b) 应测试所有接口的边界值，及边界值适当的组合，对于枚举型变量，应遍历所有枚举值；
c) 应结合 4.1.7 进行测试，并覆盖所有的接口故障模式。

4.1.6.2 评价方法：

经过 4.1.6.1 测试后，接口覆盖率应为 100%，测试用例通过率应为 100%。

4.1.7 故障注入测试

4.1.7.1 测试方法：

应在模型在环测试环境中，采用模型动态测试工具通过篡改输入接口值、内部中间变量等方式，模拟故障注入被测软件模型，以验证被测软件模型的故障处理及恢复机制、故障响应时间及恢复时间等是否符合设计要求。

注：常见的故障类型及注入方法示例见附录 B，企业可根据自身的开发需求进行调整及补充。

4.1.7.2 评价方法：

经过 4.1.7.1 测试后，测试用例通过率应为 100%。

4.1.8 背靠背测试

4.1.8.1 测试方法：

应在被测软件模型生成代码后，在软件在环测试环境中，采用背靠背测试工具在代码端输入与模型端相同的激励，对代码端与模型端的输出值进行对比，以验证被测软件模型的行为与生成的代码之间的一致性。

注：背靠背测试可结合模型端动态测试的测试用例及模型工具自动生成的测试用例进行测试，以提高测试效率。

4.1.8.2 评价方法：

经过 4.1.8.1 测试后，测试用例通过率应为 100%。

注：输出值对比时，针对布尔类型、枚举类型等状态信号的输出值，不应有偏差；针对定点型、浮点型、整型等连续信号可允许的偏差范围应在软件需求中明确定义，如无特殊定义，默认采用±1%。

4.1.9 资源使用评估

4.1.9.1 测试方法：

应在目标环境中，采用资源使用评估工具测试被测软件单元在最差情况下的 ROM、RAM、函数执行时间等资源的使用情况，以验证是否符合企业设计限值要求。

4.1.9.2 评价方法：

经过 4.1.9.1 测试后，测试用例通过率应为 100%。

4.1.10 结构覆盖率统计

测试方法：

应在基于需求的测试、接口测试、故障注入测试等测试方法的均测试完成后，采用模型覆盖率统计工具统计模型的判定覆盖率、MC/DC 覆盖率等结构覆盖率指标。

评价方法：

a) 对于 ASIL A、B、C 等级的软件，判定覆盖率均应达到 100%；

b) 对于 ASIL D 等级的软件，MC/DC 覆盖率应达到 100%。

注 1：未被覆盖的模型部分可通过模型审查的方式进行确认。

4.2 代码单元验证

4.2.1 静态代码分析

测试方法：

应在不运行软件代码的情况下，采用代码静态分析工具对被测软件单元进行编码规范符合性扫描，并统计编码规范违背项的数量。

注 1：静态代码分析工具应支持 GB/T 34590.6-2022 标准的形式化验证、半形式化验证、控制流分析、数据流分析、静态代码分析、基于抽象理由的静态分析等方法。

注 2：常见的编码规范 MISRA C、CERT、AUTOSAR C++等，企业可根据自身的编码需求对编码规范进行自定义。

评价方法：

编码规范的违背项数量应为 0。

注：经过代码审查可接受的编码规范违背项可不纳入统计范围。仅在经过代码审查专项小组评估并一致同意，且该违背项有充分技术理由、不影响功能安全、并在代码中详细记录的前提下，方可将其排除。所有排除项必须记录在案。

4.2.2 代码质量度量

测试方法：

应在不运行软件代码的情况下，采用代码度量工具对被测软件代码进行度量指标的静态扫描，度量指标应包括代码注释率、函数路径数量、函数圈复杂度、函数扇入、函数扇出、函数参数数量、函数嵌套深度。

