基于 ARINC664 帧的故障注入方法研究

刘光星[1] 焦方新[1]
LIU Guangxing JIAO Fangxin

摘 要 为完成 ARINC664 端系统被测设备的验证工作，测试其对错误消息的识别能力与鲁棒性，提升飞机飞行过程中的安全性和稳定性，需对 ARINC664 端系统向被测设备发送的数据包实施特定的故障注入操作。文章研究并实现了帧逻辑故障注入、软件故障注入两种方法：以 ARINC664 端系统芯片及相关硬件为载体，分别基于 Verilog 语言和 C# 语言进行开发；以 ARINC664 网络帧格式及网络配置文件为基础，针对不同验证场景完成特定故障注入或多类型故障混合注入；同时将两种方法集成至 Windows 可视化界面，能够在验证过程中更便捷、高效地构造激励数据。

关键词 ARINC664 协议；端系统；故障注入

doi：10.3969/j.issn.1672-9528.2025.12.004

0 引言

针对航空电子通信系统的复杂性和日益增强的高可靠性安全性的需求，网络通信在现代航空系统中的作用变得尤为关键[1]。ARINC664 协议作为一种基于以太网的航空数据通信协议，已广泛应用于各种飞行控制、导航、监控等系统中[2-3]。该标准采用以太网技术，为航空电子通信系统提供了高速、高效且可扩展的数据传输解决方案，但同时也带来了诸如链路故障、数据包错误等潜在风险。

为确保基于 ARINC664 协议的系统在各种复杂故障场景下的安全运行，故障注入技术成为验证和评估系统鲁棒性的重要手段[4]。通过在系统中注入虚拟故障，可以全面测试其故障检测、诊断和恢复能力，为系统设计优化提供关键性支持。本研究旨在提出一种基于 ARINC664 帧的故障注入方法，可模拟各种可能的通信故障，以此来评估系统的可靠性和容错能力[6-9]。通过分析 ARINC664 帧的结构特点，设计具有针对性的故障注入策略，能够有效地模拟链路中断、数据篡改、帧丢失等多种异常情况[10-14]。本研究不仅有助于验证航空电子通信系统的安全裕度，还为其后续的系统优化提供了实验基础。

1 ARINC664 端系统

ARINC664 端系统是航空电子全双工交换式以太网协议架构的关键组件，负责实现传感器、计算机、显示系统等航空电子设备与 ARINC664 交换网络的互联。是 ARINC664 协议（Part7）定义的核心功能模块，承担数据收发、流量整形、故障隔离等关键任务[15-18]。

ARINC664 端系统芯片通信过程如下：

芯片完成初始化，加载网络通信配置，通过配置获取通信发送 / 接收端口、发送 / 接收 VL 等内容信息。

主机发送消息时，根据网络配置的发送端口及端口最大长度信息，将主机待发送消息传递给芯片，由芯片按照以太网协议层次要求进行消息处理，最终通过芯片外部 PHY 芯片发送到网络中。主机将新消息的地址、长度写入寄存器。发送消息创建模块执行查询命令，根据 PortIndex 访问发送端口配置表获取 SubVLIndex，访问 SubVL 外存管理模块。SubVL 外存管理模块在以下条件均成立时，返回允许创建，否则返回创建失败。新消息长度小于等于此 SubVL 外存剩余空间大小；新消息长度大于 0 小于等于 8 192，且小于等于此 SubVL 配置的最大消息长度；消息所属端口队列未满；端口索引不超过发送端口数量 -1，且发送端口数量不等于 0；VL 索引（SubVL 的高 8 位）不超过发送 Vl 数量 -1，且发送 Vl 数量不等于 0。

如果允许消息创建，主机将消息写入 SRAM，或者由芯片启动 DMA 将消息从主机内存搬运至 SRAM；请求将内部 SRAM 数据写入外存，写入地址由 SubVL 的头指针决定；SRAM 数据写入外存后，通知 SubVL 外存管理模块更新头指针，通知 SubVL 更新消息计数。

SubVL 内部当消息计数不为 0 时，产生组帧请求，输出第一分片标识、 SubVLIndex 信息；VL 对内部 SubVL 的请求进行仲裁，选中一个 SubVL 进行输出。

