http：//myfj. cnjournals. com myfj_sadri@163. com （021）20866796

DOI：10. 19416/j. cnki. 1674-9804. 2023. 04. 008

多余度飞控计算机子系统余度重启机制研究

刘 鑫* 韩建辉

（航空工业第一飞机设计研究院，西安 710089）

摘 要：传统故障余度处理方式将故障余度减缓切除，造成飞控计算机子系统的降级，如果剩余的余度再出现故障可能危及飞行安全。有些余度通道故障可以通过重启余度通道来恢复正常运行。通过分析余度通道正常启动流程，以及飞控计算机子系统余度控制应用软件重启流程，研究了双 CPU 命令-监控模式和双命令模式下的重启流程。最后针对仿真设备节点故障、飞控节点故障以及飞控多余度故障进行了实验验证，84 个故障验证项经验证测试后全部通过。

关键词：飞控计算机；余度管理；CPU 运行模式；重启

中图分类号：V249. 1 **文献标志码**：A **OSID**：

0 引言

高可靠性是飞机的重要研究方向，飞控系统很大程度上影响着飞机的可靠性。因此，要提高飞机可靠性，必须提高飞控系统的可靠性[1-3]。飞控系统的核心是飞控计算机子系统[4]。为了提高飞控系统的可靠性，工程实践中广泛使用多余度飞控计算机子系统架构，二余度结构[5]、三余度结构[6]、四余度架构[7]是工程上最常用的几种架构。在四余度飞控计算机子系统中，余度通道故障的应对策略一般为减缓切除方法[8]，即连续指定周期或累计指定周期被判定异常就会将该余度通道切除。这样的策略下，如果继续有通道发生故障[9]，飞控功能将快速降级，最终可能影响飞机任务的完成度。

在工程实践中，发现有些飞控计算机余度故障是单粒子效应等原因引起的，系统器件在经过复位或者重新上电后，可以恢复功能。如果因为这些故障原因就将整个余度通道隔离切除，系统的效能会受到较大的影响。因此需要对这些故障进行具体分析，并提出系统功能及性能容忍的、非余度切除的恢复性处理措施，即余度通道空中重启[10]。在余度通道重启的过程中，引发故障的硬件故障原因消除，系统恢复正常，系统得以继续运行。或者，当余度通道进行重启后，系统原有的某些故障状态可能会恢复。这类故障是因为系统中未被研制过程中的验证环节所发现的系统缺陷引起的。缺陷在系统特定的运用状态下被触发，引发余度故障甚至系统失效。在系统重启后，因为系统状态的改变，又避开了触发条件，使得系统可以继续运行，系统部件可继续发挥作用，因此重启故障余度不失为一种可利用的系统容错策略。

1 余度通道正常启动流程

余度通道系统正常启动的流程如下：

1）通道控制计算机上电重启。

2）系统软件根据空中/地面状态判定进行自检、初始化。

3）系统软件根据空中/地面状态判定确定是否进行控制程序映像校验。

4）系统软件加载控制程序到内存。

5）余度同步，即同步采用握手[11]的算法进行。在余度通道内部，还会采用指令通道与监控通道配合以对本通道进行自检的技术手段。指令通道与监控通道也需要进行同步。余度通道内部的这类

* **通信作者**. E-mail：361626718@qq. com

引用格式：刘鑫，韩建辉. 多余度飞控计算机子系统余度重启机制研究[J]. 民用飞机设计与研究，2023(4)：50-56. LIU X，HAN J H. Research of redundant channel restart mechanism of redundant flight control computer subsystem[J]. Civil Aircraft Design and Research，2023(4)：50-56(in Chinese).