注 1：企业可根据自身的开发需求对代码度量指标进行扩充及调整。

评价方法：

a) 30%≤代码注释率≤50%；

b) 函数路径数量≤80；

c) 函数圈复杂度≤10；

d) 函数扇入≤5；

e) 函数扇出≤7；

f) 函数参数数量≤5；

g) 函数嵌套深度≤4。

4.2.3 代码走查

测试方法：

应由代码测试人员、非被测软件代码的开发人员与被测软件代码的开发人员面对面沟通，依据软件单元需求人工逐条验证软件单元的逻辑和完整性。

评价方法：

走查通过率为100%。

4.2.4 代码审查

测试方法：

应由项目经理、开发人员、测试人员、质量保证人员等组建的代码审查专项小组，设计代码审查单，通过人工方式逐条检查被测代码是否符合代码审查单的要求。

注1：代码审查单示例见附录B，企业可根据自身的开发需求，对代码审查单进行调整及补充。

注2：若有违背项，应给出合理的理由并在代码中的适当位置记录。

评价方法：

审查通过率为100%。

4.2.5 基于需求的测试

测试方法：

应根据软件单元设计规范和分配给软件单元的软件安全要求，分别设计正常及异常的测试用例，采用代码动态单元测试工具，对软件代码进行测试，以验证软件代码与需求的符合性。

注1：应将需求分解至原子需求。

评价方法：

需求覆盖率为100%，测试通过率为100%。

4.2.6 接口测试

测试方法：

应根据软件单元的接口设计规范，以单个函数为测试对象，分别设计正常及异常的测试用例，采用接口测试工具，对软件接口进行测试，以验证软件接口与需求的一致性。

a) 应包括被测函数的输入、输出及内调接口。

b) 应测试所有入口参数、全局变量的边界值及边界值适当的组合。对于枚举型变量，应遍历所有枚举值。

c) 接口测试应结合故障注入测试（见4.2.7）进行，并覆盖所有的接口故障模式。

评价方法：

接口覆盖率为100%，测试通过率为100%。

4.2.7 故障注入测试

测试方法：

应采用软件故障注入工具或编写故障注入脚本，在软件单元层面注入不同类型的软件故障，以验证被测软件代码故障处理及恢复机制等是否符合设计要求。

注：常见的故障类型及注入方法示例见附录C，企业可根据自身的开发需求进行调整及补充。

评价方法：

测试通过率为100%。

4.2.8 资源使用评估

测试方法：

应在目标环境中，采用资源使用评估工具测试在最差情况下的函数执行时间、ROM、RAM等资源使用情况，并评估被测软件单元的资源使用情况是否符合设计要求。

评价方法：

测试通过率为 100%。

4.2.9 结构覆盖率统计

测试方法：

应在基于需求的测试、接口测试、故障注入测试等测试方法的均测试完成后，采用模型覆盖率统计工具统计模型的语句覆盖率、分支覆盖率、MC/DC 覆盖率等结构覆盖率指标。

评价方法：

a) 对于 ASIL A 等级的软件，语句覆盖率应达到 100%；
b) 对于 ASIL B、C 等级的软件，分支覆盖率均应达到 100%；
c) 对于 ASIL D 等级的软件， MC/DC 覆盖率应达到 100%。

5 软件集成验证

5.1 模型集成验证

5.1.1 模型静态分析

按 5.1.1 的测试及评价方法进行验证。

5.1.2 基于需求的测试

测试方法：

应根据软件架构需求规范对软件进行可测条目分解，采用模型动态测试工具，对集成后的软件模型进行动态测试，以验证集成后的软件模型是否符合软件架构需求规范的要求。

注 1：可根据被测模型的需求，采用开环或者闭环的方法对集成后的模型进行测试。

评价指标：

需求覆盖率应为 100%，测试用例的通过率应为 100%。

5.1.3 接口测试

测试方法：

应根据软件架构需求规范设计接口的测试用例，采用模型动态测试工具，对软件模型接口进行动态测试，以验证集成后的软件模型是否符合软件架构需求规范的要求。

注 1：接口测试应包括被测集成后的软件模型的输入接口、输出接口及集成模块间的接口。

注 2：经过软件鉴定的组件模型的内部接口可不纳入测试范围。

评价指标：

接口覆盖率应为 100%，测试用例通过率应为 100%。

5.1.4 故障注入测试

测试方法：

应采用模型动态测试工具，通过篡改输入接口值、内部中间变量等方式，模拟故障注入集成后的软件模型，以验证集成后的软件模型的故障处理及恢复机制、故障响应时间及恢复时间是否符合设计要求。