发送组帧仲裁模块对 256 路 VL 进行仲裁，选中一个 VL

1. 西安石油大学陕西省油气井测控技术重点实验室
陕西西安 710065

授权，将此 VL 相关信息输出至发送组帧模块，请求组帧。发送组帧模块接收组帧请求后，则根据 VL 的尾指针访问外存，获取消息的 PortIndex、长度等信息；根据 PortIndex 访问发送端口配置表，根据配置信息开始组帧，包括 MAC 头部、L/T 字段、IP 头部、UDP 头部，将这些信息写入 SubVL 对应的发送片上缓冲。

组帧完成后，通知 SubVL 组帧完成；此时 SubVL 可以向 VL 请求发送；VL 等待 BAG 计时完成，按照公平原则读取 SubVL 的发送请求信息，输出至发送仲裁。

发送仲裁按照公平原则获取 VL 发送信息，将仲裁结果写入 FIFO。发送控制模块在 FIFO 非空时读取 FIFO，根据获取的 SubVL 访问发送片上缓冲，根据片上缓冲信息读取外存的 Payload，发送至 MAC，通知 SubVL 发送开始；如果发送的是最后一个分片，则更新 SubVL 外存管理模块的尾指针；发送开始后，每个 SubVL 均如此。

发送时，每个 SubVL 在外存的起始地址、占用大小由配置表配置。发送外存最大支持 32 MB。每个消息占用的空间大小固定为最大消息长度乘以发送数量除以 8，单位为 8 字节，每个 SubVL 外存大小应为每个消息占用的空间大小的整数倍。

主机接收消息时，芯片通过外部 PHY 接收网络消息后，按照 ARINC664 端系统协议要求进行完整性检查和余度管理，最后将接收正确的消息提交主机，由主机进行消息处理 [19]。MACAMACB 各自接收网络帧，分别提交给接收帧控制 AB 模块处理；添加接收时标，同时将源 IP/UDP 信息和 payload 写入接收帧缓存。接收帧控制 A/B 模块分别对网络 A/B 的帧进行有效性判断，包括 CRC、帧长、常数域字段、IP 校验和等；任一判断不通过即丢弃，统计相关计数；若全部校验通过，则检索 VL 配置表获取对应配置信息，进一步判断该 VL 是否被允许在对应网络中接收数据；若 VLID 不存在配置表中，或不允许在相应网络接收，则丢弃，统计相关计数。判断此帧为 ICMP 或 UDP 帧；若为 ICMP，则判断此 VL 是否使能了 ICMP，若未使能则丢弃；若为 UDP，且为第一分片，则提取｛VLID, IP-DEST,UDP-DEST. PORT｝进行端口配置表的查找；若端口不存在，则丢弃。并进行 MAC 级的完整性检查；如果 VL 使能 RM，则传输信息至 MAC·RM 模块处理。若余度管理通过，则将帧信息写入接收帧状态 FIFOA；否则不写，记录错误计数。如果禁止 RM，则不进行 MAC 余度管理；A/B 网络将帧信息写入各自的接收帧状态 FIFOA/B。两路接收消息处理模块读取各自的接收帧状态 FIFOA/B，将帧信息输出至相应的接收消息重组模块，判断是否丢弃此帧。消息重组通过后，根据端口号访问接收端口外存管理模块，获取当前端口消息头部外存地址和消息 Payload 外存地址，从相应的接收帧缓存 A/B 读取数据，写入外存；若接收端口外存管理模块返回端口队列满，则丢弃该分片；队列未满且接收到分片消息的最后一片时，通知接收端口外存管理模块更新外存地址信息和端口队列头指针；更新消息头部；主机通过写接收消息的端口号和内存地址，通知接收消息命令模块；接收消息命令模块访问接收端口外存管理模块，获取当前端口号的尾指针，从外存读取消息，读取完成后更新尾指针和消息队列，启动接收 DMA 将消息搬运至主机内存。接收时，每个接收端口在外存分配的起始地址、大小由配置表配置。接收外存最大支持 32 MB。每个消息分配的空间包括 1 个 head、A 网络状态和 Payload、B 网络状态和 Payload，以 8 字节为单位进行空间分配。其中 head 占用 1 个单位，状态和 Payload 占用的空间大小固定为 (3+S)，S 为接收消息最大长度除以 8；每个接收端口外存大小应为 ((3+S)×2+1) 的整数倍，每个接收端口分配的外存总大小为 ((3+S)×2+1)×N，N 为队列深度。通过上述过程完成收发通信。

