WIND™

集成模块化航空电子设备的安全关键性软件开发

Paul Parkinson
风河公司资深系统架构设计师

Larry Kinnan
风河公司航空和国防资深工程专家

WHEN IT MATTERS, IT RUNS ON WIND RIVER

文档摘要

本技术文件介绍了安全关键性航空电子系统开发中的最新趋势，对集成模块化航空电子设备（IMA）架构和标准的出现，及其对遵循标准的商用现货（COTS）实时操作系统（RTOS）开发的影响进行了讨论。

目录

介绍

很多航空电子系统通过定制的硬件和软件成功地进行了开发。但是在最近几年，定制系统的完全生命周期成本迫使原始设备制造商（OEM）考虑使用基于商用现货的系统。与此同时，业内出现了从联合架构偏离的显著趋势，其中每个单独的子系统向一般的计算平台执行专门的功能，可以供多种类型的应用程序使用，而且有时可以同时运行多个应用程序。这种方法被称为集成模块化航空电子设备（IMA），可以减少子系统的数量、减轻重量、减少能耗并减少平台的冗余。有一些民用和军用研究项目正计划对IMA架构做出定义，虽然这些项目采用的方法不同，但是要实现的高层次目标是一样的：

- **通用处理子系统**：允许多个应用程序共享和重用相同的计算资源，这样可以减少需要开发的子系统数量、更有效地利用系统资源、并为将来的扩展留出空间。
- **软件抽象**：不仅将应用程序与基础总线架构隔离，而且也从基础硬件架构分离出来。这样可以增强应用程序在不同平台间的可移植性，而且允许引入新的硬件来替换过时的架构。
- **重用最大化**：IMA架构应当允许对遗留代码的重用。这样可以减少开发时间，同时允许开发人员在不进行大量修改的情况下对已有的应用程序进行重新部署。
- **变更成本**：IMA架构将平台上在同一个处理器上执行的组成部分分离，简化了影响分析，有利于重用并降低再测试成本，因此能够降低变更成本。

IMA也有利于对功能性不断增加的应用程序提供支持，包括对复杂应用程序之间的互动提供支持，例如平视显示器（HUD）、地图显示系统以及气象雷达显示。

虽然市场上涌现出了许多IMA架构和标准，但是ACR规范1和ARINC规范6532在航空电子业得到了最广泛的应用。ACR规范处理架构方面的考虑事项，而ARINC规范653则在高层次上定义了IMA架构软件应用的实例。这些规范和其他IMA标准对软件架构，尤其是对于商用现货供应商提供的实时操作系统的实现提出了新的要求。风河公司开发出了专门应对这些需求的风河开物RTOS 653平台，并在C-130 航空电子现代化计划和767空中加油机4上加以采用。波音公司选择了风河开物RTOS 653 平台来开发其Boeing 787 Dreamliner Common Core System (CCS)5。其他的风河客户，包括EADS6在内，也使用此平台来开发航空电子系统和安全关键性的应用程序。

以下部分讨论了集成设备软件平台支持IMA应用程序的技术要求，并说明风河开物RTOS 653平台（参见图1）是如何满足这些要求的 – 尤其是在进行ARINC 653应用程序开发时。

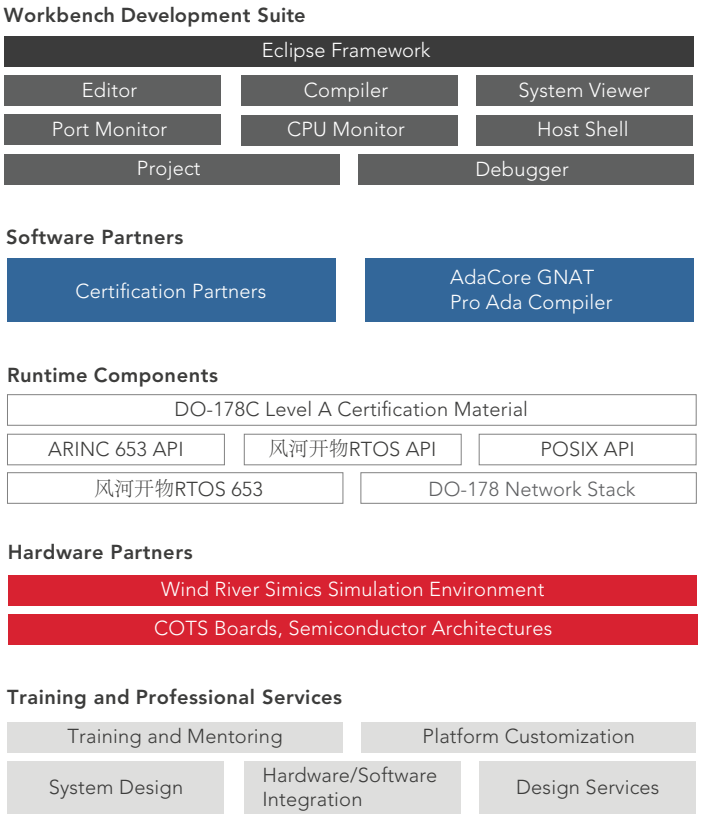

图1：风河开物RTOS 653平台

通过风河开物RTOS 653平台进行应用程序开发

ACR规范定义了IMA中被广泛使用的两个重要概念：空间分区和时间分区。

空间分区

空间分区定义了同时运行在同一个计算平台的多个应用程序间的隔离要求，也被称为模块。在这种模式下，运行在一个IMA分区内的应用程序必须不能相互剥夺共享的应用程序资源或者实时操作系统内核提供的资源。这通常由处理器内存管理单元（MMU）强制的不同虚拟内存上下文来实现。