注 1：常见的故障类型及注入方法示例见附录 B，企业可根据自身的开发需求进行调整及补充。

评价指标：

测试用例通过率应为100%。

5.1.5 背靠背测试

测试方法：

应在集成后的模型生成代码后，采用软件在环（SIL）测试工具在代码端输入与模型端相同的激励，对代码端与模型端的输出值进行对比，以验证集成后的模型的行为与生成的代码之间的一致性。

注1：背靠背测试可结合模型端动态测试的测试用例及模型工具自动生成的测试用例进行测试。

注2：输出值对比时，针对布尔类型、枚举类型等状态信号的输出值，不允许有偏差；针对定点型、浮点型、整型等连续信号可允许的偏差范围应在软件需求中明确定义，如无特殊定义，默认采用±1%。

评价指标：

测试用例通过率应为100%。

注1：输出值对比时，针对布尔类型、枚举类型等状态信号的输出值，不允许有偏差；针对定点型、浮点型、整型等连续信号可允许的偏差范围应在软件需求中明确定义，如无特殊定义，默认采用±1%。

5.1.6 结构覆盖率统计

测试方法：

应在基于需求的测试、接口测试、故障注入测试等测试方法的均测试完成后，采用模型覆盖率统计工具统计模型的判定覆盖率指标。

评价指标：

a) 对于ASIL A、B等级的软件，判定覆盖率宜达到100%；

b) 对于ASIL C、D等级的软件，判定覆盖率均应达到100%。

5.2 代码集成验证

5.2.1 集成静态分析

测试方法：

应对集成后的代码进行编码规范符合性扫描，方法具体见4.2.1。

注1：针对模型生成的代码及经过软件鉴定的组件代码可不纳入静态分析扫描范围。

评价方法：

编码规范违背项应为0。

注：经过代码审查可接受的编码规范违背项可不纳入统计范围。仅在经过代码审查专项小组评估并一致同意，且该违背项有充分技术理由、不影响功能安全、并在代码中详细记录的前提下，方可将其排除。所有排除项必须记录在案。

5.2.2 基于需求的测试

测试方法：

应根据软件架构需求规范，对软件进行可测条目分解，设计满足需求的正常和异常测试用例，并采用软件集成测试工具对集成后的软件代码进行动态测试，以验证集成后的软件代码是否符合软件架构规范的要求。

注1：集成验证应根据软件集成策略分阶段进行，如BSW组件集成验证、ASW组件集成验证、ASW与BSW集成验证。

注2：集成验证测试环境有：软件在环、处理器在环及硬件在环，应根据需求属性选择合适的测试环境进行

测试。

评价方法：

需求覆盖率 100%，测试用例通过率 100%。

5.2.3 接口测试

测试方法：

应在软件在环测试环境中，根据软件架构需求规范划分不同接口层级，针对不同层级间、层级内的接口设计正常和异常的测试用例，并采用接口测试工具对集成后的软件接口进行测试，以验证软件接口与软件架构需求规范的一致性。

a） 集成接口测试应验证接口动态调用关系、接口间数据传递关系，应包含 ASW 与 ASW 间接口、BSW 与 BSW 间接口、ASW 与 BSW 间接口。

b） 针对实时任务、中断任务等接口，应结合任务优先级、阻塞、死锁、抢占等资源竞争场景进行验证，保证数据的一致性。

注 1：经过软件鉴定的组件，其组件内的内部接口可不纳入集成验证接口测试范围。

评价方法：

接口覆盖率 100%，测试用例通过率 100%。

5.2.4 故障注入测试

测试方法：

应采用故障注入测试工具，在软件组件层面注入不同类型的故障，以验证被测软件代码故障处理及恢复机制等是否符合设计要求

注 1：常见的故障类型及注入方法示例见附录 B，企业可根据自身的开发需求进行调整及补充。

注 2：故障注入测试环境有：软件在环、处理器在环及硬件在环，应根据故障类型选择合适的测试环境进行测试。

评价方法：

测试用例通过率 100%。

5.2.5 资源使用评估

测试方法：

在处理器在环或硬件在环测试环境中，采用资源使用评估工具测试 CPU 负载率、执行时间、ROM、RAM、通信带宽等资源使用情况，并评估集成后软件的资源使用情况是否符合设计要求。

