Computer Science and Application 计算机科学与应用, 2021, 11(1), 217-232
Published Online January 2021 in Hans. http://www.hanspub.org/journal/csa
https://doi.org/10.12677/csa.2021.111022

Hans汉斯

高安全领域可编程逻辑器件软件工程技术研究与应用

王　栋[1]，刘雅清[2]，高　媛[1]，赵　静[1]

[1]北京京航计算通讯研究所，北京
[2]中国科学院高能物理研究所，北京
Email: 258434605@qq.com, liuyaqing@ihep.ac.c

收稿日期：2020年12月27日；录用日期：2021年1月22日；发布日期：2021年1月29日

摘　要

本文结合可编程逻辑器件软件的特点与优势，说明了当前航天工程等高安全领域中可编程逻辑器件软件的应用现状，并针对近年来研制过程中出现的与可编程逻辑器件软件相关的问题进行了原因分析，进而针对性提出了高安全领域可编程逻辑器件软件工程过程技术及其构成，并说明了该技术在相关领域中的应用情况，有效提升了可编程逻辑器件软件开发质量和效率。

关键词

可编程逻辑器件，软件工程

A Research and Application on the Programmable Logic Devices Software Engineering Technology in High-Safety-Requirements Field

Dong Wang[1], Yaqing Liu[2], Yuan Gao[1], Jing Zhao[1]

[1]Beijing Jinghang Research Institute of Computing and Communication, Beijing
[2]Institute of High Energy Physics, Chinese Academy of Sciences, Beijing
Email: 258434605@qq.com, liuyaqing@ihep.ac.c

Received: Dec. 27th, 2020; accepted: Jan. 22nd, 2021; published: Jan. 29th, 2021

文章引用: 王栋, 刘雅清, 高媛, 赵静. 高安全领域可编程逻辑器件软件工程技术研究与应用[J]. 计算机科学与应用, 2021, 11(1): 217-232. DOI: 10.12677/csa.2021.111022

Abstract

This paper explained the application status, combining features and advantages, of programmable logic devices (PLD) in high-safety-requirements field. Aiming at aerospace and other fields problems of research and production of PLD-related software defects, analysis of reasons was carried out, and then a programmable logic devices software engineering technology in high-safety-requirements field was given and applied in aerospace. The results prove that the technology can improve the quality and efficiency of PLD software development.

Keywords

Programmable Logic Device, Software Engineering

1. 引言

近年来，航天产品数量稳步上升，质量稳定可靠，航天发射任务连获成功，但随着可编程逻辑器件(包括 Field Programmable Gate Array FPGA，Complex Programmable Logic Devices CPLD，下文部分内容简称“FPGA”)在航天产品中的广泛应用，出现问题的情况也愈发凸显，对产品质量提出了更大挑战。

1.1. 可编程逻辑器件简介

可编程逻辑器件作为大规模集成电路技术和计算机辅助设计技术发展的结晶，出现于 20 世纪 70 年代，是一种半定制逻辑器件，它给数字系统的设计带来了革命性的变化。用户运用可编程逻辑器件，在生产商提供的通用器件上自行进行现场编程和制造或者通过对与或矩阵进行掩膜编程得到所需的专用集成电路，以满足不同设计需求。

1.2. 可编程逻辑器件具有的优势

可编程逻辑器件具有可编程、高集成度、高速和高可靠性等特点，通过配置器件内部的逻辑功能和输入/输出端口，可以将原来电路板级的设计放在芯片中进行，与传统的电路板级设计相比具有以下优势：

1) 降低产品的综合成本，用可编程逻辑器件设计和改进电子产品可大幅度减少印刷电路板的面积和接插件，降低装配和调试费用；

2) 提高了系统可靠性，大量中、小规模集成器件在向印刷电路板上装配时，往往会由于虚焊或接触不良而造成故障，并且这种故障常常难以发现，给调试和维修带来了极大的困难。可编程逻辑器件具有很高的集成度，很多功能可以在一片芯片上实现，有效地避免了上述缺点，并且内部电路不易受外界干扰，提高了系统的可靠性；

3) 降低了系统功耗，可编程逻辑器件内部电路尺寸很小，互连线短，分布电容小，驱动电路所需的功耗就大大降低；

4) 提高了电子产品的工作速度，芯片内部很短的连线能大大缩短延迟时间，有利提高速度；

5) 可编程逻辑器件由于具有较高的集成度，大大减小了电子产品的体积和重量；

6) 通过系统编程、重构等技术降低维护升级成本，可编程特点使设计更改更加灵活。

与通用数字信号处理(Digital Signal Processing, DSP)器件相比，可编程逻辑器件利用并行架构实现功能，在不少应用场合性能可超过通用 DSP 处理器的串行执行架构，同时由于具有现场可编程和在线可编程特性，有利于产品的更改、升级。

2. 可编程逻辑器件应用现状

由于具有上述优势，可编程逻辑器件非常适用于小批量、多品种、体积重量低的设备的研制，也驱使可编程逻辑器件在航空、航天等高安全要求领域应用越来越广泛。如在嫦娥三号、嫦娥四号及嫦娥五号有效载荷分系统中，可编程逻辑器件软件配置项数量占软件配置项总数量比例已达到 80%以上。

但在较早的航天型号研制过程中，大多没有将可编程逻辑器件软件纳入技术状态管理，而是随设备一起交付使用，在地面分系统试验、系统联试期间，多次发生通讯故障、信号翻转、异常复位等问题，给产品研制造成了过程反复、周期迟滞和成本增加，带来了较大的不利影响。

3. 可编程逻辑器件软件典型问题及原因分析

本小节列举了可编程逻辑器件应用过程中出现的部分典型问题，并进行了原因分析。

1) 在轨飞行任务功能失效

据统计，自 1971 年至 1986 年期间，国外发射的 39 颗同步卫星因各种原因造成的故障共计 1589 次，其中与空间辐射有关的故障有 1129 次，占故障总数的 71%，由此可见国外卫星和航天器的故障主要来源于空间辐射[1]。

国内某星载可编程逻辑器件软件在轨工作期间由于单粒子翻转事件发生重启、功能失效事件。

原因分析：芯片抗辐照性能不能满足任务需求，芯片选型等可行性分析工作欠缺，未提前对芯片抗辐照、资源等对运行环境、使用要求符合情况开展可行性和风险分析。按照国际通用的分类方法，电子元器件等级一般可分为宇航级或 883B 级、军级、工业级、商业级[2]。高安全要求领域推荐采用抗辐射加固器件等高等级芯片。