同步需要在余度间同步前进行。

6）启动控制功能及故障检测:同步结束后,通道系统软件启动控制功能程序,控制功能程序开始进行控制逻辑运行。系统软件在进行控制功能程序周期调度时,在每个控制周期的开始,为消除各个通道同步误差,还需要进行控制周期同步。同步仍采用高低电平握手方式。在同步成功后,通道系统软件再启动下一个周期的控制功能程序调度。

2　飞控计算机子系统余度重启流程分析

上一节对余度通道正常重启流程进行简单描述,该技术在工程上已经比较成熟。在实际系统中,如判定一个飞控计算机子系统余度通道出现故障,一般对其进行故障减缓切除处理,但切除故障余度后,如果剩余的正常运行的余度发生故障,可能导致系统快速降级,影响任务的完成。美国 SpaceX 公司在“猎鹰 9 号”(Falcon 9)火箭上采用了商用的 X86 处理器,控制系统使用了三个处理器,组成了 3×2 的 6 余度系统,所有的 6 个余度之间的结果数据进行互传校验,计算出错的处理器进行复位,复位后拷贝相关状态数据后重新上线运行以提高系统的可靠性。由于飞控系统的高安全、高可靠和高实时性的要求,以及在计算机和通信总线技术上与商业领域的差别,本文借鉴了国外商用分布式系统的余度重启上线思路,对切除的故障余度进行重启,即自身或其余通道(或作动器远程终端子系统)输出强制故障余度重启的指令。

故障通道如果得到了强制其重启的指令,仍然可以在两个层面上进行,一个是系统级上电重启,另一个是控制应用软件重启,可恢复不同范围的故障。如采用整个系统上电重启的措施,则故障余度通道上电重启并完成操作系统和飞控应用的加载。如果是单独的软件重启,则飞控应用退出并重新加载运行。由于第四章的实验针对控制器软件重启,本章主要描述故障余度控制应用软件重启流程。

余度通道控制应用软件重启流程如下:

1）系统软件根据空中/地面状态判定确定是否进行控制程序映像校验。

2）系统软件加载控制程序到内存。

3）余度同步:同步采用握手的算法进行。同正常启动流程。

通道间的握手同步信号传输需要具有可接受的时效,不能耗时过长,以免失去及时握手的意义。因此,可以采用通道间直接的电平信号连接。

而系统中其他通道根据余度表决的时序要求,也正进行控制周期同步。控制周期同步算法与启动同步一致。只是同步等待的时间不同。启动同步的握手超时时间可以设计的略长,可以大于控制周期,以确保不因时间太短而不能建立与其他通道的同步。

重启通道正常情况下可以与系统中其他通道完成同步。同步成功结束后,各个通道标记重启通道为重启同步成功状态。

4）系统软件进行余度间数据迁移:同步结束后,需要进行余度间数据迁移工作,同步余度控制程序间的状态数据。因为一般的控制应用软件都是所谓“有状态”程序,程序输出依赖于当前的程序状态。如果不进行状态数据同步,导致通道间输出超出差异阈值,重启通道仍然会被认为故障。

重启通道上的系统软件在同步成功后,根据控制功能程序的周期执行耗时,延迟一段时间即开始启动后续数据迁移工作和控制工作。其他通道只要在控制周期同步时,发现有新通道为重启同步成功状态,即在同步成功后的同一个控制周期内,启动数据迁移工作。如果和重启通道同步上的其他通道有多个,简单起见,重启通道按系统设定的优先级先后持续接收通道发送的迁移数据。和重启通道同步上的其他通道仍然需要发送迁移数据给重启通道,以避免重启通道优先认定的数据来源通道不能发送数据的情况。

余度间数据迁移在不透明方式(不透明方式需要控制功能程序的数据段及 BSS 段的内存部署位置是系统软件已知的)处理视角下,程序的状态数据从抽象的程序二进制实现上看,保存在程序进程空间的数据段、BSS 段以及栈段上。重启通道的控制程序可不迁移栈段数据。因为此时栈上保留的状态信息,对控制逻辑而言并非必要。重启通道控制程序可以重新建立自身的栈内容而不影响控制。

系统中其他通道在和重启通道同步上的控制周期中,待正常的控制程序周期运行结束后,由系统软件将控制功能程序数据段及 BSS 段内存数据内容通过 CCDL 一次性发送给重启通道。

对于有指令支路及监控支路的系统,CCDL 一

般同时将数据传输给两个支路。因此,迁移数据也同时传输给两个支路,两个支路的数据迁移工作可同时进行。

3　双 CPU 运行模式对余度通道重启的影响

当前工程所用飞控计算机一般为双 CPU 架构,具有双 CPU 的余度通道,又根据两个 CPU 能否访问外部外设而分为两种模式:命令-监控模式、双命令模式。下面就双 CPU 运行模式对余度通道重启的影响进行分析。