评价方法：

测试用例通过率 100%。

5.2.6 结构覆盖率测试

测试方法：

应在基于需求的测试、接口测试、故障注入测试等测试方法的组合测试完成后，统计代码的函数覆盖率和函数调用覆盖率指标。

评价方法：

a） 对于 ASIL A、B 等级的软件，函数覆盖率和函数调用覆盖率宜达到 100%；

b） 对于 ASIL C、D 等级的软件，函数覆盖率和函数调用覆盖率均应达到 100%。

注 1：结构覆盖率数据通常由代码集成测试工具自动生成。

6 嵌入式软件测试

6.1 基于需求的测试

测试方法：

应根据软件安全需求对软件进行可测条目分解，设计满足需求的正常和异常测试用例，并在目标环境中，对嵌入式软件进行动态测试，以验证嵌入式软件是否符合软件安全需求。

注 1：嵌入式软件测试的目标环境包括：硬件在环环境、电子控制单元网络环境及整车环境。

评价指标：

需求覆盖率为 100%、测试用例的通过率为 100%。

6.2 故障注入测试

测试方法：

应根据不同的故障模式及故障注入位置，选择不同的故障注入工具进行测试，以验证被测嵌入式软件的故障处理及恢复机制、故障响应时间及恢复时间等是否软件安全需求。

注 1：常见的故障类型及注入方法示例见附录 C，企业可根据自身的开发需求进行调整及补充。

评价指标：

测试用例的通过率为 100%。

附 录 A
（资料性）
模型审查单示例

用于模型审查的模型审查单示例见表 A.1。

表 A.1 模型审查单示例

序号	模型审查内容
1	模型中的适当位置是否记录了模型版本信息？模型版本是否与待审查的模型版本一致？是否有变更记录？
2	模型设置中的求解器、数学和数据类型、诊断、硬件实现、模型引用、代码生成等配置参数的设置是否符合规范要求？
3	模型是否可编译？如果编译有错误导致编译不通过，应停止模型审查；如果编译有警告，宜给出合理的理由并在模型中的适当位置记录。
4	是否进行了模型静态分析？静态分析时建模规范的选择及抑制是否合理？ 静态分析的报告是否没有违背项？若有违背项，应给出合理的理由并在模型中的适当位置记录。
5	是否进行了模型质量度量？模型度量的指标的选择及度量值是否符合设计要求？如果有违背项，宜给出合理的理由并在模型中的适当位置记录。
6	分配给它的软件需求是否在软件模型中已正确地实现？如果没有实现，应给出合理的理由。
7	所有的软件模型是否都在分配给它的软件需求中有描述？如果需求中没有描述，应给出合理的理由并在模型中的适当位置记录。
8	模型的数据字典定义及接口的定义是否符合软件接口设计要求？接口的属性定义是否完整？接口命名是否符合命名规范？是否有未实现的接口或者冗余的接口？
9	模型中是否有合适的模型注释，模型注释是否与模型的需求和设计一致？

附 录 B
（资料性）
代码审查单示例

用于代码审查的代码审查单示例见表 B.1。

表 B.1 代码审查单示例

序号	代码审查内容
1	代码中的适当位置是否记录了软件版本信息？版本是否与待审查的软件版本一致？是否有变更记录？
2	代码是否可编译？如果编译有错误导致编译不通过，应停止代码审查；如果编译有警告，宜给出合理的理由并在代码中的适当位置记录。
3	是否进行了代码静态分析？静态分析时编码规范的选择及抑制是否合理？ 静态分析的报告是否没有违背项？
4	是否进行了代码质量度量？代码度量的指标的选择及度量值是否符合设计要求？
5	分配给它的软件需求是否在软件代码中已正确地实现？如果没有实现，应给出合理的理由。
6	所有的软件代码是否都在分配给它的软件需求中有描述？如果需求中没有描述，应给出合理的理由并在代码中的适当位置记录。
7	代码接口的定义是否符合软件接口设计要求？接口的属性定义是否完整？接口命名是否符合命名规范？是否有未实现的接口或者冗余的接口？
8	代码中是否有合适的注释，注释是否与代码一致？