这些上下文在 ARINC 653 中被称为分区。每个分区包含一个应用程序，具有自己用于动态内存分配的堆和用于应用程序进程的栈（ARINC中执行上下文的术语）。这些要求对实时操作系统和语言运行时系统的设计和应用产生影响。例如， 风河开物RTOS 5.5 为应用程序采用共享的虚拟地址空间，并通过内存管理单元提供基本的支持来防止故障应用程序对程序代码进行偶然或恶意的访问操作，同时不会引起完全进程模式下的性能开销。 风河开物RTOS 6.x 和 风河开物RTOS 653提供的环境则使用内存管理单元来强制单独的上下文。

但是在IMA环境中，仅仅采用内存保护并不能防止运行在一个分区内的故障应用程序消耗系统资源，从而对运行在另一个分区的应用程序产生有害影响。当在同一个处理器上运行多个不同优先级的应用程序时，这可能会造成严重后果。单独使用完全进程模式无法解决此问题，需要开发一个实时操作系统来专门处理IMA的需求。风河开物RTOS 653操作系统针对此目的而专门设计，并且能够在内核架构的应用中支持ARINC 653模型（参见图2）。

- 模块OS与计算平台（核心模块）直接作用，为每个分区提供全局的资源管理、调度和健康监控。模块OS还使用板级支持包（BSP），在不同处理器和硬件配置上运行所需的硬件特定配置。
- 分区OS使用风河开物RTOS微内核来实现，并在分区内部提供调度和资源管理。与模块OS的通讯采用专用消息传送接口进行，以确保健壮性。分区OS还提供ARINC 653 APEX（应用程序/执行程序）接口供应用程序使用。

这种架构代表了“航空电子设备架构中的分区：要求、机制和保障” 7中描述的虚拟机方法，提供实现ARINC 653要求的途径，同时提供了单内核应用或UNIX类型应用无法轻易实现的灵活、可扩展的框架。在这种框架中，单独的分区可以使用内存保护容器实现，其中可以包含进程、对象和资源，而分区则由内存管理单元（虚拟机）强制进行。每个分区具有自己的栈和本地堆，这些堆栈无法被运行于其他分区的应用程序侵占。分区还能防止因运行在其他分区的应用程序进行错误的内存访问而引起的干扰。

图2显示了风河开物RTOS 653 架构的概念应用。实时操作系统的特点是具有单独、共享的分区OS库（共享只读文本）来简化配置、测试和验证的能力；针对一个或多个分区，分区OS也可以具有分别的不同配置（称为MPOS，或多分区操作系统）。

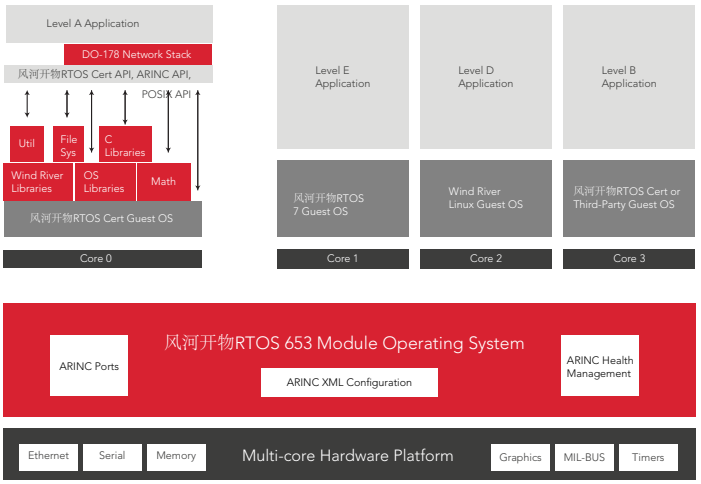

图2：风河开物RTOS 653 架构

时间分区

时间分区定义了在同一个计算平台上运行多个应用程序的隔离要求。这保证了一个应用程序占用处理器时不会超过规定的时间，以避免对其他应用程序产生不良影响。ARINC 653定义了采用基于分区的调度来解决此问题。一个分区被分配给一个固定长度的时间槽，可以给其他分区分配类似或不同长度的时间槽。在一段时间槽中，一个分区可以使用自己的调度策略，但是当此时间槽结束时，ARINC调度程序会强制将上下文切换到调度表中的下一个分区。这种模式具有足够的灵活性，能够允许已有的应用程序或单独开发的新IMA应用程序运行在一个核心模块中。不过，由于需要进行分区调度、验证时间边界和调度程序没有被侵犯、并采取相应的纠正措施，因此这种方法会不可避免地增加复杂性。