3.1　命令-监控模式

如本双 CPU 余度通道故障,整个飞控计算机余度通道会被判故障,系统会减缓切除该余度通道。如不能恢复,则飞控计算机子系统其他通道表决出重启指令使其重启。该重启指令也经过两 CPU 间数据通道传输到监控通道(如图 1 所示)。其他通道发出的重启信号经两个 CPU 板上的硬件重启逻辑综合后,可以发出使 CPU 外部中断的信号,CPU 系统软件进行相应的中断处理,如果需要进行系统级重启,则向其他余度通告本通道重启,通过系统重启硬件端口输出使能重启信号。

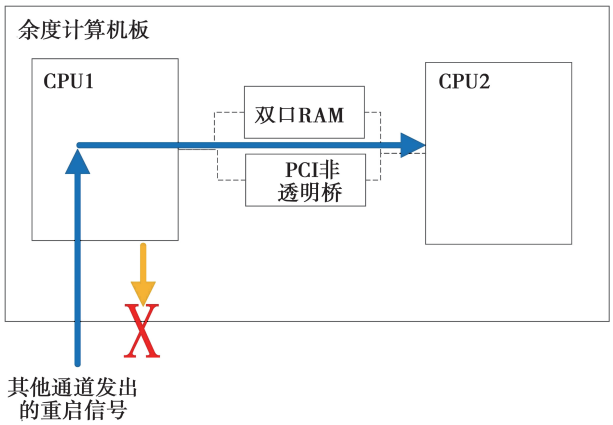

图 1　命令-监控重启

重启时,两支路同时重启。每个支路重启过程如下:

1) CPU 支路上电重启。

2) 系统软件根据空中/地面状态判定进行自检、初始化。

3) 系统软件根据空中/地面状态判定来确定是否进行控制程序映像校验。

4) 系统软件加载控制程序到内存。

5) 余度通道支路间同步:在余度通道作为一个整体与其他余度进行同步之前,两个 CPU 支路间要先进行支路间同步。支路间同步需要的硬件支持是两支路间的一对信号线连接。同样是采用高低电平握手的方式进行同步。

6)余度通道同步:余度通道内支路同步成功后,指令支路输出与其他余度通道进行高低握手的指令,指令支路也都获取其他支路的电平信号,进行通道同步判断。

7) 系统软件进行余度间数据迁移:余度通道的支路 CPU 与其他通道同步成功后,指令支路接收其他通道给本通道的迁移数据并转发给监控支路。两个支路的迁移过程同单 CPU 设计余度通道。

8) 启动控制功能及故障检测:数据迁移结束后,相应支路的系统软件启动控制功能程序。在下一个控制周期时刻到来时,系统软件调度功能程序运行。控制功能程序因为是重启后第一次被操作系统软件调度,程序应从初始化运行。初始化又可能会破坏迁移过来的数据,因此,在程序入口设置一个表征是否已经初始化的全局变量。在数据迁移后,该全局变量已经设置为真,程序判断为真,则不再进行相关初始化了,而直接进行控制功能。但是,某些通道号相关的变量若需要特殊处理,则需要进行初始化。

后续,控制功能程序开始进行控制逻辑运行。

同时,系统中其他余度通道,在数据迁移结束后的下一个控制周期,也将重启通道的状态置为有效,相应支路的状态也在两支路余度表决硬件逻辑中置为有效。

3.2　双命令模式

3.2.1　“主-主”模式

双命令模式下,当系统采用“主-主”方式运行处理故障时,整个飞控计算机余度通道会被判故,系统对该通道实行减缓切除策略。如该通道最终被切除,即故障不能恢复,则飞控计算机子系统其他通道表决出重启指令使其重启,如图 2 所示。