附 录 C
（资料性）
常见的故障类型及故障注入示例

常见的故障类型及故障注入示例见表 C.1。

表 C.1 常见的故障类型及故障注入示例

验证阶段 / 故障类型		单元验证阶段	集成验证阶段	嵌入式软件验证阶段
内存故障	堆栈溢出	—	设计无限递归函数或分配超大局部数组	—
	非法访问	—	访问未授权的内存地址	—
	缓冲区溢出	—	向固定大小的数组或缓冲区写入超量数据	—
	内存泄漏	—	手动申请内存后不释放	—
	双重释放	—	重复释放同一内存块	—
	野指针	—	访问已释放的内存空间	—
	内存过载	—	申请分配超过内存空间的内存	—
	内存损坏	—	修改目标内存地址的变量值为错误值	—
	位翻转	—	修改目标内存地址的二进制位	—
接口故障	接口数据格式错误	修改接口传递的数据类型	修改接口传递的数据类型	—
	接口数据长度错误	修改接口传递的数据长度	修改接口传递的数据长度	—
	接口数据值错误	修改接口传递的数据值	修改接口传递的数据值	—
	接口数据缺失	删除接口传递的部分数据	删除接口传递的部分数据	—
	数据不一致	—	修改多个模块或组件内部的数据状态为不同值	—
调度故障	任务异常挂起	—	调用挂起函数将被测任务强制挂起	—
	任务异常删除	—	调用删除函数将被测任务强制删除	—
	死锁	—	使两个任务互相持有对方需要的资源	—
	活锁	—	使两个任务在无阻塞的情况下陷入无效循环或持续重试	—
	任务超时	—	在任务中调用 Delay 函数，延长任务的运行时间	—

	任务异常终止	—	将运行中任务提前终止	—
	频繁中断	—	将中断信号发生器连接到被测中断引脚，频繁产生中断信号，使两个中断的间隔时间小于中断服务程序的处理时间	—
	重复中断	—	在响应中断时调用多次中断服务程序	—
	中断丢失	—	在中断服务程序的入口处有选择性或随机地直接返回，使中断的响应次数比中断的触发次数少	—
	中断吊死	—	禁用中断使能寄存器，使不能响应新的中断请求	—
	中断超时	—	在中断服务程序中调用 Delay 函数，延长中断的运行时间	—
	CPU 过载	—	—	持续运行最高优先级的循环任务，使 CPU 过载
通信故障	信息丢失	—	断开通信总线或停止发送信息	断开通信总线或停止发送信息
	信息重复	—	模拟重复发送信息，使接收端接收到相同的信息超过一次	模拟重复发送信息，使接收端接收到相同的信息超过一次
	信息延迟	—	延时发送信息，使接收端在预期时间内没有收到信号	延时发送信息，使接收端在预期时间内没有收到信号
	信息插入	—	在发送端发送的信息流中插入额外的信息	在发送端发送的信息流中插入额外的信息
	通信地址错误	—	篡改发送信息的通信地址	篡改发送信息的通信地址
	伪造信息	—	发送端发送伪造的信息	发送端发送伪造的信息
	多点收发信息不对称	—	篡改发送的数据，使不同接收端从同一发送端接收到不同的信息	篡改发送的数据，使不同接收端从同一发送端接收到不同的信息
	信息仅部分接收方接收	—	篡改发送的数据，使在多个接收方中，只有部分接收方接收到信息	篡改发送的数据，使在多个接收方中，只有部分接收方接收到信息
	通信信息次序错乱	—	在一串信息流中，修改信息发送的顺序	在一串信息流中，修改信息发送的顺序
	信息损坏	—	篡改发送信息的数据	篡改发送信息的数据
	信道阻塞	—	发送大流量数据，使通信信道阻塞	发送大流量数据，使通信信道阻塞

硬件故障	硬线故障	—	—	将硬线对地短路、对电源短路，或者断开信号线
	电源故障	—	—	输入过高或者过低的电源电压
	寄存器故障	—	—	修改寄存器读写值，使寄存器值与期望值不符
	晶振失效	—	—	将晶振的引脚短路到地、短路到电源，或断开晶振
	时钟偏移	—	—	修改时钟频率或相位，使时钟信号偏移
	异常复位	—	—	修改复位寄存器，使硬件异常复位
注：“-”：不涉及				