ARINC664 端系统通信流程如图 1 所示。

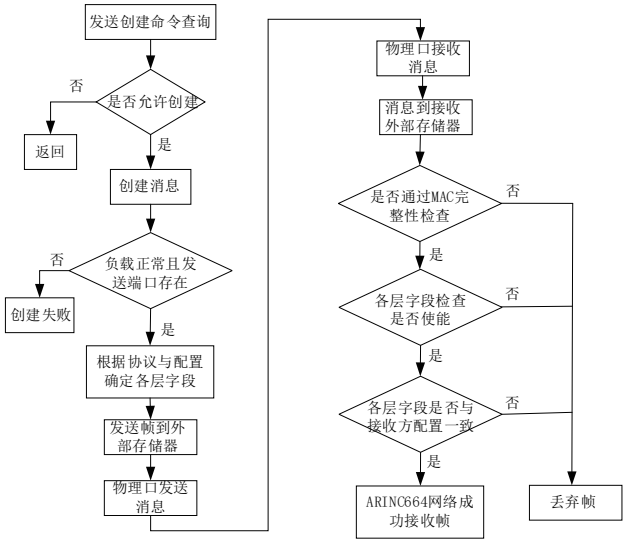

图 1 ARINC664 端系统通信流程图

2 ARINC664 帧格式

ARINC664 帧格式基于 IEEE 802.3 Ethernet，并增加了前导码、界定符、MAC 地址、VLAN 标签、以太网类型、帧校验序列、帧间隔等关键字段，以满足航空电子系统的确定性、可靠性和实时性需求。

其中前导码、界定符由 PHY 产生，数据负载自动生成，不做故障注入处理。本论文主要针对 MAC 层、IP 层、UDP 层进行故障注入。

需要故障注入的字段结构如下：

（1）目的 MAC 地址格式：第 0 位～第 15 位为 16 位 VLID；第 16 位～第 47 位为 32 位常数域。

（2）源 MAC 地址格式：5 位固定域；3 位接口 ID；8 位设备 ID；8 位网络 ID；24 位固定域。

（3）IP 包头部结构包括：32 位源 IP 地址，32 位目的 IP 地址，16 位头部校验和，8 位协议，16 位标识等。

（4）UDP 头部结构包括：16 位源端口号，16 位目的端口号，16 位 UDP 长度，16 位 UDP 校验和。

3 故障注入

针对 MAC 层、IP 层、UDP 层字段的故障注入，本文提出了两种方式，分别是逻辑注入和软件注入。

3.1 基于 FPGA 的逻辑故障注入

对于逻辑的故障注入，使用 Verilog 语言定义了 22 种错误类型，通过界面选择错误类型可以实现不同的故障注入。逻辑故障注入界面如图 2 所示。

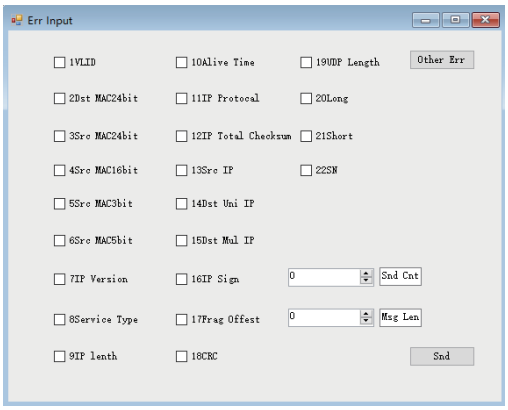

图 2 逻辑故障注入界面

其中故障类型 1~17 和 19 是在协议和配置的基础上加一个使其产生错误的数据包。对于错误类型 18，正确的 MAC CRC 的计算遵循 CRC-32 标准，CRC-32 的计算基于多项式除法，使用的生成多项式为：

$$G(x)=x^{32}+x^{26}+x^{23}+x^{22}+x^{16}+x^{12}+x^{11}+x^{10}+x^{8}+x^{7}+x^{5}+x^{4}+x^{2}+x+1 \quad (1)$$

对应的十六进制表示为 0x04C11DB7。在 ARINC664 帧中 MAC CRC 的校验范围是从目的 MAC 开始到 Payload 结束，即：

CRC=CRC32(Destination MAC || Source MAC || EtherType || Payload)

Payload 中包含着负载和一个字节的 SN 号，逻辑注入 MAC CRC 故障的方法是将 Payload 中的最后两个字节的负载和一个字节的 SN 号删掉，从而使 MAC CRC 的校验出现错误。