在风河开物RTOS 653中，模块OS执行单独分区的ARINC 653调度。在每个时间槽内，分区OS使用风河开物RTOS调度程序来执行抢占式、基于优先级的调度（参见图3）。这意味着所有的进程级别调度都在分区空间内发生，即使在高系统时钟速度时（时钟速度高于1毫秒），也能保证系统具有更好的收缩性和稳定性（抖动最小化）。这种方法也完全采用了优先级天花板策略协议来防止不受限制的优先级反转（参见下一节，“优先级反转、优先级继承和优先级天花板策略”）。

风河开物RTOS 653 的实现完全符合 ARINC 653 附录2的第1部分(653-2)。8 并且支持可选的基于模式的调度，这种方式可以针对不同的飞行模式或分阶段的初识化预先定义16个调度程序。模式间的转换通过受限制的API调用arincSchedSet()来实现，并且可以在下一个主要框架、下一个分区窗口或下一个定时器边界执行。在被采用之前，由健康监控系统（HMS）对新的调度程序进行验证（参见后面“健康监控系统和重新启动”部分的讨论）。

ARINC 653提供了高度确定性的调度方法，这种方法可能会导致过度的分区空闲时间或高时钟速率以保证数据的高速采样，因此可能不适合某些应用。为了适应这些类型的应用，风河开物RTOS 653提供了一个用于分区优先级抢占式调度（PPS）的选项。这种方法允许松弛挪用，使指定的分区能够使用在确定性ARINC调度下空闲的时间。

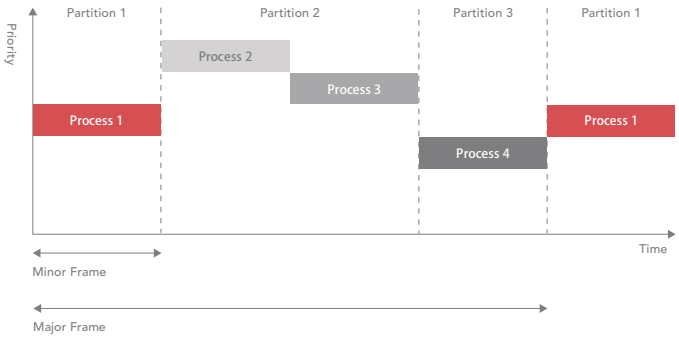

图3：风河开物RTOS 653时间分区

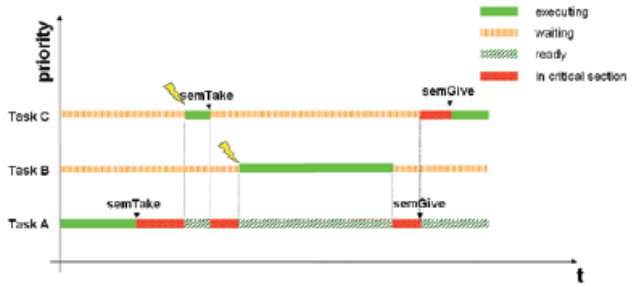

图4：优先级反转示例

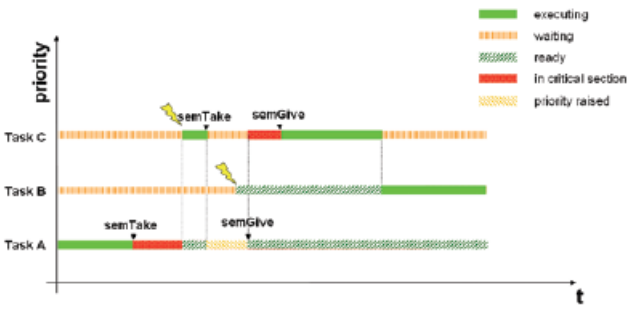

图5：优先级继承示例

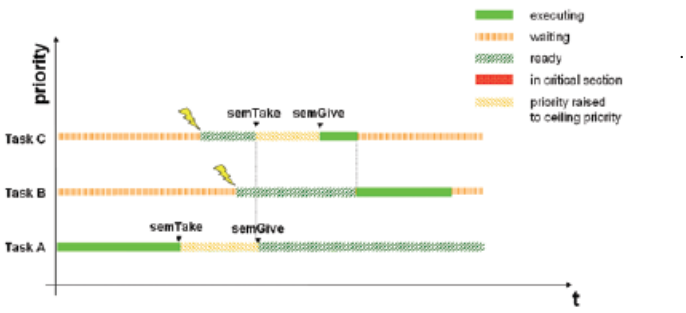

图6：优先级天花板策略示例

优先级反转、优先级继承和优先级天花板策略

优先级反转是设备软件设计人员需要关注的一个问题。当一个高优先级的任务需要获取一个互斥体（或二进制信号量）时，如果互斥体被一个低优先级的任务占用，则这个高优先级的任务就无法运行，同时低优先级的任务也会因中等优先级任务而无法运行，此时就会发生优先级反转（参见图4）。