2) 移动电机发生过流急停

某设备进行热真空环境试验时，发现移动电机异常进入过流急停模式，导致试验中止，研制进度拖延。

原因分析：安全性可靠性设计及测试缺失。系统设计说明中未明确可编程逻辑器件软件与其他软件配置项间的接口要求，也未进行系统级到配置项级的安全性可靠性分析，FPGA 软件未按系统级协议要求进行设计、实现，安全性可靠性设计措施未落实，也未进行相关测试，导致问题在系统试验阶段发生。

3) 极端环境下功能异常

某系统中，将红外图像匹配的识别结果存储在随机存取存储器(Random Access Memory, RAM)中，由 FPGA 软件读取匹配结果进行判读，极端环境下 FPGA 软件无法正确读取 RAM 中的匹配结果。

原因分析：极端条件下时序验证不充分，FPGA 软件未对极端工况条件下时序的符合性进行充分验证[3]，导致极端工况下时序问题未被发现。航天型号特别是深空探测型号工作条件恶劣、严酷，比如嫦娥三号，月球表面的温度，白天阳光直射的温度高达 127 摄氏度，晚上温度可降低到零下 183 摄氏度。这种恶劣的工作条件，如果设计要求中不能考虑，势必会大大影响设计的健壮性。

4) CAN 总线通讯异常

某试验中，终端开机后进行初始状态检查时，显示 CAN 主节点与各子节点通讯故障，试验中止。后

组织排查，耗费大量人力、物力，严重影响了型号的研制进度。

原因分析：安全性可靠性设计缺失，FPGA 软件未考虑“异步复位同步释放”，设计时钟与复位释放信号间无法满足要求的时序关系[4]，使控制通讯功能的状态机进入异常状态并无法恢复到正常状态，导致死锁，无法通讯。

5) 接收机调幅动作错误

某通讯设备根据接收的工作状态和频点信息向接收机输出调幅使能控制信号，接收机做出错误的调幅动作，影响正常收发信号。

原因分析：FPGA 软件设计者将组合逻辑的结果直接输出给外部接口，由于组合逻辑易受温度影响而导致时序变化[5]，变化过程中产生的冒险即毛刺直接输出接收机上，造成接收机误动作。

6) 抗辐照设计措施失效

某 FPGA 在设计过程中为了增加抗干扰能力，对关键控制信号进行了三模冗余设计，当接收到指令后，该指令会同时控制三路信号，三路信号在输出给电路时会进行三判二处理，只有两路以上信号有效时，电路才会工作，但最终该设计措施并未实现。

原因分析：由于综合工具约束条件设置不合理，导致三路信号在综合过程中被综合工具优化合并为一路信号，无法实现冗余设计，大大降低了设计的可靠性。而同时在研制过程中并未开展针对该措施的综合后仿真验证，导致措施并未实现的问题推迟到第三方评测阶段才被发现。

此外，以上问题也间接说明了部分单位未将 FPGA 软件质量控制工作纳入质量管理范畴，没有明确的开发、测试、安全性等技术和管理要求，也未开展充分的验证和评审工作。

4. 国外可编程逻辑器件开发过程要求

4.1. 欧空局空间产品质量保证

欧空局 ECSS-Q-ST-60-02C (空间产品质量保证-ASIC 与 FPGA 开发)标准是欧空局空间工程应用产品质量保证系列标准中的一部分，其中定义了用户在开发数字的、模拟的和模数混合的定制设计集成电路时的需求。用户开发过程包括从设定初始需求开始到验证并发布原型器件结束过程中的所有行为。该标准对 FPGA 开发过程的定义如图 1 所示。

1) 定义阶段：主要任务是建立需求描述、可行性及风险分析报告、开发计划。

2) 结构设计：主要目标是定义、验证芯片的结构，记录芯片基本实现单元的预期作用，它们的接口及相互影响；确认芯片实现的重要要素。

3) 详细设计：将结构设计转换成基本单元级的结构描述。

4) 版图设计：版图产生的布局和布线信息需满足设计规则、时序以及其他的约束；在版图设计中应该使用再综合或者物理综合的网表优化项。版图提供的可靠信息包括：负载和耦合电容以及最后设计的规则检查，以确保经过验证的网表文件能快速流片。

5) 原型实现：在此阶段中，涉及芯片制造和芯片封装，FPGA 测试和原型测试。

6) 设计确认和发布：设计确认包括确认功能，性能，接口和兼容性达到目标要求，后进行产品发布。

7) 验证：贯穿整个开发流程的过程，在每个开发过程完成后确认是否按要求完成。

4.2. 美国机载电子硬件设计保证指南

美国机载电子硬件的设计保证指南 RTCA/DO-254，主要关注确保飞行安全的硬件设备的可靠性，定义了硬件设备供应商提供的应用在航空系统中的硬件必须达到的认证目标。该标准对复杂电子硬件开发过程的定义如图 2 所示，主要包括：