重启时,两支路同时重启。支路重启过程与命令-监控模式的重启过程相似,不同点在于:步骤 6)余度通道同步。

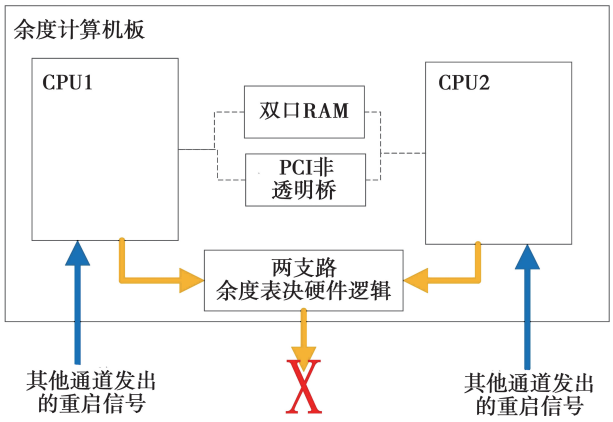

图 2　“主-主”模式重启

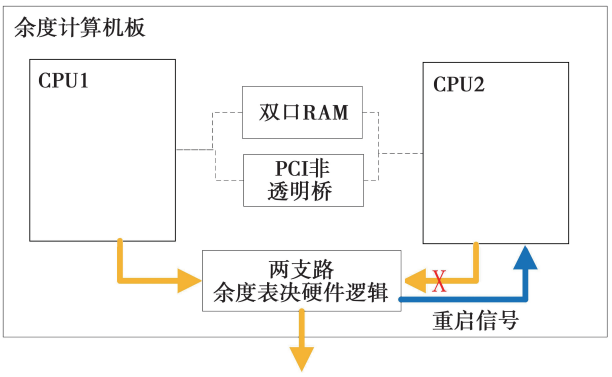

图 3　“主-备”模式重启

“主-主”模式下,余度通道内支路同步成功后,两个支路输出与其他余度通道进行高低握手的指令,两支路余度表决硬件逻辑在指令未分离的情况下优选一路,并通过离散信号端口输出高低电平给其他余度。两个支路也都获取其他支路的电平信号,进行通道同步判断。

3.2.2　“主-备”模式

当系统采用“主-备”方式运行处理故障时,整个余度通道仍然有一路 CPU 输出,该余度通道不会被其他通道判故。系统会对通道内选出的(或主动报告的)故障支路实施减缓切除余度算法,直至多次重试失败后,由两支路余度表决硬件逻辑向故障支路发出重启信号,如图 3 所示。

故障支路重启过程如下:

1)—4)　与命令-监控模式相同。

5)　支路间同步:故障 CPU 支路间要先与正常支路进行支路间同步。

支路间同步后,不再如同“主-主”故障处理方式,不再需要进行余度通道同步。

6)　系统软件进行支路间数据迁移:支路间同步成功后,正常支路获知重启支路同步成功状态,开始在控制功能程序周期运行结束后,由系统软件通过双口 RAM 或 PCI 总线给重启 CPU 支路发送迁移数据。重启 CPU 支路接收迁移数据。迁移过程同单 CPU 设计余度通道。

7)　启动控制功能及故障检测:数据迁移结束后,重启支路的系统软件启动控制功能程序。控制功能程序进行控制,系统也重新开始故障检测,包括两支路间的自监控。

4　实验与验证

验证环境由四余度飞控仿真计算机、仿真 PC 机和 PC 宿主机组成的硬件环境以及在各硬件上所运行的软件组成的软件环境共同构成,如图 4 所示。实验所用飞控计算机 CPU 为命令-监控模式。