在很多实时操作系统中，通过应用优先级继承协议来解决此问题。在这种方案中，每个互斥体与一个优先级关联，而持有此互斥体的任务的优先级会上升到与请求此互斥体的任务中最高的优先级，如图5所示。使用优先级继承或优先级天花板策略协议时支持速率单调分析（RMA），但是当使用优先级继承时，由于可能发生联锁的情况，因此可能需要进行附加分析来确定最坏情况下的执行时间。

优先级天花板策略协议是一个可选方法，主要用于防止联锁。在这种方式中，互斥体在初始化时被赋予一个比可能请求它的任何任务都高的优先级。当一个任务锁定此互斥体时，就被提升到优先级的最高限度（天花板）（参见图6）。一些实时操作系统开发商在优先级天花板策略协议中采用了专有实现，而风河公司则采用了POSIX9来实现，这是因为POSIX是国际上认可的标准，能够方便地将遗留应用程序移植到运行 风河开物RTOS 653 的IMA平台上（参见“异构应用程序支持”部分）。

风河的风河开物RTOS 653扩展了这一概念，在编译时为系统集成商提供了限制并发阻塞APEX调用数量的能力。这通过对内核工作线程数量的静态配置来实现。（工作线程代表进行APEC调用的ARINC进程执行内核操作）。当所有的工作线程都在忙着处理API调用时，下一个请求核心交互的APEX调用将会造成API调用锁定，直到某一个工作线程可用为止。这一点通过对定义在分区基础上可允许的操作和资源的配置数据来强制实现。配置数据与分区及其应用程序分开，这样可以在不改变分区应用程序本身的情况下对变更进行隔离和重建，从而大大降低了变更成本。

ARINC 653应用程序开发

ARINC 653 APEC有时也被称为ARINC 653 API，提供了操作系统和应用程序软件之间的一般用途接口。ARINC 653 API还提供了一个抽象层，可以将遵循ARINC 653的特定实时操作系统的安装详细信息对应用程序和核心模块的基础架构进行隐藏。这有助于应用程序移植到其他ARINC 653平台上，对于飞行器电脑等要求双重冗余、甚至三重冗余系统（例如波音777）10的任务关键性IMA系统来说，这是需要考虑的一个重要因素。

ARINC 653 APEX也提供了一个静态系统配置和初始化的模式。在这种模式中，ARINC进程的数量事先已经知道，而且这些进程通过使用CREATE_PROCESS() API的启动代码在分区中创建。所有其他分区对象从分区堆中创建，分区被创建后，通过SET_PARTITION_MODE(NORMAL)调用激活。

此时分区OS调度程序被激活并开始在分区内对进程进行调度。应用程序一旦被启动后，就不能再动态创建其他对象或进程。这样可以保证安全关键性应用程序有控制、确定性的启动顺序和对资源确定性和固定的使用。

ARINC 653 也为分区内部的通讯和分区间的通讯提供了良好的结构。ARINC 653 黑板和缓冲区可以协助分区内部的通讯。黑板提供了一次写入/多次读取的方便机制；缓冲区提供了发送和接收消息的能力，这些消息始终按照先进先出（FIFO）的顺序存储，但是接收进程既可以按照先进先出的顺序，也可以按照优先级的顺序接收它们。此外，信息量和事件也可以用于同步。

ARINC 653 端口可以方便分区间的通讯。常驻于同一个IMA隔间或另一个IMA隔间的同一个处理器上或另一个核心模块的端口可以使用同样的命名方案。这样可以防止应用程序生成依赖于架构和/或依赖于配置的假设，从而提高可移植性，并方便系统集成商进行重新配置。风河开物RTOS 653 提供的ARINC 653端口工具还允许假端口的定义和使用，通过将一个ARINC采样或队列端口连接到模块OS设备驱动程序来实现模块间的通讯，同时为应用程序提供标准的ARINC API。

异构应用程序支持

虽然很多IMA应用程序都是从头进行开发的，但是在联合系统中也存在大量的现成应用程序。这些应用程序可能是用不同的编程语言开发，并且使用不同的调度模式，但是仍然需要在IMA环境中相互通讯。

风河为运行在单独ARINC分区内的异构应用程序提供支持，从而满足了这一需求。这种方式可以允许一个 Ada 95 应用程序使用Ravenscar受限任务配置文件在风河开物RTOS 653分区OS上运行（Parkinson & Gasperoni对此进行了详细探讨11）。遵循POSIX的应用程序可以类似的方法在分区内运行，而这些应用程序间的通讯则使用ARINC 653端口实现（参见图7）。

系统配置

ARINC 653架构通过使用系统和分区配置纪录（在IMA圈内也称之为系统蓝图）来保证资源的可用性。这样做的目的是使由一个或多个原始设备制造商开发的IAM应用程序的配置能够应用到由系统集成商配置的共享IMA平台上。分区配置纪录定义每个OEM应用程序的特性，包括内存要求、处理器要求和ARINC端口的使用。系统配置纪录定义IMA平台的性能和容量，并引用和验证单独的分区配置纪录。这种方案使系统集成商能够保证应用程序需求在性能和平台方面的一致性，以及个别的应用程序不会使用超出为其分配的资源。