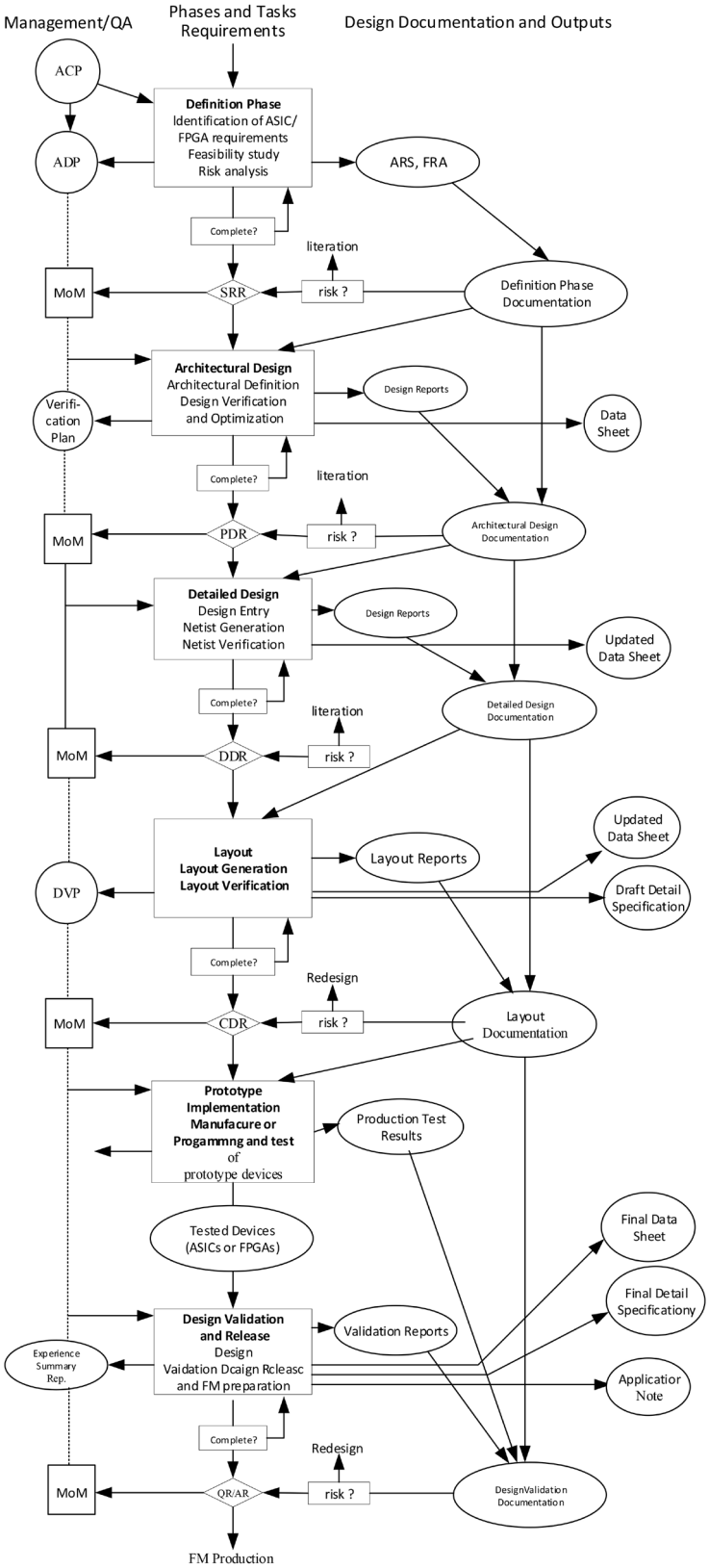

Figure 1. The ASIC/FPGA development process of ECSS-Q-ST-60-02C
图 1. ECSS-Q-ST-60-02C 的 ASIC/FPGA 开发过程

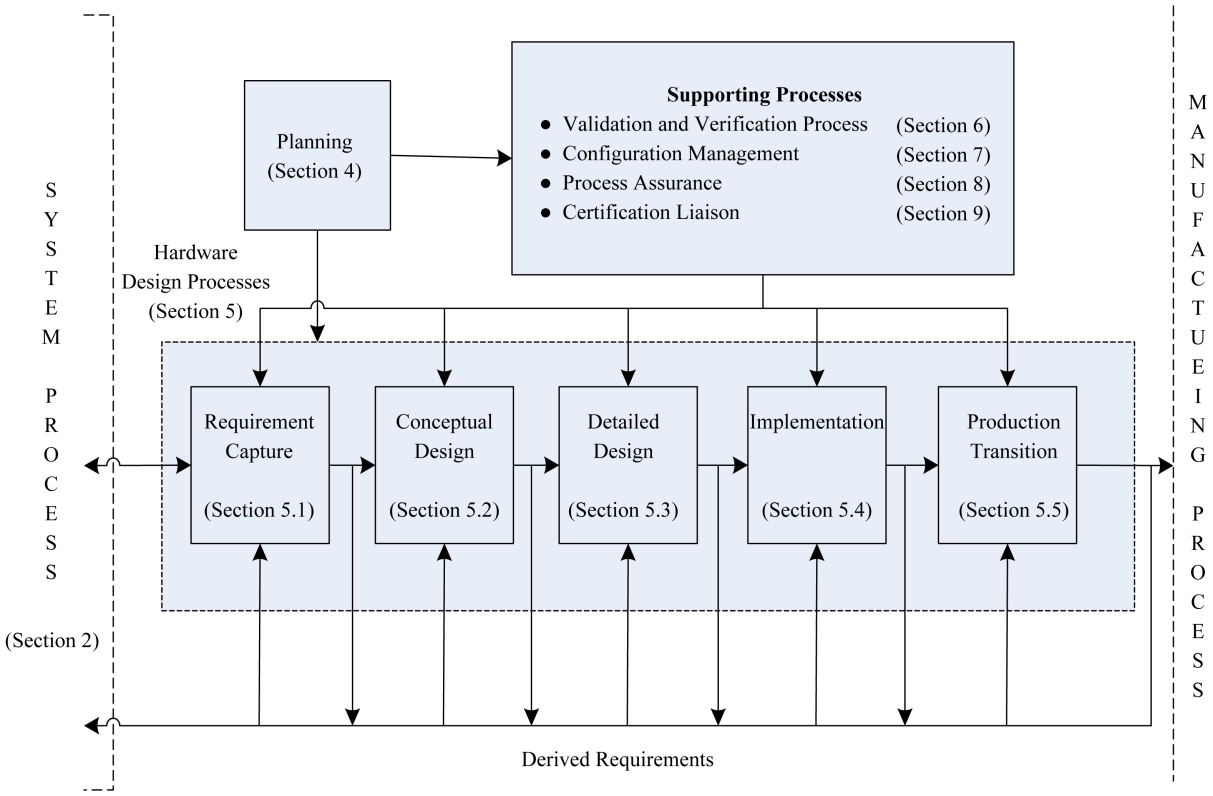

Figure 2. The development process of RTCA/DO-254
图 2. RTCA/DO-254 定义的开发过程

1) 需求获取：定义并记录硬件需求，包括硬件系统结构、技术选择、基本或可选功能，环境及性能需求，该过程可能存在反复，因为某些需求只能在设计过程中获得。

2) 结构设计：主要目标是定义、验证芯片的结构，记录芯片基本实现单元的预期作用，它们的接口及相互影响；确认芯片实现的重要要素；实现要满足需求定义阶段输出的需求。开展评审。

3) 详细设计：本阶段高级结构设计转换成基本单元级的结构描述。产生诸如版图约束、管脚布置、生产测试流程和详细的管脚描述等信息。

4) 实现：在此阶段中，涉及芯片制造和芯片封装，FPGA 测试和原型测试。

5) 产品发布：发布经过设计确认后的产品并进行流片。

6) 验证：贯穿整个开发流程的过程，在每个开发过程完成后确认是否按要求完成。

5. 可编程逻辑器件软件工程技术研究

结合国内外相关现状及国内研制过程中出现的典型问题，为确保不同类型、不同研制单位、不同安全关键等级的可编程逻辑器件软件都能在质量上有保证，规范可编程逻辑器件软件工程技术要求非常必要。图 3 为一种基于工程过程、方法、工具环境及安全性要素四维模型的高安全要求领域可编程逻辑器件软件工程技术体系构成，其中将安全性要素作为可编程逻辑器件软件工程技术的关键要素提出并贯穿整个工程过程。