对于错误类型 20、21，ARINC664 协议规定的非分片包的帧长应为 64~1 518 字节，对于小于 64 字节的数据包，芯片会自动给数据负载补 0 使帧长达到 64，而对于帧长大于 1 518 字节的数据包，在不允许分片的情况下会组帧失败。这里对超短帧故障注入的方法是让芯片按照给定的负载发送数据，禁止补充功能，从而使系统芯片发出小于 64 字节的数据包；对超长帧故障注入的方法是对不允许分片的数据包帧长的最大限制增加到 2 047 来实现超长帧故障。

对于错误类型 22，是 UDP 消息的 SN 号故障，正常发送队列消息的 SN 号应从 0~255 递增，然后返回到 1 再递增到 255，如此循环，这里使用的方法是将队列消息的 SN 号改为 0、0、1、1、2、2……，每两帧消息时的 SN 号相同并依次递增，若接收方使用了 SN 检查，则只能收到第一帧 SN 为 0 的数据，丢弃后面的数据并上报 SN 错误。

3.2 软件故障注入

软件故障注入的方法是将数据包除负载外的所有字段全部用软件写入，负载根据软件写入的负载长度递增填充，其他字段不再基于协议和配置。为了方便地进行故障注入，这里将根据协议和配置先将除负载外的其他字段提取出来显示在界面上，然后根据需求修改相关字段进行不同的故障注入。软件故障注入界面如图 3 所示。

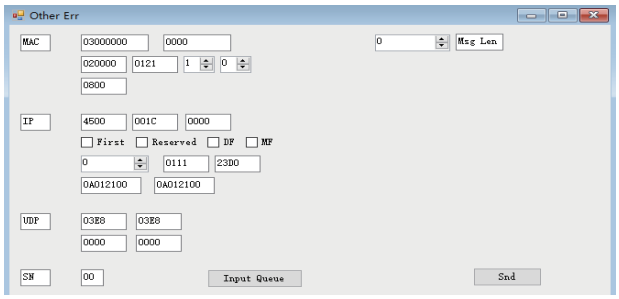

图 3 软件故障注入界面

4 故障注入结果分析

通过上述两种方法得到故障注入的结果，使用了可显示 MAC CRC 的工具来识别 MAC CRC 故障，使用 Wireshark 抓包来识别其他故障。

基于配置与协议的正确包的格式应为目的 MAC 常数域—VLID—源 MAC—IP 层字段—UDP 层字段—数据负载—递增的 SN 号。这里将 MAC 常数域配置为 0x03000000，VLID 配置为 0，目的 IP 和源 IP 配置为 10.1.33.0，IP 头部总长度为数据负载 +28 自动生成，UDP 总长度由数据负载 +8 自动生成。下面分析两种典型的故障。逻辑注入 MAC CRC 故障如图 4 所示。

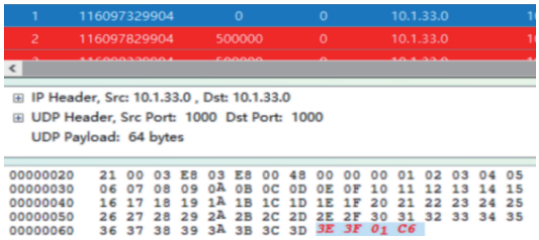

图 4 逻辑注入 MAC CRC 故障

发送的负载为64个字节，则消息中的负载应为0x00~0x3f，这里拿走了0x3e、0x3f两个字节的负载和SN号，导致图中最后四个字节的MAC CRC错误，负载变成了0x0-0x3d，此时最后一个字节的负载0x3d作为SN号，即SN=61。

软件注入多种类型混合故障如图5所示。

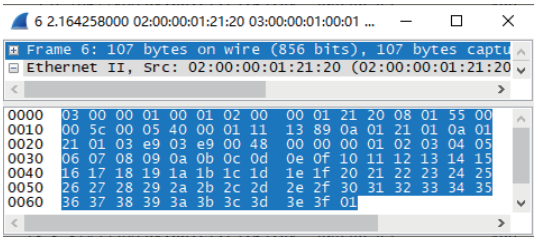

图5 软件注入多种类型混合故障

逻辑注入其他类型的故障多为在正确的字段上加1，如目的MAC常数域注入为0x03000001，VLID注入为1等。软件故障注入可将各字段在位宽决定的范围内根据需求注入，例如可以构造后包SN号小于前包的两帧数据并依次发送。以上功能通过Wireshark抓包观察数据帧的各字段，发现皆可以成功实现故障注入。