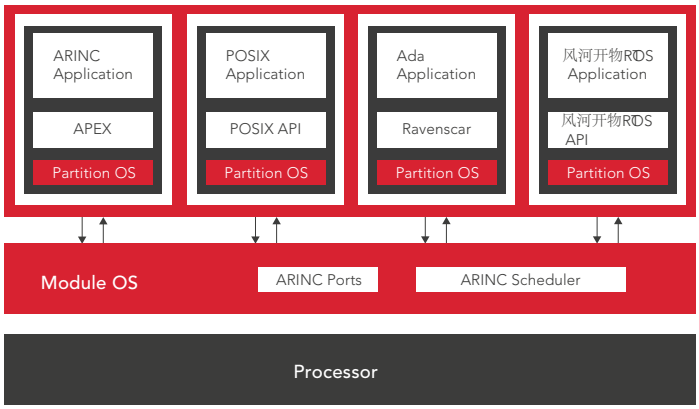

图7：风河开物RTOS 653异构应用程序支持

ARINC规范653的当前版本仅提供了配置纪录结构和内容的高级定义，具体的实现方法则交由RTOS应用方处理，但是也提供了一个基于XML的示例配置。风河开物RTOS 653采用以下步骤：

步骤1：当系统初始化时，启动代码加载模块OS和系统及分区配置纪录。
步骤2：模块OS对自身进行初始化，启动自己的子系统。
步骤3：模块OS加载应用程序分区及分区上的应用程序。

这一流程将模块OS二进制映像和系统及分区配置纪录二进制映像的配置与分区应用程序分离。这样可以分别开发单独的应用程序和子系统，然后再将它们方便地集成到目标文件系统。单独的分区应用程序也可以用直截了当的方式升级，而无需对模块OS的配置进行改动，从而显著减少了重新验证的工作，并为原始设备制造商和系统集成商提供了更多的灵活性。

风河开物RTOS 653对基于XML的配置示例进行了扩展，为应用程序开发人员、平台提供商和系统集成商提供了一套完整、经过验证的工具集，以及用于配置和初始化IMA平台的数据文件。这一被称为独立编译连接和加载（IBLL）的过程降低了变更成本，同时提供了一个可以完全配置的运行时环境，并且完全实现了DO-297“集成模块化航空电子设备（IMA）开发指南和验证考虑事项”12中规定的目标。

无需重新编译整个应用程序或平台就可以对系统和分区配置进行更改，从而在升级和修改现有系统时大大降低了系统集成商进行影响分析的工作量。由于在风河开物RTOS 653中用来生成配置纪录的工具从XML配置数据中直接生成二进制数据，因此非常容易使用，而且比其他实现方法更易于被验证。这些工具通常依赖更为通用的机制（例如C语言编译器）来生成供系统使用的二进制配置数据。

健康监控系统和重新启动

ARINC 653 定义了IMA系统中的健康监控器（HM）概念。健康监控器负责“监控硬件、应用程序和操作系统的故障和失效”，并且应当“隔离故障以防止失效蔓延”。

虽然这个概念看起来很简单，但是在实践中则比较复杂，需要有一个完善的系统范围健康监控器来跟踪错误并执行重新配置和恢复工作。对单独某个故障做出的反应取决于故障的性质、故障的严重程度、以及系统集成商定义的故障管理策略。

作为风河开物RTOS 653 架构的一个固有组成部分， 风河开物RTOS 653 HMS是一个完善的框架，能够完成ARINC 653的所有要求，并为需要使用动态重新配置（特别是基于模式的调度）的系统集成商提供了相关的扩展。风河开物RTOS 653 HMS的设计和安装内容极为丰富，因此这里只能提供一个概览。

HMS架构包括驻留在单独分区中的系统范围HM服务器和HM代理（在ARINC 653中被称为进程级处理程序），并且提供对模块OS的支持。HMS处理系统中需要被关注的事件，这些事件被称为故障，虽然它们既可能是负面事件，也可能是正面事件，例如硬件异常或超过限值（在软件中使用警报代表故障）。此框架还支持消息，用于记录的另一种事件类型或系统集成商配置的其他行为。注意HMS的使用在 风河开物RTOS 653 中并不限于ARINC的支持，从而为应用程序开发人员提供了更多的灵活性。

此框架提供了通过三种服务类型（警报检测、警报纪录、警报响应）在三个层次上（进程HM、分区HM、核心模块HM）进行健康监控的能力。分区HM和模块HM是表格驱动的，提供了代码和适当处理程序间的映射。为了提高可移植性，框架使用了ARINC 653中的错误代码定义，包括错过最终期限、数值错误、非法请求和电源故障等事件。对框架进行配置以及创建供系统运行时使用的表格驱动映射时，采用的是基于XML的配置数据。警报响应取决于错误级别：模块级别的响应包括复位和关机；分区级别的响应包括重新启动分区。