5.1. 可编程逻辑器件软件工程过程

可编程逻辑器件软件工程过程包括如图 4，主要内容包括[6]：

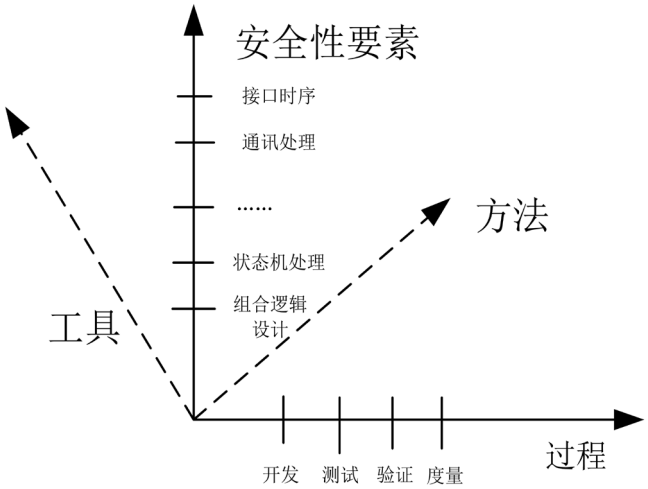

Figure 3. PLD software engineering system
图 3. 可编程逻辑器件软件工程体系

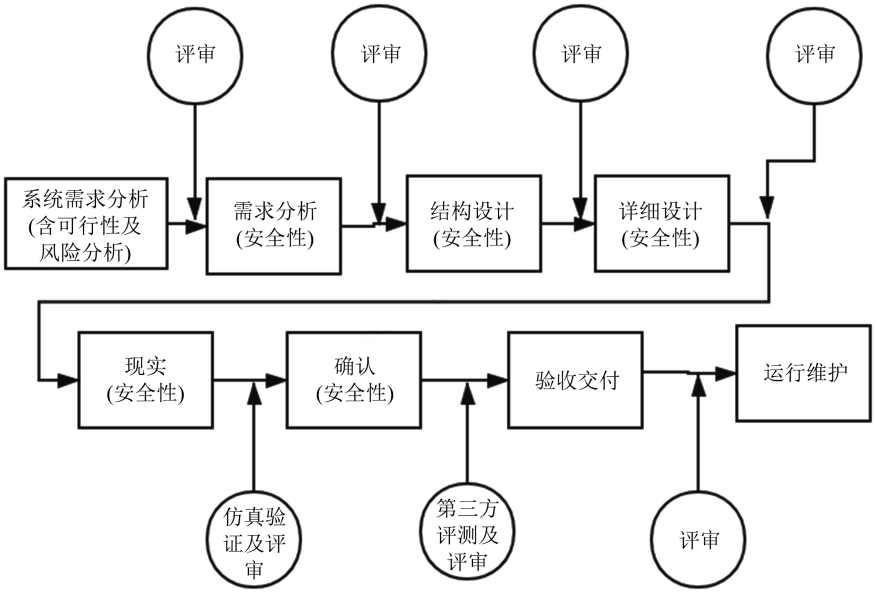

Figure 4. PLD software engineering process
图 4. 可编程逻辑器件软件工程过程

1) 系统需求分析(含可行性及风险分析)：针对器件可获得性、开发便利性、功耗、时钟频率、芯片资源、抗辐照方法、器件等级、封装方式、工作温度、存储温度、辐照总剂量等系统要求开展必要性与可行性分析；开展风险分析，确认风险项并制定对应的预防措施和应急响应，形成可行性及风险分析报告；明确关键等级，必要时为每个功能确定关键等级；明确功能、性能、接口、功耗、降额、工作条件等技术指标；明确外部接口的通讯协议、电气特性、时序特性等与可编程逻辑器件相关的软件/硬件接口；明确安全性和可靠性要求，如结构、算法、容错、冗余、抗状态翻转等；明确关键算法的算法和技术指标、语言和设计方法约束、资源和时序余量要求；明确固化、工作环境、实时性、可测试性要求；明确IP 核使用要求；明确安全保密性、重用、进度、验收与交付、运行与维护、配置管理、质量保证等要求；明确测试级别、测试内容、测试充分性等测试要求。

2) 需求分析：定义功能、性能、接口、管脚分配及约束等需求；开展安全性和可靠性分析，确定安全性和可靠性需求，明确抗状态翻转等技术方法；开展算法原型仿真分析；确定开发环境、工具及版本；描述芯片型号、等级约束和封装形式；确定语言和设计方法约束；确定资源、时序等余量需求；确定固化、工作环境需求；确定实时性、可测试性需求；确定 IP 核使用需求；确定安全保密性、重用、进度、配置管理、质量保证、测试等需求；明确与软件相关的地址分配；定义实现系统需求产生的派生需求，对其单独标识并反馈至系统需求分析过程，评估派生需求对系统需求分析的影响。

3) 结构设计：确定可编程逻辑器件软件结构及单元划分，定义单元及单元间的关系，描述单元的功能、性能、接口，确保全部需求分配至相应单元；描述单元间的数据流和控制流；描述结构及单元的设计决策；通过仿真和分析验证结构设计是否满足要求；确定时钟、复位方案；根据安全性和可靠性需求，开展安全性、可靠性设计；完成算法由抽象层次到实现层次的转换，确定算法的输入、输出接口，描述算法的实现过程；开展验证方案设计；结构设计过程中产生的派生需求应反馈至需求分析过程并评估影响；重新评估可行性和风险。

4) 详细设计：对各单元进行详细说明，包含各单元地址分配、控制方式、接口、存储器空间、时序说明、性能指标、测试要求等内容；述各单元的设计原理和所采用的技术方法及过程；详细说明各单元在实现时采用的设计输入方法；列出厂商、版本等 IP 核属性；完善安全性、可靠性设计，并分析详细设计是否符合安全性与可靠性设计要求；完善验证方案设计；标识出未被使用的功能并评估其对安全性的影响；约束软件的设计、固化和操作；详细设计过程中产生的派生需求应反馈至结构设计过程或其它过程并评估影响。

5) 实现

编码：依据设计开展编码活动，形成源代码或原理图。

仿真测试环境设计与实现：编制仿真测试计划和仿真测试说明，并依据仿真测试说明设计仿真测试环境，编制仿真测试向量集。

功能仿真：对源代码或原理图开展功能仿真，执行测试用例。应通过统计语句、分支、状态机等覆盖率信息保证仿真测试的充分性。

综合与布局布线：在完成编码与功能仿真后开展综合与布局布线，完善设计说明。

网表验证：开展时序仿真、静态时序分析，必要时，应开展门级仿真和逻辑等效性检查。

分析和记录：在仿真测试报告中记录仿真测试及分析的结果。

修改和回归测试：根据仿真测试的结果对软件进行修改，开展回归测试并根据需要更新文档和相关产品。

编程下载：将程序固化在可编程逻辑器件芯片或配置芯片中。

6) 确认：在真实的目标板、系统或在需方批准的替代环境上开展确认测试。

7) 验收与交付：开展验收测试、评审，编写用户手册、研制总结报告、版本说明等。

8) 运行与维护：实施维护，以消除缺陷或满足需求变更，做好技术状态控制、配置管理、回归测试和评审等工作。

5.2. 可编程逻辑器件软件工程方法

为达到与可编程逻辑器件软件工程过程有效结合，并解决第 3 节中所述的典型问题，结合国内工程实践，提出了可编程逻辑器件软件开发技术规范、可编程逻辑器件软件测试方法、可编程逻辑软件安全性设计与验证方法和规范。