主机端软件包括飞控机载软件开发环境、主机端余度故障注入软件(见图 5)、主机端余度运行与重启过程监控软件(见图 6)。

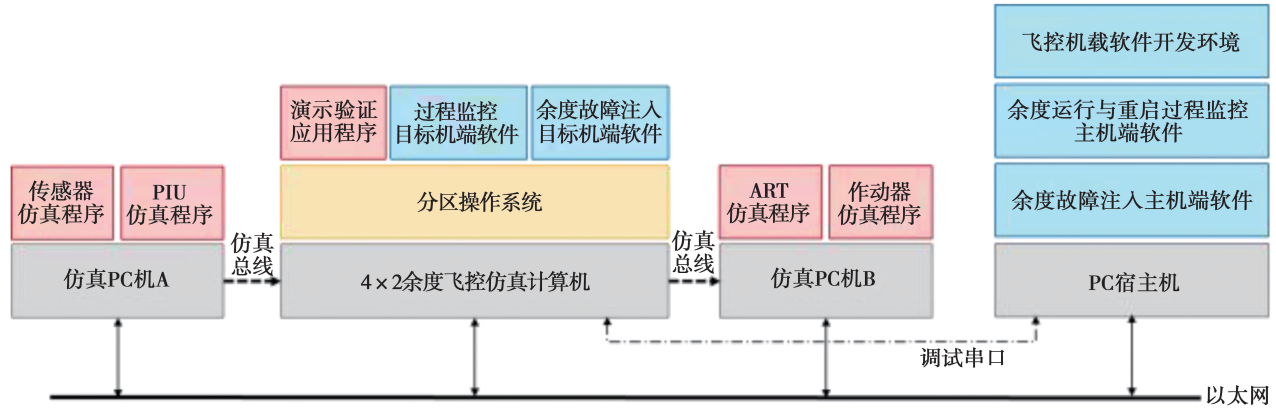

图 4　演示验证环境构成图

图 5　主机端余度故障注入软件

图 6　主机端余度运行与重启过程监控软件

针对仿真设备节点故障、飞控节点故障以及飞控多余度故障进行验证，具体情况如表 1 所示。

表 1　仿真设备节点验证测试项

序号	验证项	验证项说明	测试用例个数
1	飞控节点单余度故障验证(4)	每个飞控余度的 20 个故障	80
2	硬件永久故障不可恢复验证	4 个飞控余度	4

在正确搭建验证环境的情况下，飞控计算机故障验证操作通用流程如下：

1）打开数据集中交换程序(即 DataComm 通信框架中心服务器)、飞控 PC 端代理软件、各个仿真程序、主机端余度运行与重启过程监控软件和主机端故障注入软件及操作连接上网络服务节点。

2）给仿真飞控计算机上电。

3）等待主机端余度运行与重启过程监控软件显示各仿真设备节点以及飞控余度各节点状态正常。

4）操作主机端余度故障注入软件对需要验证的飞控余度节点注入测试故障。

5）观察主机端余度运行与重启过程监控软件对应被测对象飞控余度节点状态变化，即是否产生故障，根据具体故障类型，进入自动复位状态或是等待用户手动操作复位，复位后恢复正常运行状态，同时可查看其它非被测飞控余度节点中其它通道状态是否发生变化。

表 1 中“飞控余度的 20 个故障”包括：

1）同步故障导致的余度失效：故障模型使故障注入的飞控计算机余度不进行同步操作，不能与其它余度完成同步。

2）CCDL 故障导致的余度失效：故障模型使故障注入的飞控计算机停止 CCDL 的输入和输出，模拟 CCDL 故障状态。

3）通道故障逻辑导致的余度失效：故障模型使故障注入的飞控计算机停止通道故障逻辑的离散量输入和输出，模拟通道故障逻辑的故障状态。

4）飞控计算机易失存储器可持续故障导致的飞控计算机失效：故障模型持续的修改对飞控计算机输出指令有影响的状态或积分变量为一个错误值，引起飞控计算机余度输出指令与其它余度产生足够大的差异。

5）飞控计算机易失存储器瞬时故障导致的飞控计算机失效：故障模型连续多个(3 个以上)帧周期内修改对飞控计算机输出指令有影响的状态和积分变量为一个错误值，引起飞控计算机余度输出指令与其它余度产生足够大的差异。

6）飞控计算机输入信号/通信可持续故障导致的飞控计算机失效：故障模型停止本飞控计算机余度的数据输入，让本飞控计算机余度认为没有数据被接收进来，触发余度飞控应用的对应处理程序。

7）飞控计算机输入信号/通信瞬时故障导致的飞控计算机失效：故障模型连续多个(3 个以上)帧周期停止本飞控计算机余度的数据输入，让本飞控计算机余度认为没有数据被接收进来，触发余度飞控应用的对应处理程序。