创建分区时，采用冷启动来分配并初始化分区对象；而在对分区进行重新初始化或重新启动时则采用热启动方法，将分区对象重新初始化，但是并不进行分配。对进程错误的响应是应用程序驱动的，采取的行动取决于错误类型及其上下文。为了便于热启动，风河开物RTOS 653 支持持久数据类型，这种数据类型能够在热启动操作中保留关键数据。此方法在这些情况下可以简化操作并提供了一个增加启动速度的机制。

安全关键性系统开发工具

虽然实时操作系统的运行时功能性是一个主要考虑因素，但是如果不涉及开发和调试工具，那么对IMA应用程序开发的讨论也将是不完整的。开发和调试工具的质量对开发时间有很大的影响。针对联合应用程序开发而设计的工具可能并不适合IMA的开发，因为它们需要支持IMA模型和调度模式。

风河风河开物RTOS 653 平台提供了带有基于Eclipse13风河工作台开发套件14的集成开发环境（IDE）。这一最新式的开发环境包括项目配置、代码浏览和编译、风河开物RTOS 653模拟器和目标调试、以及风河系统察看器分析器。图8显示了工作台对运行在ARINC分区中的一个应用程序进行调试的情况。除了风河系统本身提供的功能外，用于开源和合作伙伴工具的Eclipse插件可以进一步对开发环境进行扩展和定制。

动态可视化功能可以对ARINC、POSIX和风河开物RTOS应用程序的行为、分区间的相互作用以及HMS的运行提供图形回馈，从而为应用程序开发人员带来切实的好处。工作台可以用来浏览、导航并理解Ada、ARINC、POSIC和风河开物RTOS应用程序。风河系统浏览器可以识别分区，并且可以显示ARINC进程、POSIX线程和风河开物RTOS任务，对于应用程序内部行为的显示具有重要的作用。如图9所示ARINC进程通过ARINC队列端口进行进程间的通信。

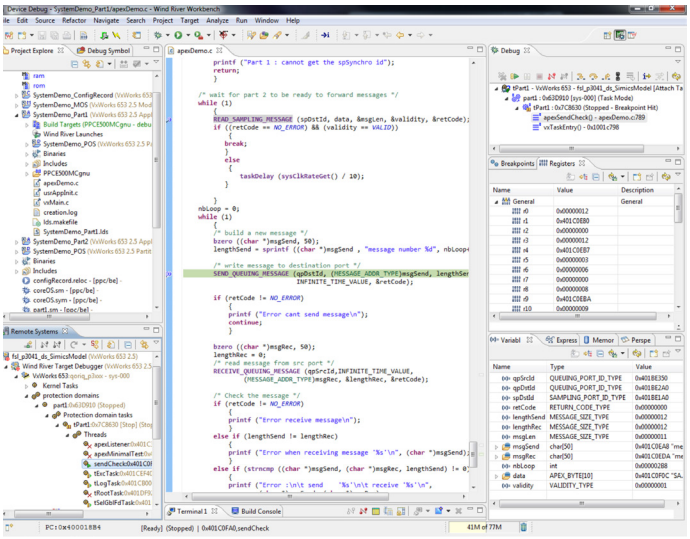

图8：正在显示 风河开物RTOS 653分区调试的风河工作台

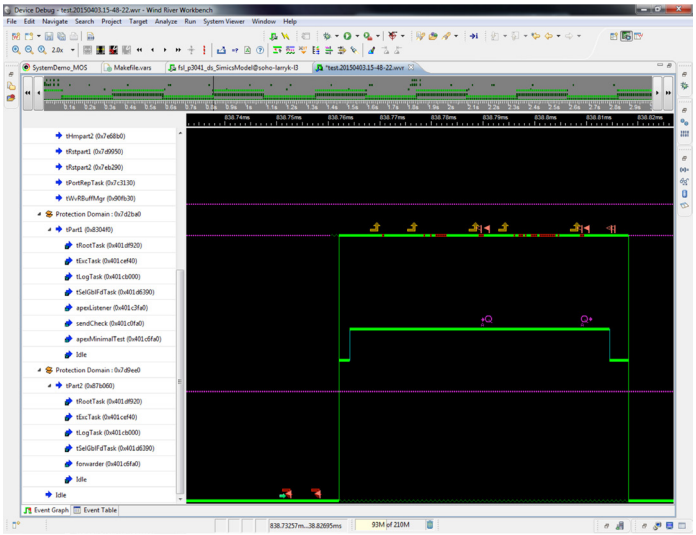

图9：正在显示ARINC分区行为的风河系统察看器

对于ARINC应用程序开发人员来说，很重要的一点是不仅要察看单独ARINC分区内的行为，而且要察看可验证环境中应用程序的运行，以及通过ARINC端口和通道的分区间通讯。这一点通过内置于实时操作系统并经过验证成为风河开物RTOS 653运行时系统一部分的CPU时间使用监控、内存使用监控和端口监控工具来实现。这些RTOS监控器可以和DO-178B验证的主机工具一起使用，用于对信任环境进行测试时显示并记录数据。从开发环境到最终验证的飞行配置，监控器和工具对系统的运行提供了前所未有的洞察力。