5.2.1. 可编程逻辑器件软件开发规范

可编程逻辑器件软件开发技术规范包括开发过程技术要求(见 5.1 节描述)、编码要求、安全性设计规范和文档编制规范。围绕相关要素，基于国际先进标准和工程实践相结合的方法，采取由点到面，层层推进的方式，建立了可编程逻辑器件软件开发技术体系(见图 5)。

开发过程技术要求涵盖了 8 个开发活动的技术要求及相应成果文档编制规范。编码要求包括例化类、结构设计类、敏感列表类、声明定义类、命名类、运算类、循环控制类、分支控制类、时钟类、复位及初始化类、状态机类、综合/约束类、注释类、编码格式类共计 77 条规范(见表 1)。

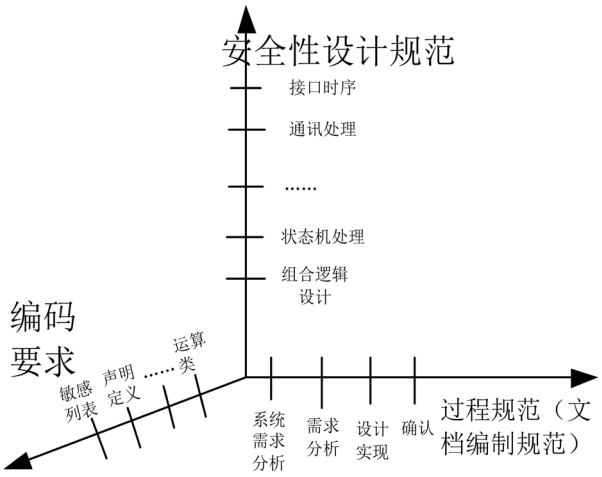

Figure 5. PLD software development technology system
图 5. 可编程逻辑器件软件开发技术体系

Table 1. PLD software coding guideline
表 1. 可编程逻辑器件软件编码要求

类型	详细要求
例化类	端口例化应采取名称映射的方式，避免位置映射 端口例化时不应进行逻辑运算操作
.....	.....
敏感列表类	应保证进程中的敏感列表完整、正确避免敏感信号多余或缺失
运算类	同一模块中的单向信号不应有多重驱动
综合/约束类	在用于综合的代码中避免使用不可综合的代码
.....	.....

制定了研制任务书、需求规格说明、设计说明、研制总结报告、测试计划、测试说明、测试报告等文档编制规范[6]。

5.2.2. 可编程逻辑器件软件测试规范

可编程逻辑器件软件测试规范体系包括测试过程、测试类型、测试方法(工具)三维要素(见图 6)。测试过程包括测试需求及策划、测试设计与实现、测试执行和测试总结。测试类型包括文档审查、代码审

查、功能测试、时序测试、安全性测试、恢复性测试、性能测试、接口测试、余量测试、强度测试、边界测试等。测试方法包括代码规则检查、代码审查、功能仿真测试、时序仿真测试、门级仿真测试、逻辑等效性检查、静态时序分析[7]。测试方法与测试工具对应关系见图 7 所示。

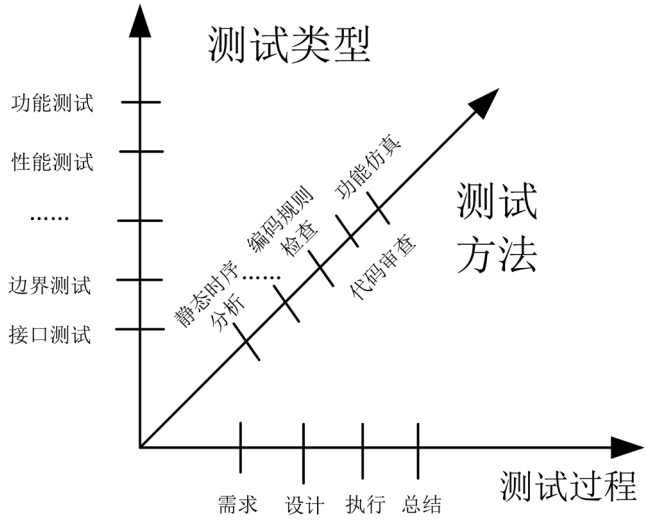

Figure 6. PLD software testing technology system
图 6. 可编程逻辑器件软件测试技术体系

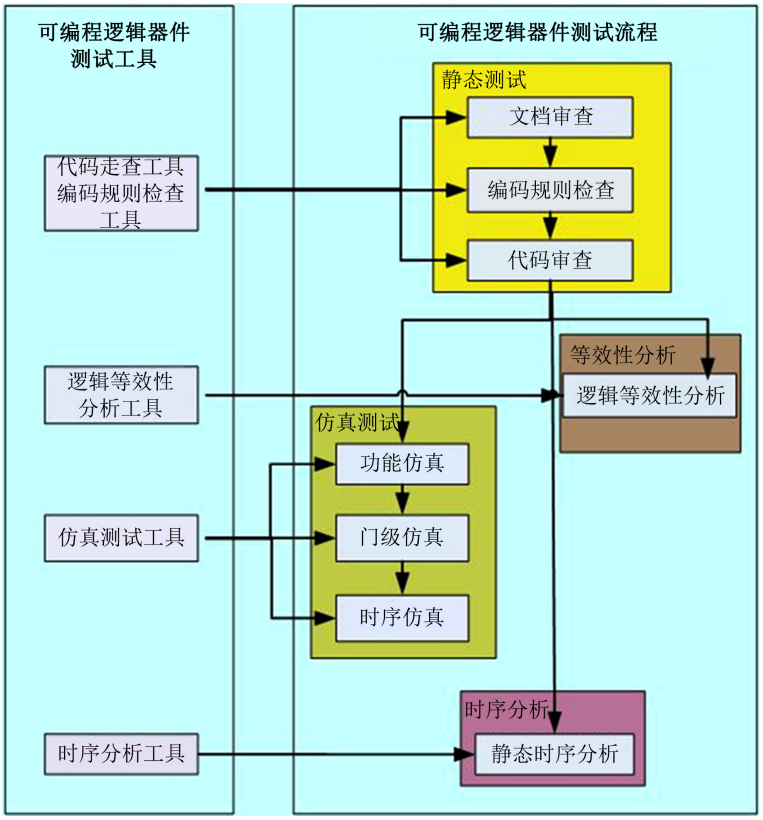

Figure 7. PLD software testing methods and tools
图 7. 可编程逻辑器件软件测试方法与工具

此外，提出了基于设计检查和门级仿真的状态机安全性及三模冗余有效性测试方法、基于覆盖率统计的可编程逻辑器件仿真测试方法、基于可重用仿真接口库的动态仿真验证方法，有效提高了 FPGA 软件的测试效率和质量。