综上所述，这两种方法皆可实现故障注入，其中通过逻辑故障注入更为快速，可以更加方便地构造大量数据包的故障，软件故障注入则更加灵活，可以对不同字段进行特定的故障注入，也可以实现多种故障类型的混合注入，从而测试被测端系统设备对不同故障优先级的判定。将这两种方法集成到Windows的界面上，可以使验证过程中的稳定性、鲁棒性验证更加快速方便。

参考文献：

[1]VERA-SOTO P, VILLEGAS J, FORTES S, et al. An event-driven link-level simulator for the validation of AFDX and ethernet avionics networks [J]. Aerospace,2024,11(4):247.

[2]WANG H C, NIU W S. Design and analysis of AFDX network based high-speed avionics system of civil aircraft[J]. Advanced materials research,2012,462:445-451.

[3] 陈芳，吕广喆，林伟，等 .AFDX 网络配置测试技术研究 [J]. 西北工业大学学报 ,2024,42(3):514-520.

[4] 胡磊，柳杨，郭卿超，等 . AFDX 网络设备故障诊断技术 [J]. 西北工业大学学报 ,2023,41(3):546-556 .

[5] 张荣华，刘红红，王瑶，等 .AFDX 网络端系统芯片设计与实现 [J]. 电子技术应用 ,2016,42(4):11-14.

[6] 于峰，李雯，张逸飞 . 一种 AFDX 网络端系统确定性仿真方法 [J]. 航空计算技术 ,2018,48(4):97-99.

[7]PIEPER P, HERDT V, DRECHSLER R. Advanced embedded system modeling and simulation in an open source RISC-V virtual prototype[J]. Journal of low power electronics and applications,2022,12(4):52.

[8] 张丽 . AFDX 端系统虚拟链路层的研究与仿真 [J]. 工业控制计算机 ,2022,35(2):8-10.

[9] 朱晓飞 . AFDX 总线传输特性测试要求标准指标体系的确定 [J]. 航空电子技术 ,2021,52(1):61-66.

[10]DÍAZ E, MATEOS R, BUENO E J, et al. Enabling parallelized-QEMU for hardware/software co-simulation virtual platforms[J]. Electronics,2021,10(6):759.

[11]CHU G, ARIKAN O, BENDER G,et al.Discovering multi-hardware mobile models via architecture search[EB/OL].(2021-04-23)[2025-11-22].https://doi.org/10.48550/arXiv.2008.08178.

[12]JIANG S N, OU Y H, PAN P T, et al.UMOC: unified modular ordering constraints to unify cycle- and register-transfer-level modeling[C]// 2021 58th ACM/IEEE Design Automation Conference (DAC).Piscataway:IEEE,2021:883-888.

[13]JIANG S N, PAN P T, OU Y H,et al.PyMTL3: a python framework for open-source hardware modeling, generation, simulation, and verification[J].IEEE micro, 2020,40(4):58-66.

[14]WATSON V, BEJIGA M. Dependability analysis of the AFDX frame management design[C]// Computer Safety, Reliability, and Security.Berlin:Springer, 2018:188-202.

[15] 王伟鹏，张延园，姚进华 . 基于 FPGA 的 ARINC664/AFDX 端系统设计 [J]. 微处理机 ,2009,30(2):1-4.

[16] 张志平，李晓庆，周耿 . 端系统余度管理功能设计和实现 [J]. 电子测试 ,2020(7):79-80.

[17] 邱收 . 带擦除信息的里德所罗门乘积码解码器的 CMODEL 实现 [J]. 计算机与数字工程 ,2004(4):54-56.

[18] 王宁，牛玥瑶，崔西宁 . 基于 QEMU 仿真的 ARM 多核启动技术研究 [J]. 航空计算技术 ,2023,53(5):96-99.

[19]MOON J, JEONG H.The study: disk information acquisition in QEMU-based virtualization[J].International journal of recent trends in engineering and research,2019,5(6):6-10.

【作者简介】

刘光星（1975—），男，陕西宝鸡人，硕士，副教授，研究方向：自动化装置、智能控制、钻机控制等。

（收稿日期：2025-09-19 修回日期：2025-12-08）