联网IMA系统的安全考虑因素

安全问题现已成为航空电子系统中越来越重要的考虑因素（参见Tingey & Parkinson所作的论述15）。通过使用防火墙来限制不同类型子系统间的相互作用，并将飞行系统与OEM系统和航线系统分开，可以实现飞行器级别的安全性16。不过，随着IMA的流行，提出了在这些领域内通过可认证方式增加网络连通性的需求。这一目标带来了一些让人感兴趣的设计挑战。

TCP/IP和相关的网络协议要求花费大量工作进行认证，而系统设计工程师需要在功能性和认证适宜性之间取得平衡。特别是完全TCP/IP堆栈的DO-178B级别A认证是相当繁重的。一些人提议采用专有实现方法，利用一个从属处理器来实现主处理器的网络堆栈。

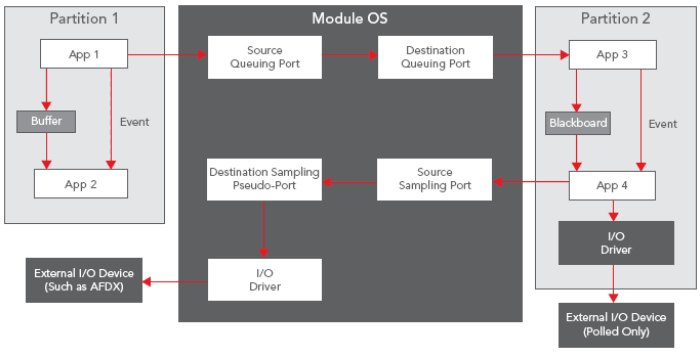

图10：风河开物RTOS 653设备驱动程序模型

不过，这一定制的配置使用了额外的硬件并限制了软件的可移植性。由于这与前面说明的两个IMA目标相抵触，因此只能被认为是一种退步。风河的方法是从风河开物RTOS 653 2.2版开始提供可选的可配置网络堆栈，其中采用了UDP/IPv4功能并针对DO-178B级别A认证而设计。这满足了大多数客户所需的功能性要求，同时依然提供了以后可以添加其他协议层（例如IPv6）的可扩展性。

在核心模块级别也有一些安全因素值得特别关注。例如，运行在处理器用户模式下的应用程序分区无法执行有特权的处理器指令。此外，如果分区操作系统无法满足应用程序的APEX调用，会在执行之前将其传送给模块操作系统进行验证。验证类型包括：分区可视内存范围内的地址验证；边界检查；模块操作系统对象访问权限；以及数据结构完整性/一致性检查。风河开物RTOS 653还提供了可伸缩的系统调用特权机制，其中一个分区比其他分区具有更多的权限，为满足ISO-15408的要求提供了坚实的基础17（参见风河的相关白皮书18）。HMS也有一些相关的安全限制，例如，只有作为系统模式管理器的特权分区可以请求ARINC调度变更。所有这些技术都有助于增强安全性。

IMA系统的安全考虑因素

虽然IMA系统的认证相对来说是一个新的工作，但是很多方面都建立在针对各种认证标准对现有联合系统进行认证的方法上19 & 20。例如，在安全关键性应用程序中通过DO-178B认证文档来重用软件组件的概念已经记录成文，并成功地在一个FAA计划中应用到一个联合应用程序的风河开物RTOS认证中22。

通过使用空间分区在风河开物RTOS 653 中将模块操作系统和分区分离可以将这一概念进一步延伸。目前，一个已经通过DO-178B等级C认证的风河开物RTOS 5.x应用程序可以在同样的IMA平台的单独分区上作为新的DO-178B等级A应用程序使用，而无需对等级C的应用程序重新进行等级A认证。这一技术也可以应用到I/O驱动程序和网络堆栈上（例如TCP/IP）。这些都放置于与模块操作系统和应用程序分区隔离的单独风河开物RTOS 653 I/O分区上。与应用程序分区的通讯使用ARINC端口实现，而与模块操作系统的相互作用则限制为超级用户模式的驱动程序例程。这防止了未经认证的代码影响模块操作系统或应用程序分区的正常操作（参见图10）。

总结

航空电子工业正处于向IMA转变的重大过程之中，而IMA架构和标准的不断发展向标准化组织、原始设备制造商和商业厂商均提出了挑战。

风河提供了一个集成的设备软件平台。风河风河开物RTOS 653平台将一个遵循标准的商业现货实时操作系统和成功开发安全关键性IMA应用程序所需的所有工具集合到一起，不仅提高了开发人员的生产效率，而且保证了认证过程中的复杂性和工作量不会对开发人员产生不利影响。

此外，在IMA环境中对ARINC 653 、Ada、POSIX和风河开物RTOS应用程序的异构支持有助于最大化地重用软件以及将已有的联合应用程序移植到风河开物RTOS 653。