5.2.3. 安全性设计与验证方法

可编程逻辑器件软件安全性工作贯穿于整个可编程逻辑器件软件开发全生命周期[8]。

1) 系统要求阶段分析系统的结构、功能、性能需求、工作环境、实际的外部接口时序(考虑电路板信号延时等因素)等对可编程逻辑器件软件的设计需求，包括可编程逻辑器件软件实现的功能、接口、功耗要求、性能、降额要求、工作条件要求等。

2) 需求分析阶段分析系统特定的可编程逻辑器件软件安全性需求的开发与分析应依据系统安全性需求、环境需求、标准、项目专用规范、工具或者设施需求和接口需求，查找安全关键功能，确定安全性需求。

3) 设计阶段为了保证可编程逻辑器件产品的安全性，仅在软件上实现安全性设计或仅在硬件上实现安全性设计是不完全的，还应以软件和硬件相结合的方式开展安全性设计。

4) 实现与集成过程需要落实安全性设计要求，如跨时钟域信号同步处理、异步复位同步释放、寄存器初始化、无效状态处理有效性、设置状态机初始状态、无不可达分支或冗余代码、敏感列表完整正确、条件判断表达式中使用逻辑运算符、注释率、无不可综合对象、典型工况时序满足、最大工况时序满足、最小工况时序满足、逻辑等价性、编译芯片型号正确性、无建立/保持时间冲突、总线各个位线避免集中布线、资源使用率、三模后资源使用率、时序余量。

5) 测试与验证过程是提高可编程逻辑器件软件安全性的重要阶段，在此阶段需进行静态测试和动态测试，测试工作的质量会影响可编程逻辑器件投入使用后能否安全工作，因此“安全性测试充分性”和“安全性测试覆盖性”非常重要，测试是否充分、全面，关系到可编程逻辑器件软件安全性是否得到保证。测试与验证安全性要求如下：测试用例执行率、测试用例通过率、故障密度、故障排除、故障概率、安全性需求覆盖率、语句覆盖率、分支覆盖率。

6) 可编程逻辑器件交付与验收后，需要对软件进行更改时，应进行软件更改危险分析，并对更改后的软件进行回归测试：分析系统、分系统、接口、逻辑、规程和软件的设计更改以及程序更改以及程序更改对安全性的影响，以确保更改不会产生新的危险、不会影响已解决的危险、不会使现存的危险变的更加严重和不会对任何有关的(或接口的)设计或程序有不利的影响；对更改进行测试，以确保新的软件中不包含危险；确保将更改适当和正确地纳入了程序之中；评审和修改有关文档，以反映这些更改。

为了保证以上要求落实，提出了一种基于软件故障模式及影响分析(FMEA)、故障树分析(FTA)技术、软件安全性双向分析(BDA)技术、硬件－软件接口分析(HSIA)技术的可编程逻辑器软件故障模式分析方法[8] (如图 8 所示)。

1) 可编程逻辑器件软件故障模式及影响分析 FMEA 技术

可编程逻辑器件故障模式及影响分析 FMEA 用于分析产品所有可能的故障模式及其可能产生的影响，并将每种故障模式对可编程逻辑器件影响的严重程度进行分类，从而发现设计中潜在的薄弱环节，提出可能采取的预防性措施，以消除或减少故障发生的可能性，保证可编程逻辑器件产品的安全性。这种技术从最低级的组件发出来揭露系统故障，是由“由下而上”的归纳分析方法，追踪来自最低层的问题。

2) 可编程逻辑器件软件故障树分析 FTA 技术

可编程逻辑器件软件故障树分析(FTA)用于检查和识别可引起危险的可编程逻辑器件软件故障及其原因，它采用自上而下的、演绎的方式找出可能导致顶端事件的状态或关键路径的最小组合。

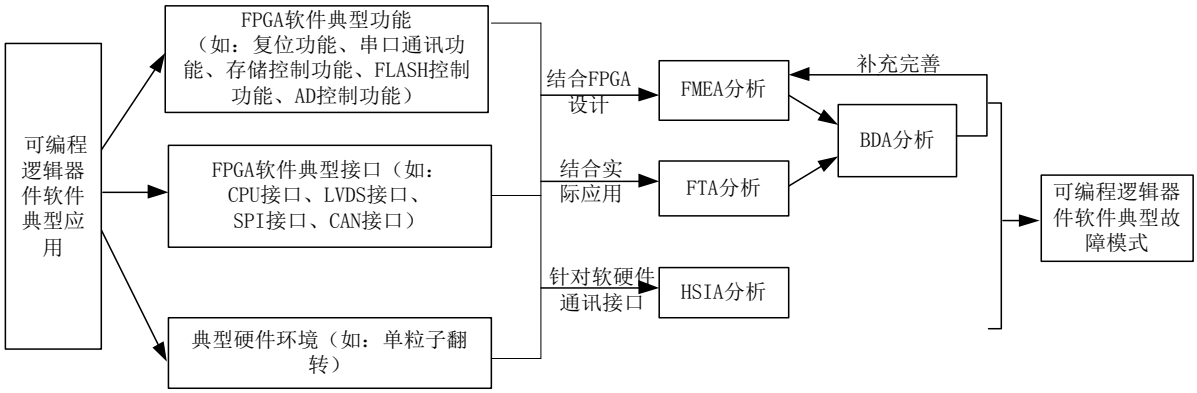

Figure 8. A PLD software typical fault model analysis method
图 8. 一种可编程逻辑器件软件典型故障模式分析方法

FTA 以不希望发生的作为可编程逻辑器件软件故障判据的一个事件(顶事件)作为分析目标，首先分析寻找所有能引起顶事件发生的直接原因，然后寻找引起上述每一个直接原因的所有直接原因。通过这样逐层向下推测所有可能的原因，直到无法再进一步分析的事件(或底事件)为止。从而形成故障树，故障树建成后，再定性分析各个底事件对顶事件发生影响的组合方式和传播途径，识别以顶事件为代表的各种可能的可编程逻辑器件故障模式，以及定量计算这些影响的轻重程度，计算出可编程逻辑器件软件故障概率和各个底事件的重要度次序。最后，根据分析结果，找出设计上的薄弱环节，并采取改进措施以提高产品的安全性。