8）飞控计算机输出信号/通信可持续故障导致的飞控计算机失效：故障模型停止本飞控计算机余度的数据输出，让其它飞控计算机余度发现本余度没有数据输出。

9）飞控计算机输出信号/通信瞬时故障导致的飞控计算机失效：故障模型连续多个(3 个以上)帧周期停止本飞控计算机余度的数据输出，让其它

飞控计算机余度发现本余度没有数据输出。

10）飞控计算机处理单元可持续故障导致的飞控计算机失效:故障模型停止本飞控计算机余度的所有软件(操作系统和应用软件)运行,但由于操作系统的一些底层操作无法被完全停止,因此主要是停止应用软件和操作系统对定时器中断的响应,模拟处理单元故障导致的软件完全停止运行的情况。

11）飞控计算机处理单元瞬时故障导致的飞控计算机失效:故障模型连续多个(3 个以上)帧周期停止本飞控计算机余度的所有软件(操作系统和应用软件)运行,但由于操作系统的一些底层操作无法被完全停止,因此主要是停止应用软件和操作系统对定时器中断的响应,模拟处理单元故障导致的软件在一段时间内停止运行的情况。

12）系统初始化软件故障导致的飞控计算机操作系统失效:这种故障模型无法在飞控软件运行过程中进行模拟,因为对应的初始化过程已经运行结束了,只能通过单独给预期发生故障的余度固化一个特殊修改后的软件映像,对应的软件映像中会在第一次上电运行的系统初始化时触发故障。

13）中断/异常管理软件故障导致的飞控计算机操作系统失效:故障模型在飞控软件正常运行的过程中,模拟中断/异常管理软件,不对正常中断进行处理,导致本余度飞控软件运行出现故障。

14）设备驱动软件故障导致的飞控计算机操作系统失效:故障模型在飞控软件中模拟设备驱动软件发生故障而不能正常工作,使得飞控软件无法进行数据的输入和输出。

15）内存管理软件故障导致的飞控计算机操作系统失效:这种故障模型无法在飞控软件运行过程中进行模拟,因为飞控软件在运行的内存分配都是静态的,对应的内存分配是在系统初始化时进行的,在飞控软件正常运行后对应的初始化过程已经运行结束了,只能通过单独给预期发生故障的余度固化一个特殊修改后的软件映像,对应的软件映像中会在第一次上电运行的系统初始化时模拟内存管理软件故障,使得操作系统在分配内存时出现故障。

16）运行调度软件故障导致的飞控计算机操作系统失效:故障模型模拟飞控软件中操作系统的运行调度软件发生故障,让各飞控应用软件分区中的部分分区不能被调度运行。

17）时钟管理软件故障导致的飞控计算机操作系统失效:故障模型模拟飞控软件中操作系统的时钟软件发生故障,让操作系统无法有效的获得时钟数据。

18）输入与监控软件故障导致的飞控计算机应用软件失效:故障模型模拟飞控软件中应用软件的输入与监控软件分区发生故障,不能有效的获取该余度飞控计算机的输入数据。

19）控制律解算软件故障导致的飞控计算机应用软件失效:故障模型模拟飞控软件中应用软件的控制律解算软件分区发生故障,不能计算出正确的输出指令数据。

20）输出表决监控软件故障导致的飞控计算机应用软件失效:故障模型模拟飞控软件中应用软件的输出表决监控软件分区发生故障,使得该飞控计算机余度不能输出指令数据。

实验结果为:

对于飞控节点单余度故障验证,余度运行与重启过程监控软件界面被测飞控计算机节点状态变化如下:

1）被测飞控计算机节点注入故障后,通道状态先由正常运行状态切换为故障触发状态,其它飞控节点的其它余度通道状态按照相对绝对通道转换关系反应出被测飞控计算机余度故障;

2）被测节点本余度通道状态切换为重启状态;

3）被测飞控计算机节点复位同步成功,并且恢复数据,余度系统计数也恢复正常。

对于硬件永久故障不可恢复验证余度运行与重启过程监控软件界面被测飞控计算机节点状态变化如下:

1）被测飞控计算机节点注入故障后,通道状态先由正常运行状态切换为故障触发状态,其它飞控节点的其它余度通道状态按照相对绝对通道转换关系反应出被测飞控计算机余度故障;

2）被测节点本余度通道状态切换为重启状态;

3）被测飞控计算机节点无法自动复位成功,手动在“主机端余度故障注入软件”点击该通道重启后,依旧无法复位成功,保持被切除状态。

被测 84 个测试用例全部通过实验。

5　结论

1）飞控计算机余度通道的 CPU 运行模式(命令-监控模式以及双命令模式)对通道重启流程的

大框架影响不大,区别主要在支路间同步算法。

2）飞控计算机单故障余度通道重启可以恢复第四章实验所罗列的 20 类故障,使重启余度与正常余度同步并恢复正常运行。而对于永久性的硬件故障,无法通过重启余度恢复。

3）余度重启机制效果稳定,能够在四余度飞控计算机的环境中,在单故障余度的情况下,使重启故障余度能够与正常运行的余度同步,从而提高飞控系统的可靠性,保证飞行控制系统的品质。

参考文献:

[1] 王轩, 马超, 闫闯, 等. 四余度飞控计算机架构及其余度管理算法[J]. 信息技术与信息化, 2022, 264(3): 48-51.

[2] 牛文生. 机载计算机技术[M]. 北京: 航空工业出版社, 2013: 8-10.

[3] 马超, 郭勇, 刘意, 等. 双双余度飞控计算机余度管理算法设计与实现[J]. 科技风, 2017, 307(1): 42-43.

[4] 欧阳润宇. 容错飞控计算机架构设计与虚拟化验证平台构建[D]. 成都:电子科技大学, 2018: 12-13.

[5] 刘鑫, 李梓衡. 一种无人机飞控系统容错设计方案[C]//中国航空学会. 2019 年(第四届)中国航空科学技术大会论文集. 北京: 中航出版传媒有限责任公司,2019: 981-987.

[6] 白雪琛. 无人机多余度飞行控制系统设计与应用研究[D]. 杭州:浙江大学, 2017: 20-21.

[7] 张锐. 四余度飞控计算机信号表决与监控算法研究[J]. 信息系统工程, 2018, 294(6): 142-144.

[8] 吴科, 吕文亭, 姚超雷. 浅析无人机软硬件余度设计方法[J]. 价值工程, 2022, 41(26): 127-129.

[9] 郭佳. 四余度电传系统"两两表决"故障分析[J]. 航空维修与工程, 2022, 370(4): 95-96.

[10] 中国航空工业集团公司西安飞机设计研究所. 一种四余度飞控计算机余度空中重启方法及装置: CN202211243234. X[P]. 2023-01-24.

[11] 郭勇, 段海军, 马倩, 等. 任务管理计算机系统余度软件关键技术研究[C]//《科技与企业》编辑部. 决策论坛——政用产学研一体化协同发展学术研讨会论文集(下). [出版地不详:出版者不详], 2015: 284-286.

作者简介

刘 鑫 女,硕士,工程师。主要研究方向:飞控软件设计、飞控系统设计。E-mail: 361626718@qq.com

韩建辉 男,硕士,工程师。主要研究方向:飞控电子设计、飞控系统设计。E-mail: 1093312324@qq.com

Research of redundant channel restart mechanism of redundant flight control computer subsystem

LIU Xin * HAN Jianhui

(AVIC The First Aircraft Institute, Xi'an 710089, China)

Abstract: The traditional fault redundancy process method is delayed excision, which causes degradation of flight control computer subsystem. If the residual redundancy fails again, the flight safety may be compromised. Some redundant channel faults can be restored to normal operation by restarting the redundant channel By analyzing the normal startup process of the redundancy channel and the restart process of the redundancy control application software in the flight control computer subsystem, the restart process in the dual-CPU command-monitor mode and the dual-command mode was studied. Finally, the simulation equipment node fault, flight control node fault and flight control redundancy fault were verified by experiments, and all 84 fault verification items passed the verification test.

Keywords: flight control computer; redundancy management; CPU operating mode; restart

* Corresponding author. E-mail: 361626718@qq.com