参考资料

1. DO-255, "Requirements Specification for Avionics Computer Resource (ACR)." www.rtca.org

2. ARINC Specification 653, "Avionics Application Software Standard Interface," January 1, 1997. www.arinc.com

3. 风河开物RTOS 653 Platform product page. www.windriver.com/ products/ platforms/safety_critical/

4. Wind River, ACT, and Smiths Aerospace C-130AMP press release. www.windriv-er.com/news/press/pr.html?ID=296

5. Smiths Aerospace Boeing 7E7 Dreamliner Common Core System. www.windriv-er.com/customers/customer-success/aerospace-defense/ smiths787.html

6. EADS/CASA. www.windriver.com/news/press/pr.html?ID=201

7. John Rushby, DOT/FAA/AR-99/58, "Partitioning in Avionics Architectures: Requirements, Mechanisms and Assurance," March 2000

8. ARINC Specification 653-2, "Avionics Application Software Standard Interface," December 1, 2005. www.arinc.com

9. POSIX Specification, ANSI/IEEE POSIX 1003.1-1995; ISO/IEC standard 9945–1:1996

10. Y. C. (Bob) Yeh, "Design Considerations in Boeing 777 Fly-by-Wire Comput-ers." Third IEEE International High-Assurance Systems Engineering Symposium, 1998. http://doi.ieeecomputersociety.org/10.1109/ HASE.1998.731596

11. P. Parkinson & F. Gasperoni, "High Integrity Systems Development for Integrated Modular Avionics Using 风河开物RTOS and GNAT." 7th International Conference on Reliable Software Technologies, Ada Europe, 2002. http://link.springer.de/link/service/series/0558/bibs/2361/23610163.htm

12. DO-297, "Integrated Modular Avionics (IMA) Development Guidance and Certification Considerations." www.rtca.org

13. Eclipse Consortium. www.eclipse.org

14. Wind River Workbench product page. www.windriver.com/products/develop-ment_suite

15. P. Tingey & P. Parkinson, "Avionics Security." Defense Procurement Analysis, Summer 2003

16. Jean Paul Moreaux, EADS-Airbus, "Evolution of Future Aircraft Data Communications." NASA Workshop on Integrated CNS Technologies, May 2001. http://spacecom.grc.nasa.gov/icnsconf/docs/2001/CNS01_Session_ F3-Moreaux-.pdf

17. SO/IEC 15408: 1999, Information Technology—Security Techniques—Evaluation Criteria for IT Security. www.iso.org

18. G. Kuhn, "风河开物RTOS Secure Architecture." Technical paper, Wind River

19. DO-178B, "Software Considerations in Airborne Systems and Equipment Certification." www.rtca.org

20. "Safety Management Requirements for Defense Systems," Parts 1 & 2. Interim Defence Standard 00-56, Issue 3, December 17, 2004, UK Ministry of Defense. www.dstan.mod.uk/

21. FAA Draft Notice, N8110 RSC. www.faa.gov Raytheon WAAS Customer Success Story. www.windriver.com/news/press/ pr.html?ID=393

关于作者

Paul Parkinson 是风河的资深系统架构设计师，他在英国与航空及国防工业的客户一起合作。Paul的专业领域包括集成模块化航空电子设备和智能、监视、目标获取和侦查系统（ISTAR）。通过网址http://blogs.windriver.com/parkinson 可以访问Paul关于航空及国防工业课题的博客。

Larry Kinnan是风河北美公司在航空及国防方面的资深工程专家，负责ARINC 653解决方案的开发。他具有参加众多航空计划的经验，例如767空中加油机、波音787、C130-AMP以及其他商业或军用飞机的项目。在加盟风河之前，Larry就职于医疗设备设计和开发社区，在那里他参加了安全关键性设备的设计、开发和部署。Larry的个人兴趣包括模型火箭、一般飞行、以及商业太空飞行。

关于风河公司

风河公司是全球领先的设备软件优化（DSO）厂商，能够帮助企业客户更快、更好、以更低的成本、更为可靠地开发、运行和管理设备软件。我们的平台是预先集成、完全标准化、企业范围的开发解决方案。从构思到产品的部署，这些平台在设备软件开发流程的各个阶段可以减少工作量、降低成本和风险，并对质量和可靠性进行优化。

风河公司创建于1981年，总部位于美国加利福尼亚州的阿拉米达市，在全球范围展开业务。要了解更多信息，请访问网址www.windriver.com，或拨打电话800-872-4977。

Wind River 就在您身边

北京代表处　北京市朝阳区望京中环南路9号望京大厦B座18层　邮编：100102　电话：010-8477 7100

上海代表处　上海市西藏路585号新金桥广场3-H,I,J室　邮编：200003　电话：021-63585586/87/89/90

深圳代表处　深圳市福田区车公庙天安数码时代大厦A座606室　邮编：518040　电话：0755-25333408/3418/4508/4518

关于风河更多内容请访问：**http://www.windriver.com.cn　Email: inquiries-ap-china@windriver.com**

官方微博　官方微信

WIND