3) 可编程逻辑器件软件安全性双向分析(BDA)技术

在可编程逻辑器件软件故障树分析(FTA)和可编程逻辑器件软件故障模式影响分析(FMEA)技术研究成果的基础上，借鉴基于软件故障树分析和软件故障模式影响分析相结合的双向分析技术方法(BDA)，将其应用在可编程逻辑器件软件双向分析中。

基于 FMEA 和 FTA 的双向分析法的基本思路是：首先采用 FMEA 对可编程逻辑器件软件设计进行分析，以确定异常输入或异常事件能否产生不安全的系统行为，即通过正向分析确定这些异常输入或异常事件可能导致不期望的结果(故障或失效)；然后采用 FTA 对这些不期望的结果(故障或失效)进行分析，以确定在系统设计下，这些不期望的结果不可能发生，即使发生了也能得到安全的控制，或者找出造成这些不期望结果的原因集合。

4) 硬件－软件接口分析(HSIA)技术

系统软硬件接口交互通讯的安全性对整个系统的安全性有着重要影响。在进行时应充分估计硬件－软件接口的各种可能故障，并采取相应的措施。

分析确定可编程逻辑器件软件典型的功能、典型应用接口、典型硬件环境，结合可编程逻辑器件设计，对可编程逻辑器件软件典型功能/典型接口/典型硬件环境进行故障模式及影响分析(FMEA)，形成 FMEA 表。

通过可编程逻辑器件软件故障模式及影响分析、故障树分析、安全性双向分析、硬件－软件接口分析技术开展安全性分析，形成系统设计、跨时钟域设计、复位设计、状态机设计等方面的安全性设计准则，如表 2 所示，用于指导安全性设计过程。

5.3. 可编程逻辑器件软件工程环境

可编程逻辑器件软件工程环境包括：软件开发环境、软件测试环境和管理环境[9]。在整个软件寿命

周期中，须使用受控的可编程逻辑器件软件正版开发工具，建立必要的测试环境和条件，并根据需要配置相关的软件管理工具，确保软件的开发、测试和管理有可靠的、适用的工具环境支持，可编程逻辑器件软件工程环境构成见图 9。

Table 2. PLD software safety designing guidelines
表 2. 可编程逻辑器件软件安全性设计准则

类型	详细要求
系统设计	可编程逻辑器件软件设计中，宜遵循面积和速度的平衡与互换原则
	组合逻辑模块的输出信号不应直接或通过层次关系间接连接到同一模块的输入端口
跨时钟域设计	宜采用同步设计，避免使用异步设计
	应避免组合逻辑信号跨时钟域传递
复位设计	异步复位建议采用同步释放的方式
	同一复位信号应使用唯一的有效电平，保证系统模块同时复位
状态机设计	对状态机的无效状态应进行适当处理
	避免出现状态机死锁
……	……

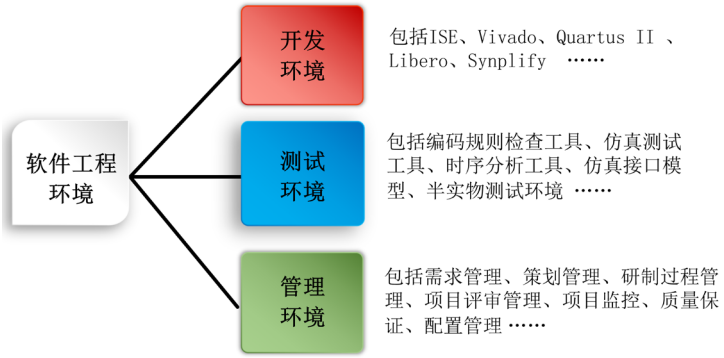

Figure 9. PLD software engineering environment
图 9. 可编程逻辑器件软件工程环境构成

5.3.1. 可编程逻辑器件软件开发环境

全球提供可编程逻辑器件开发工具的厂家很多，大体分为两类：一类是专业软件公司研制的 FPGA 开发工具，独立于器件厂商；另一类是器件厂商为了开发本公司产品研制的 FPGA 开发工具，只能用来设计开发本公司的产品。目前，FPGA 市场高度集中，主要厂商包括 Xilinx, Intel, Microchip Technology Inc. 和 Lattice Semiconductor. Xilinx 的开发工具包括 ISE 和 Vivado, Intel 的 FPGA 软件开发工具是 Quartus II, Microsemi 的 FPGA 软件开发工具是 Libero，而 Lattice 也推出了 Lattice Diamond, LatticeMico 等若干设计和验证工具套件。

5.3.2. 可编程逻辑器件软件测试环境

可编程逻辑器件软件测试环境主要包括：软件测试过程控制及分析工具、编码规则检查工具、仿真测试工具、仿真接口模型、侦错分析工具、时序分析工具、跨时钟域测试工具、逻辑等效性检查工具、

安全测试工具、半实物测试环境、板级验证工具、仿真验证平台等，包括：

1) 软件测评管理平台：主要功能包含软件测试过程控制及管理，文件管理，标准规范管理，人员信息管理，周例会管理，评审管理、新功能扩展等。

2) 半实物测试环境：通过搭建完整的自动化测试设备软件半实物测试环境，配合通用的软件动态测试工具例如用例生成、故障模式分析、故障注入、目标码验证、覆盖率分析、性能分析等工具。自动化测试设备软件的动态测试环境能够较真实地接近系统的实际运行环境，确保自动化测试设备软件测试的全面性、正确性、可靠性。同时在此基础上开展功能测试、性能测试、边界测试、强度测试、余量测试等，也更加有利于充分验证软件的需求。

3) 可编程逻辑器件仿真接口库：针对可编程逻辑器件仿真测试环境的共同需求，建立仿真测试环境和具体的接口模型库，使用该环境开展每个可编程逻辑器件测试环境的搭建工作，使仿真测试实现一定程度的自动化测试，提高测试效率。

5.3.3. 可编程逻辑器件软件管理环境

为保证设计开发的质量，需要建立 FPGA 软件管理的工程环境，提供可编程逻辑器件软件研制全生命周期的过程管理，支撑可编程逻辑器件软件系统需求分析、软件需求分析、设计、编码、测试和验证的设计开发平台，并完成编码规则检查、研制文档自动生成、测试数据自动生成等重要辅助功能。

可编程逻辑器件软件工程环境的配置在确定其主要目标和基本需求的基础上，把技术和管理、环境配置和工程需求、工具和方法、开发平台和目标平台、投入和产出等相关因素有机结合起来，全面考虑，以需求为牵引，以实用高效为准则，配置和完善软件工程环境。提高工具利用率，确保高技术高投入带来高效率高产出。

6. 高安全领域可编程逻辑器件软件工程技术应用

2009 年起，可编程逻辑器件软件工程技术陆续成功应用于探月工程、空间科学卫星工程、航天装备、核工业等高安全要求领域。统计数据表明，该技术的应用降低可编程逻辑器件软件缺陷率 90%以上，有效避免了过程反复，提高了研制效率。

6.1. 空间科学卫星工程应用情况

2011 年 1 月，中国科学院空间科学战略性先导科技专项正式立项实施。空间科学先导专项的总体目标是：在最具优势和最具重大科学发现潜力的科学热点领域，通过实施自主和国际合作的科学卫星计划，实现科学上的重大创新突破，带动相关高技术的跨越式发展，发挥空间科学在国家发展中的重要战略作用。空间科学先导专项一期的暗物质粒子探测卫星(在轨命名为“悟空”)、实践十号返回式科学实验卫星(“实践十号”)、量子科学实验卫星(在轨命名为“墨子”)和硬 X 射线调制望远镜卫星(简称“HXMT 卫星”，在轨命名为“慧眼”)四发四捷，并已产出系列重大成果，使我国逐渐成为国际空间科学领域一支重要的新兴力量[10]。

空间科学卫星工程的创新性，决定了其软件产品具有创新性强、技术难度大、工作模式复杂、任务要求高、可编程逻辑器件类型多、处理能力强等特点[10]。

结合空间科学卫星软件工程特点，该工程制定了《空间科学卫星工程软件管理规定》、《空间科学卫星工程 ASIC 和 FPGA 开发指南》，规定了空间科学卫星工程 ASIC/FPGA 产品研制的原则和要求，加强了工程软件产品开发过程控制和管理，明确了专项总体单位、系统总体单位、软件研制单位、软件专家组的职责，针对系统可编程逻辑器件软件研制的各阶段提出详细要求。

此外，该工程还组织软件专家组结合 FPGA 安全性和可靠性设计要点制定了系统/分系统级和配置项级的软件安全性和可靠性专项检查单，检查内容覆盖了 FPGA 安全性和可靠性分析、设计和验证；初样研制阶段，在研制单位自查基础上进行专项检查，提出整改意见，推动了研制工作的深入进行；正样研制阶段，针对系统安全性和可靠性设计，在轨故障预案对可编程逻辑器件软件的安全性和可靠性分析、设计、测试和验证对任务需求的覆盖性等方面进行了专项检查。同时，对各型号研制和评测过程中发现的问题进行总结、分析，形成了具有普遍适用性和指导意义的 FPGA 安全性和可靠性设计准则及典型案例(包括系统设计类、接口设计类、时钟设计类、复位/初始化类、综合类等共计 68 项)。为了规避同类问题重复发生，专项总体多次组织 FPGA 安全性和可靠性设计准则培训以及典型问题和案例分析专题培训，有效加强了可编程逻辑器件软件的安全性和可靠性[10]。

6.2. 探月工程应用情况

2004 年，中国正式开展月球探测工程，并命名为“嫦娥工程”。嫦娥工程分为“无人月球探测”“载人登月”和“建立月球基地”三个阶段。2009 年，探月与航天工程中心制定了国内首份重大工程可编程逻辑器件软件技术要求《探月工程可编程逻辑器件项目开发实施细则》，结束了重大工程可编程逻辑器件软件研制无据可依的阶段，本文所论述的工程过程技术成功应用于探月工程可编程逻辑器件软件开发与测试过程中，其中嫦娥三号着陆器、巡视器有效载荷分系统 14 个可编程逻辑器件软件据此完成了开发、测试和第三方评测工作，为嫦娥三号飞行任务圆满成功提供了有力保障。

此外，探月工程制定了《探月工程软件第三方评测管理办法》，首次提出了国内重大工程可编程逻辑器件软件评测技术要求，并据此开展了工程领域内第三方评测机构的资质认定工作。2013 年 8 月，探月工程制定了《探月与航天工程软件/FPGA 典型案例分析与设计参考》，并面向工程研制人员开展宣贯培训，有效避免了同类问题重复发生，大大提高了工程研制效率。

6.3. 其他高安全领域应用情况

此外，在核工业、电力、航天装备等多领域，可编逻辑器件软件工程技术也得到了成功应用，特别是通过开展可编程逻辑器件软件(第三方)测试工作，有效降低了领域内可编程逻辑器件软件缺陷率。

7. 总结与展望

随着可编程逻辑器件管理规范性的提高，高安全领域可编程逻辑器件相关的问题正呈现逐渐减少趋势，可编程逻辑器件软件研制也逐渐进入了体系化、标准化阶段。各单位在可编程逻辑器件软件研制过程中，需要积极总结经验，不断完善可编程逻辑器件软件设计，不断强化可编程逻辑器件质量要求，开展可编程逻辑器件产品设计、软件开发以及过程管理的培训，建立可编程逻辑器件软件知识库和资产库，采用各种技术和管理手段，更有效地保证可编程逻辑器件软件质量。

参考文献

[1] 侯成杰. 国外航天软件故障原因分析[J]. 航天器工程, 2012, 21(1): 89-96.

[2] 姜秀杰, 孙辉先. 航天电子系统中电子元器件选用的途径分析[J]. 电子器, 2005, 28(1): 40-43.

[3] 中国航天科工集团第三研究院第三〇四研究所, 等. GJB 9433-2018 军用可编程逻辑器件软件测试要求[S]. 北京: 国家军用标准出版发行部, 2018.

[4] 中国航天标准化研究所. QJ 20356-2014 航天型号可编程逻辑器件软件编码要求[S]. 北京: 中国航天标准化与产品保证研究院, 2014.

[5] 中国信息技术标准化技术委员会. GB/T 37691-2019 可编程逻辑器件软件安全性设计指南[S]. 北京: 中国标准出

版社, 2019.

[6] 中国信息技术标准化技术委员会. GB/T 33781-2017 可编程逻辑器件软件开发通用要求[S]. 北京: 中国标准出版社, 2017.

[7] 中国航天科工集团第三研究院第三〇四研究所, 等. GJB 9432-2018 军用可编程逻辑器件软件测试要求[S]. 北京: 国家军用标准出版发行部, 2018.

[8] 窦立刚. 安全性分析在提高 FPGA 质量过程中的应用[J]. 电子科学技术, 2014, 1(2): 153-156.

[9] 王栋. 航天型号可编程逻辑器件软件工程[M]. 北京: 中国宇航出版社, 2020: 20-21.

[10] 李超. 我国空间科学卫星软件工程化管理实践与思考[J]. 项目管理技术, 2020, 18(1): 80-87.